The Concise

UNIX PRIMER 2007

THE UNIVERSITY OF
NEW SOUTH WALES

School of Computer Science and Engineering

Preface

The *UNIX Primer* has a long tradition as being the local guide to the UNIX computing platform in the School of Computer Science and Engineering (CSE), UNSW. The book is primarily intended for staff and students of this University. Embodied in the *UNIX Primer* are the efforts of many who have in one way or another helped either directly or indirectly with this project.

The *Concise UNIX Primer* takes the sections of the *UNIX Primer* specific to the CSE student environment, making it an ideal reference for use in the CSE facilities. The *Concise UNIX Primer* can also be found online at `http://www.cse.unsw.edu.au/help/doc/`.

As with any document that keeps track of local knowledge, things can change quite rapidly. We recommend you regularly refer to `http://www.cse.unsw.edu.au/` on the World Wide Web for updated information and useful resources.

We hope you enjoy the 2007 edition of the *Concise UNIX Primer*. Any comments or suggestions are always welcome and appreciated. They may be sent to the address below or, preferably, by e-mail to `loch@cse.unsw.edu.au` or `helpdesk@cse.unsw.edu.au`

Loc HUYNH (Editor)

School of Computer Science and Engineering
University of New South Wales
Sydney NSW 2052
Australia

February, 2007

Printed February 2007

Typeset using LATEX
Original layout by Ken Robinson
Converted to LATEX2e by Tim Chambers and Geoff Oakley
Converted to *Concise Primer* format by Angie Szczepanik and Geoff Oakley

contents

1 Local Knowledge

1.1 How to Get Help

There is a list of answers to frequently asked questions which is posted on the world wide web at:

`http://www.cse.unsw.edu.au/faq/`

or new site at:

`https://cgi.cse.unsw.edu.au/~csg/`

It is advisable to look at this before looking for help elsewhere. Section 5.5 on page 83 of this book, *The World Wide Web*, explains how to run a web browser so you can see this document.

The *Help Desk* is where you are able to get help with any problems that you might experience with the hardware, the system software, your account, the booking system, etc. It is situated in the Mechanical Engineering Building Undercroft, room G03. The Help Desk is *not* the place to go to when you require help with your assignment or laboratory work.

Help Desk Location:	Room G03, Ground Level, Mechanical Engineering Building

Help Desk opening hours are subject to change, see opening page for latest opening times:

`http://www.cse.unsw.edu.au/~helpdesk/opening.html`

Lab opening times may vary depending on demand and the time of session, and are independent of the Help Desk opening times listed above. Notices will be put up describing any variation. You can also get details of lab opening times on the web for different times of session at `http://www.cse.unsw.edu.au/help/computing/facilities/index.html`.

You can contact the Help Desk in person (in the room listed above), by telephone on **938 55438**, or by email. The Help Desk email address is `helpdesk@cse.unsw.edu.au`, so you can email them by typing:

```
% mail helpdesk
```

See *Electronic Mail* (section 5.1, page 76) for further information on the mail command and other mail readers.

A great deal of useful information can also be found on the Help Desk web page at:

```
http://www.cse.unsw.edu.au/~helpdesk/
```

This page provides information on the types of services provided by the Help Desk, laboratory opening hours, online forms, laboratory maps and usage reports, along with a host of other information.

It is very important that when you go to the Help Desk to have a problem resolved that you take with you your student card and the details of the problem you are experiencing.

Information that may be needed in order to solve your problem includes:

- Your login or user name (or student number — the supervisor on duty can determine your login name with this information).
- The exact name of the machine you are currently logged into if you are currently logged on, or the name of the machine you were using when you experienced the problem.
- A brief and coherent description of what the problem is you want resolved.
- A brief and coherent description of the events that lead to the problem occurring.
- Any error messages that appeared on your console — you should write down the exact error message because an approximation is often not good enough!

You must take some responsibility in getting your problems solved. You can not expect to get appropriate help with your problems if you fail to provide adequate or reliable information.

It is often a good idea to write these things down so that you don't have to chant error messages like a mantra all the way to the Help Desk! Armed with this information, you should approach the Help Desk supervisor and supply them with the required information. If they are unable to solve your problem, they will contact someone who can.

Some problems will not be resolved immediately. If your problem can not be resolved immediately you should be given an estimate as to when you might reasonably expect your problem to be solved. Make sure that you write down the date and time that you reported your problem, the person that you reported it to (the current supervisor on duty at the Help Desk should have their name displayed at the front of the desk) and the estimated time you were given for the problem to be resolved. If your problem has not been resolved in the estimated time, return to the Help Desk,

being sure to take with you the details that you carefully wrote down, and check up on the progress of the problem.

1.2 Where is Everything?

1.2.1 Laboratories

In order to do just about any of the things described in this manual, you will need access to appropriate hardware. At present, we have general purpose laboratories in three buildings.

The location of the laboratories and the type of hardware available in each are listed in table 1.1 and on the School website at `http://www.cse.unsw.edu.au/help/computing/facilities/index.html`. Maps of the laboratory and printer locations are also available in figures 1.1 to 1.4 on pages 5 to 6.

The labs you will be able to use depend on the courses you are doing. For example, thesis students will mainly use the banjo laboratory and students enrolled in COMP1001 will use only the oboe lab.

The restrictions on which laboratories you will be able to use will be enforced by the booking system (see section 1.5 on page 17 for a discussion of the booking system).

1.2.2 Accessing the Labs

After hours access will be required for all buildings and labs if students wish to use the labs after 7 pm on weekdays or any time on weekends. To allow students access to the labs during the entirety of our advertised lab opening times, students may have their Student ID card encoded to provide access during these times.

Students doing a CSE Program or CSE courses will have their access encoded on their Student ID card prior to the commencement of session (provided that these students have a current and valid Student ID card).

General access laboratories in the ME Undercroft (near the Help Desk) and in the EE Undercroft are accessible for the entirety of the advertised lab opening times, so students may use these labs while access is being organised if they have accessible problem.

Please note, there are students who fall in categories for which after-hours access is not granted at all. Please contact the Help Desk or check the following Help Desk webpage for more specific information:

`http://www.cse.unsw.edu.au/~helpdesk/services/swipeaccess.html`

Laboratory Name	Type of Hardware	Operating System	Location	Num
Bell	PC	Linux	J17 G01A	20
Moog	PC	Linux	J17 G01	20
Spoons	PC	Linux	J17 G04A	20
Leaf	PC	Linux	J17 G04	20
Drum	PC	Linux	G17 LG06	20
Harp	PC	Linux	G17 LG08	20
Tuba	PC	Linux	G17 LG06B	22
Oboe	PC	Linux	G17 LG07	20
CHI	Apple	Mac	K17 G11	21
Lyre	PC	Linux	K17 G12	20
Banjo	PC	Linux	K17 G13	23
Oud	PC	Linux	K17 G14	23
Bugle	PC	Linux	J17 G14	21
Pipe	PC	Linux	J17 G15	21
Piano	PC	Linux(Win)	K14 LG18	20
Organ	PC	Linux(Win)	K14 LG19	20
Clavier	PC	Linux(Win)	K14 LG20	20

Table 1.1: Location of laboratories and type of hardware*

*All of the numbers provided are subject to change. Please see `http://www.cse.unsw.edu.au/help/computing/facilities/index.html` for updated information.
*Course specific software may only be installed in certain laboratories. You will be informed of any specific software to be used in any of your courses by your lecturers.

1.2.3 Accessing Individual Computer Terminals

On many occasions individual lab computers will be inaccessible for a number of reasons. An easy way to determine a computer's state is to check the background colour of the login screen:

- A **blue** background indicates that the machine is free for normal use.
- A **maroon** background indicates that the machine has been booked by a subject (lab) class. Only students enrolled in this subject (or those partaking in the lab exercise) can use this machine.
- A **purple** background indicates a personal booking, where only the student who made the booking can use the machine.
- A **black** background indicates a machine which has been removed from the general pool of available computers. You will not be able to use, or book this machine.

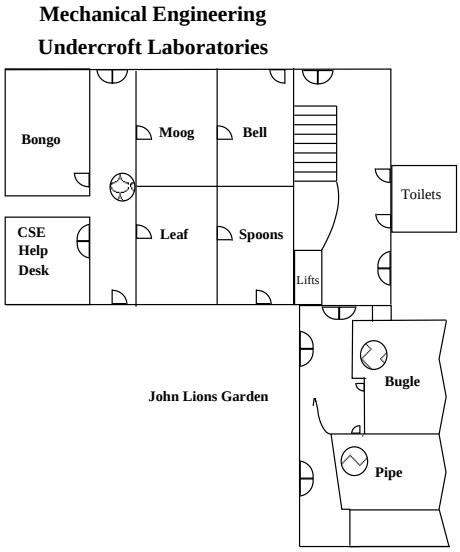

Figure 1.1: The J17 laboratories and printers.

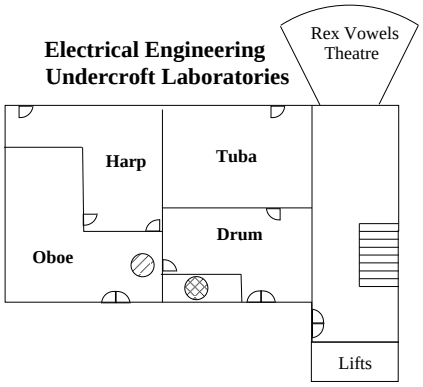

Figure 1.2: The G17 ground floor laboratories and printers.

1.2.4 The Wireless Network

The School provides connectivity for students with laptops equipped with a wireless networking card. These can be used anywhere within the K17 building, on the lawn outside K17 and J17 undercroft.

All Internet traffic associated with your wireless card/laptop will be added to your IP usage in the same way as your Internet usage via any of our lab computers or other servers. Please note, this means that excessive use of your laptop when downloading from the Internet could cause you to exceed your Internet quota. See section 1.9.6 on page 32 for more information about IP Quota.

You will first need to register the MAC address of your wireless card (usually printed on the wireless card itself) and your CSE computer account name. These details will be added to the database of cards allowed to use the wireless network. You will

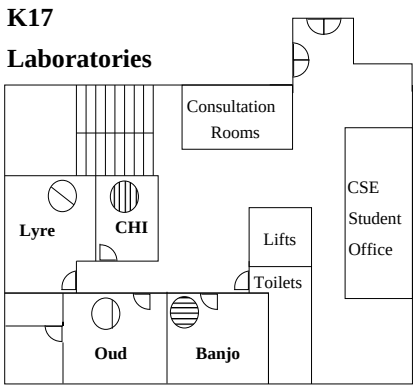

Figure 1.3: The K17 laboratories and printers.

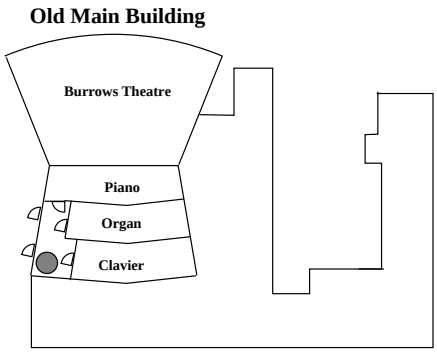

Figure 1.4: The K14 laboratories and printers.

also need to configure your laptop to use DHCP while using CSE's wireless network.

For more information on how to register the MAC address of your wireless card and configure your laptop for a wireless connection, please refer to:

`http://www.cse.unsw.edu.au/faq/questions/network-wireless.html`

1.2.5 The Laptop Lab

The laptop lab is the recommended place for you to use your own laptop. The lab is available in the basement of the K17 building. To take advantage of the network sockets, you will need a network-capable computing device (e.g. laptop with network card) and a cable. You will need to configure the settings of your laptop so that it behaves as a DHCP client and set your internet browser to use the School's proxy in order to get started.

For more information on how to do this, please see:

`http://www.cse.unsw.edu.au/faq/questions/laptop-connect.html`

The laptop lab has a collective IP quota of 15Mb per 15 minutes.

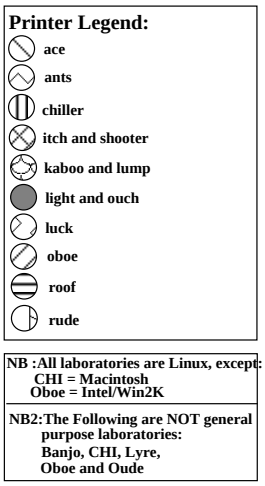

Figure 1.5: Printer legend.

Warning: Do not attempt to plug your laptop any where else other than this labs!

Attempting to connect any machine (including your laptop) other than the lab workstation will result in the switch blocking all connections to that port (including reconnection of the registered workstation), and has been known to result in more serious network related problems.

1.2.6 Printers

There are several black and white laser printers available for undergraduate students. The names and locations of these printers are listed in table 1.2. There are other printing facilities (including colour laser printing) available for staff and postgraduate research students. Maps of the laboratory and printer locations are also available in figures 1.1 to 1.4 on pages 5 to 6.

To determine which printer is closest to you, simply type the following command at your prompt:

```
% whichprinter
```

Plain laser printing in general is quota restricted, each student who has login access will be given a print allocation per session. An extra print allocation may be purchased at the Help Desk. See *Print Quotas* (section 1.9.5, page 31) for more information about your print quota. The colour laser printer is only available to Staff and Research Postgraduate Students. Other students enrolled in courses for which colour printing is required will also be given a quota as required.

Printer	Type	Location	Access
itch	HP9000DN	EE Undercroft	All
shooter	HP8150DN	EE Undercroft	All
lump	HP9000DN	ME Undercroft	All
kaboo	HP8150DN	ME Undercroft	All
luck	HP8100DN	ME Eatery	All
ants	HP8150DN	ME Eatery	All
chiller	HP4100DTN	K17 CHI Lab	CHI Students
roof	HP8100DN	K17 Banjo Lab	Thesis, PGResearch Students
rude	HP815000DN	K17 Oud Lab	Thesis, PGResearch Students
ouch	HP8150DN	Old Main Building	All
light	HP9000DN	Old Main Building	All

Table 1.2: Printer Locations*

*Printer types are subject to change. For updated information please see `http://www.cse.unsw.edu.au/help/computing/facilities/index.html`.

1.2.7 Booking Terminals

Booking Terminals are located inside some general purpose labs as well as areas outside the labs. Currently there are two ternimals in OMB labs, two in Bugle lab, two at Help Desk and two near the Student Office. They are mainly used to make bookings using the `book` and `tkbook` programs, check email or to run a web browser. Details on how to use the `book` and `tkbook` programs can be found in section 1.5 on page 17 or at any booking terminal.

Please note that these machines are only intended for limited amounts of work and should not be used for extended periods of time.

1.2.8 Information Terminals

Information terminals (infoterms) can be found in all major lab areas, and provide useful information pertaining to the current state of the lab. Each infoterm screen is divided into several areas:

- A lab usage summary providing an overall snapshot of the number of free, booked, and used machines in each lab.
- The booking status of individual machines. You can use these to get an indication of how many machines have been booked in the current slot, and to which machine you have been allocated if you have made a booking. For more information on making bookings see section 1.5 on page 17. Particular information about the sky terminal display can be found in *Taking Up Your Bookings* (section 1.5.4, page 20).

- Print queue information indicating the status of printers, how far down the queue your job(s) are, along with the type and size of files being printed. This information is updated every 5 seconds or so as jobs progress up the queue.

  You can view this information from your own `xterm` by using the `lpq` command, the general form of which is:

```
% lpq -Pprintername
```

For more information on the `lpq` command and checking printer queues, please see section 2.2.14 on page 49.

1.3 Accounts

Now that you know where the hardware lives, you will need to obtain an account so that you can use the system.

1.3.1 What is an account?

Essentially an account provides you with an allocation of disk space that is yours to play with, work in, and store your files; and it allows you to interact with the computers. Without an account you will be unable to work on the School's machines. You are responsible for making sure that your account is used in an appropriate manner and is not abused, either by yourself or anyone else! There are stiff penalties for breach of conduct with respect to computer use. Read *Local Etiquette* (section 1.6, page 22).

1.3.2 How to get an account

Initialising your account

At the time your account is created (or at the commencement of each academic year) you will need to activate your account for use. This involves viewing and acknowledging the Yellow Form Policy as well as setting a new password for yourself. This can be done using two different methods. Each are described below.

Using newuser

`newuser` allows you to activate your account from the CSE lab PC.

Figure 1.6: An example of an initial login screen.

1. Find a vacant workstation, sit down at it, and relax. If the screen appears black, move the mouse or press the shift key, this should make your screen spring to life, if it does not produce the desired result, go and seek help. Your screen should appear similar to figure 1.6.

2. Log on to the machine as `newuser` (NOTE: you must use all lower case letters), and press RETURN and requires no Password.

3. A browser displaying the Yellow Form will appear. Read the Yellow Form carefully. Don't just acknowledge without reading, because you are legally bound by the terms. At the end of the page, press the button "ACCEPT YELLOW FORM AND ACTIVATE ACCOUNT"

4. An authentication window will appear. Enter the following:

   Username: `your UNSW loginID`
   Password: `your unipass`
   (Your UNSW loginID is "z" followed by your studentID e.g. z3126789, as you use with myUNSW)

5. Following the instructions on the screen.

   These conditions must be accepted before you are allocated a computer account and you are bound by these conditions for the duration of this account. You should, therefore, take the time to read this advice carefully and understand it before you proceed to accept it. If you require clarification on any of the conditions presented, please see the Help Desk in room G03 of the ME Building.

   Upon acknowledgement of the conditions, `newuser` will allow you to enter a password, which you will use each time you login to your account. For more information on choosing a password, see section 1.3.4 on page 13.

When you have finished with `newuser`, you should be able to log on to the machine using your CSE login name and password within ten minutes. For more information on login names and logging in read *Logging On and Off* in section 2.2.2 on page 38.

When you login for the first time your home directory will be set up for you.

Using newuser.php

`newuser.php` is a web interface that allows you to activate your account using your student number and a valid UNIPASS (password). It has the same functionality as `newuser` but allows you to initialise your account over an internet connection.

1. `newuser.php` can be accessed by going to the this URL: `https://www.cse.unsw.edu.au/newuser.php`
2. You will be asked to enter your student ID and your UNIPASS. Please make sure you add a *z* in front of your student ID, otherwise the `newuser.php` script will reject you.

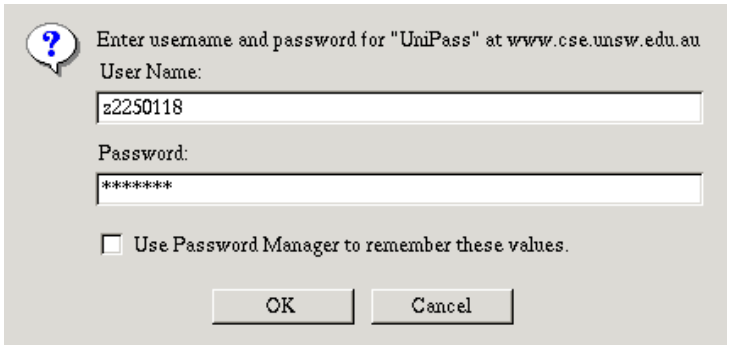

Figure 1.7: Newuser login screen

3. Once your student ID and password have been accepted, read the Yellow Form notices and click on the acknowledgement button. Read the Yellow Form carefully. Don't just acknowledge without reading, because you are legally bound by the terms.
4. Enter your date of birth and set a password. This password will be used every time you log into your CSE account. For more information on choosing a password, see section 1.3.4 on page 13.

   Once your password is accepted, your account will be ready for use. Don't be discouraged if you cannot use your account straight away, as it can take up to ten minutes for your password to be processed.

Figure 1.8: Acknowledge the Yellow Form

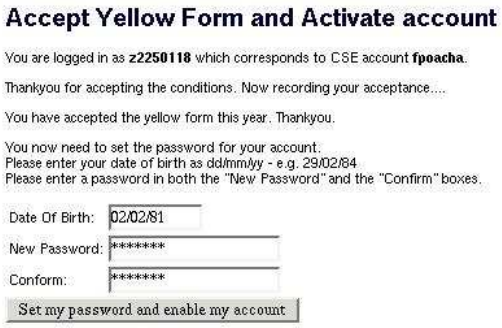

Figure 1.9: Set a password

Those who had an account last year

All people who are doing coursework (which includes all undergraduate and some postgraduate students) should log on using `newuser` in the CSE labs or run `newuser.php` over an internet connection at the beginning of each session. Existing students who have not undergone this process will have their passwords reset, thereby forcing either `newuser` or `newuser.php` to be run.

1.3.3 Thesis Accounts

Being a thesis student brings with it some special privileges. This includes being able to use the Banjo and Oud laboratory, having a larger portion of the available disk space than you would as an undergraduate, and you become a real person and not just a random set of letters and numbers! Yes, that's right. You can have your login changed from the standard combination of your name and random numbers to your name only (well, some variant of your name that makes it a unique login). For information on acceptable login names, please see:

`http://www.cse.unsw.edu.au/faq/questions/account-choosename.html`

To have your account set up as a thesis account you will need to either go to the Help Desk and ask for a thesis account application form or access it from the following webpage:

`http://www.cse.unsw.edu.au/~helpdesk/services/thesis_acc.html`

Fill out this form, making sure that you answer ALL of the questions. Be sure to have your thesis supervisor sign the form before you return it to the Help Desk. The friendly Help Desk supervisor will then update your account details while you wait.

As a member of a thesis class, you will also be entitled to swipe-card and login access to our 24-hour Banjo and Oud labs on the ground floor of the K17 building. This access is finalised after the HECS census date each session.

All of this information assumes you are a CSE student with a supervisor from CSE. If you do not fall into this category, processing times may be delayed. Please visit the following webpage for more information on the steps you may need to take:

`http://www.cse.unsw.edu.au/~helpdesk/services/thesis_acc.html`

1.3.4 Choosing a Password

Your password is the key to your account and all your files. It is therefore up to you to choose your password well, and not to share it with *anyone*. There are serious penalties for disclosing passwords — your account may be suspended and you may be fined. See *Why is my Account Access Denied?* (section 1.6.2, page 23) and *The Online Yellow Form* (section 1.6.3, page 24) for more information on penalties.

Why should you care about the security of your account? Well, when you first registered your intent to take on the mantle of a CSE course, you acknowledged and accepted the conditions as presented by the *Online Yellow Form* (see section 1.6.3, page 24 for more details). By doing so you promised not to disclose your password to another living soul, amongst other things! Apart from this, if someone other than yourself knows your password, they can use your account to read your files, delete your files, send offensive mail as you, attempt to disrupt the system and engage in other anti-social activities for which you will be blamed and held responsible. People who have your password can also use your print and IP quota allocations.

It is your responsibility to choose a secure password. Choosing an 'unguessable' password is not really very hard. It just takes a little bit of imagination. Below are some simple rules and guidelines for choosing a secure password.

1.3.5 Some Rules

- **DO NOT** choose any part of your name, or student number or other information about you that is recorded in the computer.

- **DO NOT** choose first or last names, even if they belong to imaginary people.
- **DO NOT** choose any word that can be found in any dictionary.
- **DO NOT** choose a run of characters from the keyboard; *qwerty* is NOT a secure password!
- **DO NOT** choose any of the above spelt backwards, or with unusual case, or repeated (like *wordword*), or with an extra character added. These are the most obvious things to guess and a computer can generate and check several thousand variations of your name in a few seconds.

Obviously you need to be able to remember whatever password you choose, so just choosing a random string of characters is not necessarily the best approach. One good approach is to choose one or two words that you can remember, and then distort them in some way that is hard to guess. Below are some ideas for distortions. It is a good idea to have more than one distortion as this dramatically increases the time that a computer search would require to have any hope of guessing your password.

1.3.6 Some Suggestions

Make up a sentence that you will remember and use bits of it, like the second character of every second word, or the first character of the first word, the second character of the second word and so on. As an example, using the sentence *'Life is too short to drink bad wine'* the password *litstdbw* can be developed.

Start with a longish word, or a couple of short words (with or without a space).

Insert one or two punctuation characters, !, ?, etc, or other strange characters, +, &, $, etc, somewhere in the word or words, preferably not at the beginning or end. For example *str8_out* or *i+s2!mux*.

Omit some characters of a long word.

Use imaginative capitalisation, such as capitalising the middle letter. Having too many changes of case is not a good idea as it will become wearisome to type.

Misspell the words, such as using 'k' instead of 'c', using 'cw' instead of 'q' or choosing a different vowel. Make sure this is not the only distortion you use, as this sort of distortion is also easy to guess and a computer can be programmed to try misspelt words.

Remember: combining several of these ideas is always a good idea. For example, starting with *'Life is too short to drink bad wine'* we can add some misspelling and numbers: *'Life is 2 short 2 drink bad 'y'-ne'*, and end up with *'li2s2db Y'*.

1.3.7 A WARNING

DO NOT use any of the above examples for your password, that would be very silly!

REMEMBER — Do not tell *anybody* your password.

1.4 Using Classrun

`Classrun` is a command that is used to submit assignments, assignment submission details, and access class records, as well as retrieve marked assignments. The behaviour of `classrun` depends upon the course and option selected:

```
%  1911 classrun
    usage: classrun OPTION [args...]
    where OPTION and args matches one of
        -give assignment [files...]   assignment submission
        -check assignment             show deadlines, submission status
        -collect assignment           view copy of marked submission
        -assigns                      show assignment names
        -sturec                       show class record
```

The `assigns` option allows you to see the full collection of assignment names for a particular subject (COMP1911 course in this case):

```
% 1911 classrun -assigns
    assg1
    assg2
```

Each subject will have a unique naming scheme for its assignments. The command for assignment submissions varies, and is supplied with the assignment itself. For this reason, it is not documented here.

To confirm an assignment submission, or to check submission history for an assignment, use the `check` option. This option also indicates the status of each submission. That is, whether it compiled successfully, or if there were any problems that occured.

```
    % 1921 classrun -check ass4

Assignment: ass4  Submission ID: 1234567

Current day and time:   Thu Feb 15 15:45:42 2007
Assignment deadline:    Fri Feb 16 16:59:58 2007
A submission now would be 1 day and 1 hour early
Event           Day and time              Details
========================================================================
Submission      Fri Feb 16 15:10:25 2007  1234567 wed09-bell ass4 [1 hour early]
Submission      Fri Feb 16 15:40:22 2007  1234567 wed09-bell ass4 [1 hour early]
Submission      Fri Feb 16 16:58:27 2007  1234567 wed09-bell ass4 [on time]
```

```
Testing         Mon Feb 19 19:57:53 2007 kettledrum.orchestra.cse.unsw.EDU.AU

Most recent submission:
-rw-r----- xxxx859/xxxx859 312 2007-02-15 22:20 GRAPH.h
-rw-r--r-- xxxx859/xxxx859  63 2007-01-30 19:07 Item.h
-rw-r----- xxxx859/xxxx859 1145 2007-02-12 18:33 Djik.c
-rw-r----- xxxx859/xxxx859 3112 2007-02-16 22:04 Prim.c
-rw-r----- xxxx859/xxxx859  695 2007-02-16 22:06 gen.c
-rw-r--r-- xxxx859/xxxx859  132 2007-02-16 21:50 make1
-rw-r----- xxxx859/xxxx859  131 2007-02-16 22:36 make2
-rwx------ xxxx859/xxxx859 52610 2007-02-07 19:56 Report3.pdf
-rw------- cs1921/cs1921     155 2007-02-16 23:58 !dryrun_record

Your submission has been marked. Do you want to examine it (y/n)?
```

To obtain an assignment mark, use the `collect` command. In the event an assignment submission has not been marked, you will receive a message similar to this:

```
% 2041 classrun -collect assg2
    Assignment: assg2  Submission ID: 2250118

    classrun: Your submission not yet marked.
    login: fpoacha
    stuID: 2250118
    class: cs2041
```

The `sturec` option provides assignment and tutorial marks as well as other information relevant to the subject. Each subject tailors `sturec` to its own needs:

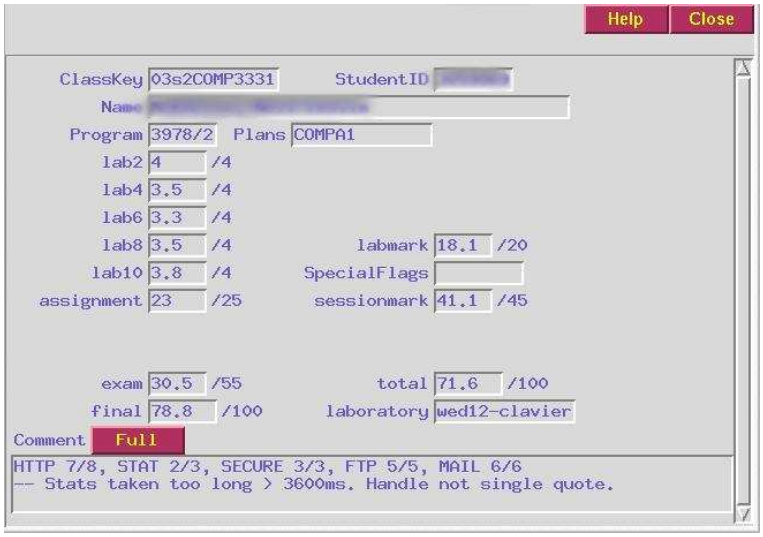

Figure 1.10: An example of a sturec screen.

1.5 The Booking System

Access to the School's computer laboratories is fairly free. The only restrictions are that some labs are restricted to students studying certain courses, and that only one person can use any given workstation at one time. At busy times, this can make it hard to find a terminal to use.

In order to enforce a fair sharing of the limited number of terminals at these busy times, a Terminal Allocation and Booking System (TABS) has been developed. It is a **System** which allows people to **Book** access to a **Terminal** at a particular time in a particular lab, and which will, when the time comes, **Allocate** a terminal to that person, making sure that no-one else uses it.

In order to interact with the booking system, you can use one of three applications called `book`, the text-based user interface to the booking system or `tkbook`, the graphical user interface, or `Control_Panel`, the graphical-web user interface. You are able to use `book` or `tkbook` at any workstation or booking terminal (see section 1.2.7 on page 8 for more information on booking terminals). Using `book` or `tkbook` you will be able to make, inspect, or cancel personal bookings. More information on how to make a booking is provided in section 1.5.3 on page 18.

1.5.1 Booking Tokens

Each student's computer account is allocated a number of *booking tokens* which are needed to make bookings. Each token gives the right to make a booking for one half hour period in one of some selection of labs. The selection of labs you are allowed to book depends on which courses you are enrolled in. After this booking has been utilised or cancelled by either yourself or the booking system (see section 1.5.4 on page 20 *Taking Up Your Booking*), the token is returned so that further bookings can be made.

Each student is given a number of tokens based on the expected assignment load of the courses that they are enrolled in. Courses with a heavier load will be given more tokens, and so students in those courses (who book) will get a correspondingly greater share of available terminal time.

You can find out how many tokens you have by typing `tokens` at the `BOOK >` prompt. You can also find out in the `tkbook` application by clicking on the *tokens* drop down menu or similarly by typing `tokens` at the `BOOK >` prompt within the interface.

1.5.2 Class Bookings

Some courses, typically first and second year courses, run tutorials in the terminal laboratories.

For these tutorials, an entire lab is booked for that course. The booking system may only allow people enrolled in that course to log in (the tutor may impose further restrictions). Such class bookings are different to normal student bookings in that the terminals are reserved for the class for the entire time of the booking. As mentioned below, terminals are only reserved for student bookings for the first 7 minutes.

1.5.3 Making a Booking

There are three programs which may be used to make, view, and cancel bookings.

Book

The first is called `book` which provides a simple, textual, interface: commands are typed in and responses are displayed on the screen.

Students may invoke the textual booking system interface by either:

1. Logging onto any workstation and typing the command `book` at the shell prompt '`%`', (the shell is discussed in later sections) as shown:

   ```
   % book
   ```

2. Logging onto a booking terminal which runs the booking system interface for you.

Students can book certain lab terminals depending on the courses they are enrolled in. If you use `tkbook`, the terminals available for you to book will be displayed.

The following are examples of book commands:

```
BOOK > available 1:30pm - 3:30pm mon,wed
```

This checks the available booking slots for next Monday and Wednesday, from 1:30 pm to 4:00 pm. Note that the time of the beginning of the slot refers to a whole slot; a slot available at 3:30 pm refers to the period 3:30 pm – 4:00 pm.

```
BOOK > book 1:30pm - 3:30pm
```

This books one (half-hour) slot in the period between 1:30 pm and 4:00 pm.

```
BOOK > book 1:30pm - 3:30pm consec=4
```

This books the next four available consecutive slots from 1:30 pm till 4:00 pm.

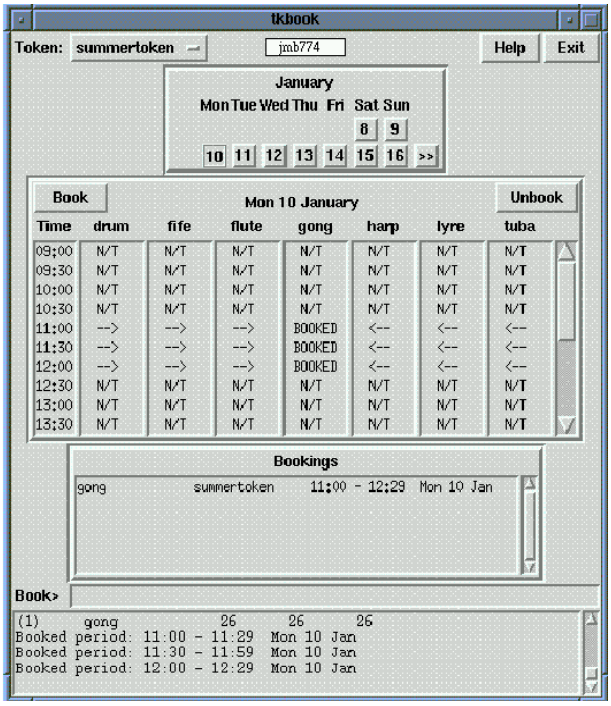

Figure 1.11: The tkbook program.

Tkbook

The second program is called `tkbook`.

It provides a visual, *point and click* interface for booking and unbooking. By selecting a day and time you can book and unbook as well as claim any existing bookings at the `BOOK >` prompt. For more information on claiming bookings, please see section 1.5.5 on page 21. Online help is also available when running `tkbook` by clicking on the **Help** button.

The tkbook program appears as in figure 1.11.

In this particular example, student with login jmb774 has made 3 consecutive bookings in the gong lab on Monday the 10th of January from 11 am to 12:30 pm with her three summer tokens. (The number of tokens are listed in the **Token** box at the top left hand corner of the window). This is done by selecting a day (the default will be the current day) and period with the mouse and clicking on the **Book** button. Similarly, these bookings can be cancelled by selecting the booked period and clicking on the **Unbook** button. N/T means that this student has used all her available tokens and has no tokens left with which to make further bookings at this stage. These tokens are continuously refundable, meaning that when one token is used, it is returned so another slot can be booked with this previously used token.

The `tkbook` program also specifies when a lab is not available for bookings due to class reservations or when a lab has been completely booked by individual students.

URL Control_Panel

The third program is a web interface program. To use this program is simply login onto the page below then click on Lab Bookings icon:

```
https://status.cse.unsw.edu.au/Control_Panel
```

1.5.4 Taking Up Your Booking

Shortly after each booking period, the booking system allocates each user (who has booked for that period) to a terminal in the lab. Where possible, users are allocated to terminals that they are already using, or that are currently unused. If there are insufficient unused terminals, terminals which have been in use for a long time by users who were not booked are the first candidates for eviction.

This means that people who book are less likely to be evicted when their booking ends than people who don't book. Of course if the lab is fully booked, then anyone who is not booked will be evicted.

The allocations that the booking system makes are displayed on the Information Terminals (or infoterms) which are distributed around the laboratories.

For more information on infoterms, please see section *Information Terminals* (section 1.2.8, page 8).

The infoterms list all allocations which have not been taken up, all unused terminals, and all broken terminals. The listing is grouped by laboratory (most infoterms report on several laboratories). Within each group the allocations are listed first, followed by available terminals and finally broken terminals. The allocations show the user's login name, and the name of the terminal that they have been allocated.

In some cases, the name of the terminal will be replaced by the word `CLAIM` which means that the user should log in to a booking terminal, and use the `claim` command to claim their bookings. This is described in more detail under *Being Late For a Booking* (section 1.5.5, page 21).

The allocations for each period are displayed approximately 10 minutes before the period starts, so if you have a booking for 10:30 am in the tuba laboratory, the terminal that you have been allocated will be displayed on the tuba infoterm from 10:20 am until you log on.

When you are allocated a terminal, the person who is logged on to it will be forced to log off by the start of the period, and the terminal will be reserved for you until 7 minutes into the period. During this time, no one else will be able to log in but you.

1.5.5 Being Late For a Booking

After the initial 7 minute grace period your terminal will not be reserved any more. If you have not come to claim your booking, anyone will be permitted to log on.

Further, if you had made several consecutive bookings, the subsequent bookings will only be *tentatively* allocated to terminals. This means that the person logged on will not be evicted, and the terminal will not be reserved. This is done because it is assumed that if you didn't turn up to the first period, you may well not turn up at all. On the infoterm, tentative allocations are shown by having the workstation name replaced by the word `CLAIM` followed by the lab name in parentheses.

If you do turn up late for a booking, you can still claim the booking. This is done by logging on to a booking terminal and giving the **claim** command. You will be told which terminal you have been allocated, and the person on that terminal (if anyone) will be given 5 minutes to leave. The terminal will then be reserved for you for the following 7 minutes.

If you display a pattern of not turning up to claim your bookings, the booking system will consider you to be a *defaulter* and will not reserve a workstation for you. That is, it will only make a tentative allocation, and you will have to claim your booking through the booking terminal. Once you display a pattern of claiming your bookings, your bookings will once again cause workstations to be reserved for you.

If you default twice in a fortnight, you become a *defaulter*. To reinstate your good name, you will need to either wait two weeks or turn up to, and claim two consecutive bookings.

1.5.6 Dishonoured Bookings

It may occur that you have made a booking for a terminal for a particular time slot and your login does not appear in the allocation list of the appropriate infoterm. The booking system makes its machine allocations on a *first booked first allocated* basis using only those machines that are 'up' (ie working) at the time of allocation. Bookings, on the other hand, are accepted on the basis of the number of machines that are physically present in the laboratories. However, the number of machines that are physically present in a laboratory and the number of machines that are 'up' and available for use in a laboratory, may not necessarily be the same at allocation time. For this reason, the booking system will warn users who book the last few available terminals in a lab, that their bookings might not be able to be honoured.

When a laboratory is fully booked and some machines in that laboratory are down and unavailable for use, the booking system is unable to honour all the bookings for that laboratory. The bookings that were the last to be made for that laboratory will not be honoured. The booking system will refund the tokens that were used to make those bookings that are dishonoured.

1.5.7 Help with Book and Tkbook

The help facility of book can be invoked by typing the `help` command at the `BOOK >` prompt

```
BOOK > help
```

or by clicking on the **Help** button in the `tkbook` program.

1.6 Local Etiquette

This section outlines some local conventions and protocols to help ensure that students and staff, can work together efficiently and effectively.

1.6.1 Good Laboratory Behaviour

Despite what some people think, the laboratories are places in which to work. In order to make this working environment as pleasant and stress-free as possible for everyone, it is essential that all students attempt to adhere to a few simple guidelines when in the laboratories:

If you are not actually working at a machine, then you should not be there. Socialising, meeting people or having your lunch on a rainy day in the labs makes for a very unpleasant environment for those who are there to work, so please be considerate and leave.

When you are working in the laboratories, please talk **quietly**. If you need to discuss something with someone who is not next to you, move closer before you have your conversation. That way you are doing your bit to keep lab volumes down, and you can have a well-deserved stretch at the same time!

Please do **not** bring food into the laboratories. Apart from encouraging lingering odours and unwanted insects in the laboratories, you are also putting the equipment at risk. We lose several keyboards each year due to food and drink related accidents in the labs. This puts workstations out of action unnecessarily and creates extended delays in getting our machines in working order for students. Besides, nobody wants to work at a greasy workstation!

If you see someone enjoying their lunch in one of the labs, please report it to the CSE Help Desk immediately. If you are found enjoying your lunch in the labs on more than one occasion, you may be interested in reading the section *Why is my Account Access Denied?* (section 1.6.2, page 23).

Don't bombard the printers with excessively large print jobs, or lots and lots of small print jobs for that matter. Please note that the size of a job on the print queue may

be larger than you orginally anticipated. See section (section 2.2.14, page 49) for more information on checking printer queues. Please also note that the Help Desk superviors will not hesitate to remove excessively large or frequent print jobs during peak times, or at any time they consider a student to be inconveniencing others.

Please treat other people's printouts as you would have them treat yours. Don't steal other students printouts, even when you are desperately stressed and your assignment deadline is looming (see *Originality of Assignment Submissions* in section 1.7, page 24).

Don't tamper with any machine that has an **Out of Order** sign displayed on it in any way. These signs are used to indicate to you that a machine is not functioning as it should be and that support staff are in the process of fixing it. Interfering with a machine that has an **Out of Order** sign on it can often lead to extending the time it takes to get the machine back into working condition.

If you discover a fault with a workstation or printer, or any other piece of hardware in the labs, please do not attempt to fix it yourself. Report it to the Help Desk and they will immediately take the necessary steps to rectify it.

Don't log people off when they are absent from their terminal. They may have just gone to the toilet, to pick up a printout, or to get help.

On the other hand, if you have to leave your terminal unattended it is best to protect yourself against others using your terminal illegally by running a program called `xlock`. This is a screen saver that locks your display until you unlock it with your password. It is up to you to make an effort to prevent others from gaining illegal access to your account. `xlock` is there to help you keep your account safe. It should *never* be used to obstruct others from legally using the machine; or as a means of securing extra time on the machine. For the forgetful, the `xscreensaver` program will automatically lock your screen if it has been inactive for 10 minutes. For more information on `xscreensaver`, see section 4.8 on page 75.

Please do not use the lab computers for non-course related activities (games, irc, web surfing, extended personal email, etc) when there is legitimate demand for machines due to assignment work. During session when the work load on the laboratories is low, non-course related activities are usually fine if you are **quiet**.

1.6.2 Why is my Account Access Denied?

You may be denied access to your account for several reasons, almost all of these involve you doing something wrong. The list of 'something wrong' includes:

- Letting someone else use your account.
- Bringing food or drink into the laboratories; eating, drinking and/or smoking in the laboratories. Such items should be kept *out of sight* at all times!

- Attempting to violate the system security (such as attempting to crack another user's password), attempting to access unauthorised files (these include other users' files and system files), attempting to damage, delete, or alter unauthorised files.
- Misuse of resources or interfering with the equipment in any way.
- Violating any other reasonable directive made by a member of the support staff.

Of course, if you have not done anything wrong, it may mean that a member of the support staff needs to have a chat to you, and switching your account off is an effective way of gaining your attention quickly.

Further information can be found on the CSE Yellow Form. You can read this at `http://www.cse.unsw.edu.au/~studentoffice/policies/yellowform.html`

1.6.3 The Online Yellow Form and University's Rules Relating to Student Use of Computing and Electronic Facilities

Be warned that some of the actions listed above carry penalties including having your account suspended for a fixed period of time and being fined. In serious cases the penalty may result in exclusion from the program or even from University. All this is described in the *Online Yellow Form* which you acknowledge upon activation of your account and which you must accept before your account will be enabled. These are the conditions by which you are bound whilst holding a computing account at the School of Computer Science and Engineering. All students are required to acknowledge and accept these conditions each academic year they are enrolled in at least one computing course.

Warning: please read the *Online Yellow Form* very carefully before accepting it as you will be bound by the terms once you do. The *Online Yellow Form* covers a multitude of things including computer use and academic misconduct.

Copies of the *Yellow Form* (in more specific detail) and the University's *Rules Relating to Student Use of Computing and Electronic Facilities* are also available on the School and University website if you wish to read them online:

`http://www.cse.unsw.edu.au/~studentoffice/policies/yellowform.html`

`http://www.its.unsw.edu.au/policies/policies_home.html`

1.7 Originality of Assignment Submissions

Most courses offered by the School of Computer Science and Engineering require students to submit assignments for assessment in the form of a file containing their

solution. Every student is provided with an account which to work and the student is totally responsible for all uses of the account. In order not to be the victim of dishonest or mischievous actions by others, you should:

1. Not allow any other person to log on to your account (your password should remain known only to yourself).
2. Not leave listings of partial or complete assignment solutions lying around — take them home and dispose of them there.
3. Politely but firmly deny requests to 'borrow' solutions, or to mail files to another account.

1.7.1 Plagiarism

Plagiarism will be suspected if a substantial part of any submission could be converted to part of another student's by purely mechanical means. Checks for this possibility are routinely made on all submissions and comparison methods are improving all the time. Please note that plagiarism checks may not be conducted until the end of the teaching session in some cases.

Submitting an assignment that has been partially or wholly derived from another person's work, with or without that person's consent, is a **serious offence**. *Be warned* that an automatic failure may be awarded if you are caught cheating. In addition, permitting someone else to use your solution as a basis for theirs may result in a mark of *zero* for *your* assignment. Repeat offenders may be referred to the Registrar for consideration for exclusion from the University.

This penalty is determined by the lecturer, and may take the form of penalty or failure for the assignment and/ or course, depending on the severity of the offence.

For more information, please visite:

`http://www.cse.unsw.edu.au/~chak/plagiarism/plagiarism-guide.html`

`http://www.cse.unsw.edu.au/~studentoffice/policies/plagiarism.html`

1.7.2 Collusion

Copying is not the only form of unfair dealing. Except where groups are *expressly formed by the lecturer in charge of a course*, working closely together on assignments is **not permitted**. There are two reasons for this prohibition — firstly, it is not fair for those who work alone to be judged on the same basis as a co-operative effort; and secondly, it is unusual for equal contributions to be made in cases of collusion.

1.7.3 Similarity Detection and Followup

Assignments submitted for assessment are regularly checked by similarity detection software. Although not all assignments are checked, many cases of suspected copying are uncovered. Spot checks are also conducted.

In the event that unreasonably similar assignments are detected, the students concerned may be asked to complete a personal questionnaire. Completed questionnaires are considered as part of the process of determining any penalty to be applied. Lecturers may also interview students about assignment similarity.

Remember: Plagiarism is covered in the *Online Yellow Form* you accepted, so you cannot claim ignorance.

1.7.4 A Word of Advice

Many students panic when an assignment deadline passes and their program still doesn't work. The temptation to rummage through a waste paper bin or to copy from a friend is very strong. *It is not, however, worth the risk of failing the course.* If you are placed in this position, please let your tutor know of your current difficulties via electronic mail. You should also attend a scheduled consultation for the course, who you may be able to get help.

The best advice, however, is to begin planning for an assignment as soon as the specifications are available; to be confident the design is logically correct *before* using the terminals; and never to underestimate the time needed to complete the work.

1.8 Loans From the Help Desk

In general, you can borrow from Help Desk software CDs and notes are available and vary depending on course content each year as developed by your lecturers. Other materials may also become available from time to time and any software available relevant to coursework will be announced in lectures.

You can ask for further information at the Help Desk at any time.

1.8.1 Where is it?

The Help Desk is located in room G03, ground level of the Mechanical Engineering building. There is more detailed information in *How to Get Help* (section 1.1, page 1).

1.8.2 Who Can Borrow?

To be eligible to borrow an item you must have both:

- a valid account on the School's machines; and
- a valid student card (you cannot borrow using another student's id card).

Some software items have further restrictions placed upon them, such as required enrolment in a particular course.

For more information on software loans from the Help Desk, please see

`http://www.cse.unsw.edu.au/~helpdesk/services/software_loans.html`

1.8.3 How to Borrow

To borrow an item follow these simple steps:

1. Decide which item it is you want to borrow.
2. Go to the Help Desk, making sure you take with you your student card.
3. Ask the Help Desk supervisor for the item you would like to borrow.
4. Present your student card. The item you have requested will be handed over to you (depending on availability) after the loan has been recorded on our database.

Generally, items are loaned for a period of two working days. You will be informed of the loan period when you are issued the item. After this time the item must be returned to the Help Desk. When you return the item you have borrowed, in the same condition in which you borrowed it, your loan will be removed from our database.

Since many items are in high demand, overdue loans will result in an automatically generated e-mail addressed to yourself and the lecturer-in-charge of the resource in question.

1.8.4 Responsibilities

This is a service provided to you in an attempt to make your life as a student easier, more efficient and more productive. You as a user must share the responsibility of keeping the item(s) you borrow in good usable order. As a user of this service your responsibilities include:

- Keeping the item(s) that you borrow in good usable order. Don't remove pages from manuals, books, notes, etc. If you would like a copy of a page go and photocopy it! Don't write on or use highlighter pens on any of the items you borrow. You can treat your own things any way you choose but please treat borrowed items with respect so that other people can use them in pristine condition.

- Return items that you borrow on time, someone else may be waiting for them Any student holding an item of demand for longer than the permitted loan period will be referred to the Lecturer in Charge of the relevant course.

- Notifying the Help Desk supervisor if any item is damaged or is missing pages, etc. Before you borrow an item make sure that the item it is in good condition, no pages written on or missing. When any item is found to be damaged the person who was the last to borrow the item will be held responsible for the damage and will be asked to replace the damaged material.

1.9 Other Useful Things to Know

1.9.1 Restoring Files

Incremental backups of the filesystem are made each night. These backups record any files which have changed since the previous incremental backup. If you delete something and then realise that you want it back, you can restore it yourself from the incremental backups by using the `tkrestore` command.

The `tkrestore` command is very useful and simple to use. To run `tkrestore`, simply type

```
% tkrestore &
```

If you are unsure of how to use it, there is a good explanation available by clicking on the *help* button in the `tkrestore` window.

Please note, any changes you made to a particular file on the day you restore it will not be salvaged, it will only be restored as the file it was backed up as on the previous night. Any files which are more than six months old will not be listed in `tkrestore` but will need to be extracted from full backups. You will need to contact System Support (email *ss*) in order to have this done.

More information about backups can be found at:

```
http://www.cse.unsw.edu.au/faq/questions/account-backups.html
```

1.9.2 Accessing Your Floppies

When you are using the PC based workstations, you can access the floppy drive using a set of software tools for manipulating MS-DOS or Windows files, called `mtools`. Using these tools you are able to copy MS-DOS files to your UNIX environment, and back again. For example, to copy a MS-DOS file called `ass1.m` from a floppy to the current directory type:

```
% mcopy a:/ass1.m .
```

To copy a file called `ass1.m` from your current directory to the top directory on your floppy disk, type:

```
% mcopy ass1.m a:
```

Note that the conversion between UNIX and MS-DOS format is handled for you.

To get a directory listing of the floppy disk, type:

```
% mdir
```

These tools are discussed further in *Floppy Disk Commands* (section 2.3.3, page 56). There are many other tools available in the `mtools` suite. To find out more about them have a look at the online manual pages.

```
% man mtools
```

1.9.3 Accessing Your USB memory-stick/drive

Plug the USB stick in the USB slot on the front of the computer then mount your USB stick by typing in an xterm:

```
% priv usb mount
```

To access all files on your USB stick, type in an xterm:

```
% cd /mnt/usb/N
```

where N is the number that was displayed when you mounted the USB stick.

To remove your USB stick, type in an xterm:

```
% priv usb unmount
```

Now you can remove your USB stick safely. You are advised to **unmount** the USB stick before remove it otherwise the USB file allocation table (FAT) file may be corrupted, you may end up with lost all the files that you save onto the stick.

1.9.4 Disk Quotas

Due to the unfortunate fact that resources are limited, each student has a fixed amount of disk space available — a disk quota. Undergraduate students are generally allocated fifty megabytes (50MB) plus twenty megabytes (20MB) per course. This should be sufficient for all of your coursework. Some courses allow you more space because they involve the use of particularly large files. You can check your quota using the `rquota` command, which produces an output along the lines of:

```
    blocks   quota   limit  #warns   files   quota   limit  #warns
      3496    5048   10096             169   10480   21480
```

'`blocks`' is the amount of disk space you are currently using, '`quota`' is the amount you are allowed indefinitely ('soft block limit') and '`limit`' is the absolute maximum you can use ('hard block limit'). When you exceed your soft limit you will receive a warning email informing you of how much time you have to reduce your usage before your soft block limit is *enforced*. When this limit is enforced you will not be able to create any more files, receive email or, in fact, do much except delete or compress files to reduce the amount of space you are using. When you exceed your quota you have approximately one week to get it back under control.

Similar to the disk space quota is the quota on the number of files you may have. This is shown in the second half of the `rquota` output and works in much the same way as the disk space quota. In both cases '`#warns`' is the number of days left before the soft limit is enforced.

It is extremely easy to fill up disk space and exceed your quota without realising it. Any file automatically generated by the computer can quickly grow very large, as can graphics files (especially in *postscript* or *bitmap* formats). Copious amounts of email, or very large email messages and/or attatchments will quickly eat up space while they sit in your mailbox. Web downloads, particularly images and sound, should be treated with extreme caution.

You can get more detailed information by using the `disk_guess` command which will list your six biggest files and six biggest directories. If you run `disk_guess -p`, you will also get a listing of recent file size changes.

Running `disk_guess` will produce an output similar to:

```
You are currently using 2264K which is below your quota of 10240K
Your 6 biggest files are:
  208K ( 9.19%) in ./.netscape/cert7.db
  148K ( 6.54%) in ./.netscape/cache/1C/cache3B14957C003044B.pdf
   48K ( 2.12%) in ./.incoming-mail
   40K ( 1.77%) in ./.netscape/history.db
   36K ( 1.59%) in ./Library/TEX/SimpleLibrary.mch.prf.0.dvi
   32K ( 1.41%) in ./.netscape/cert5.db

Your 6 biggest directories are:
```

```
  592K (26.15%) in ./.netscape
  380K (16.78%) in ./GNUstep
  336K (14.84%) in ./GNUstep/Library
  328K (14.49%) in ./GNUstep/Library/WindowMaker
  288K (12.72%) in ./JavaFiles
  244K (10.78%) in ./Library

You have 732K used up in dot files (or directories) larger than 20K.
The dot files are:
   92K in ./.kde
  592K in ./.netscape
   48K in ./.incoming-mail

You have 200K (8.8%) used in netscape cache files. These are almost certainly unnecessary.
To remove them, run rm -r .netscape/cache/*
```

`core` files are another easy way to consume disk space. When a program crashes, it will often 'dump core', creating a `core` file containing information about the state of the program when it crashed. `core` files are often very large (megabytes), but are also very useful when trying to work out what went wrong. If you don't intend to use them for this purpose, they can be safely deleted by using the `rm` command.

```
% rm core
```

1.9.5 Print Quotas

One thing you should know about printing — there is a limit to how much you can print. This limit is called your *print quota* and it is the number of pages you are allowed to print. The command `acc` (demonstrated in section 1.9.7 on page 32) will show you your current print quota.

Your print quota is determined by which courses you are enrolled in. Undergraduate students are allocated 100 pages plus 150 pages per course. You will be given the 100 pages and 20% of the course based pages prior to the HECS census date for that session. The remaining 80% of the course-based quota is issued after the HECS census date (when enrolment information is confirmed). Your quota is renewed every session. Some courses will also give you a *colour print quota*.

A 'page' of print quota actually refers to a printed side of a sheet of paper, not to an entire piece of paper. Thus, a sheet of paper printed on both sides counts as two pages as far as print quotas are concerned. (The cost of the paper is actually only a small proportion of the total cost of printing. Other costs which outweigh the cost of paper include the toner and maintainence of the printers.)

Try to be conservative with your printing. If you can read it on the screen, why do you need a printout? Be very careful not to print binary files (such as anything called `a.out`). They will probably be rejected; but they may appear as many pages of random gibberish and each page will be deducted from *your* print quota. Any quota consumed in this manner *will not be refunded*.

If you want to print more than your quota will allow, you can purchase more print quota. You can collect a form from the Help Desk, which you take to the University Cashier in the Chancellory (C22 on the University maps). Pay at the Cashier, and take the receipt back to the Help Desk staff, who will increase your printer quota. For $1 you receive 10 pages, for $10 you receive 100 pages.

You can check your print quota usage by typing `acc` in an xterm window. Alternatively, you can view your entire printing history by using the command:

```
% priv pquota -v
```

1.9.6 Internet Quotas

The School has in place a system for imposing quotas on Internet usage. This means that once you exceed your quota you will no longer have WWW, FTP, SSH or other IP-based access to sites outside of the University. Unfortunately Internet access costs the School money and we must seek to encourage users to only download material relevant to their coursework.

You can view your current quota using the `ipq` command on any UNIX workstation. You are only allocated half of your Session IP Quota at the start of the session. The other half is allocated after the HECS census date, similarly to your print quota. It is also possible for you to purchase extra quota in the same manner as you purchase print quota. The form needs to be obtained from the Help Desk, extra quota needs to be paid for at the University Cashier and the receipt returned to the Help Desk for quota allocation and update. There are limitations imposed on Internet Quota purchases, and as a result, any additional quota your purchase must not exceed you *base session allocation*. Your purchase quota limit can be obtained by running the command `priv ipq_explain username` on yourself.

For more information about Internet quotas and purchasing conditions, please see the Help Desk or `http://www.cse.unsw.edu.au/~ipq`.

1.9.7 The acc Command

You can find out a lot of useful information about your account by using the `acc` command, which will produce something like the output illustrated below. Your disk quota is given as 'Disc Limit', about half way down, and is measured in kilobytes. Print Quota is measured in pages and is updated several times a day. Your Session IP Usage, updated nightly, is at the end.

```
% acc
            User Name : jmb077
                  Uid : 9876
              Expires : 27 Feb 2001
```

```
          User classes : 3648_Student, COMP1011_Student

                  Name : Josephine Bloggs
    Encrypted Password : viDPjb.eZeTng
Password last changed : 99/05/07.12:55:47
        Home Directory : /import/bizet/1/jmb077
            Disc Limit : 5048
           Login Shell : /usr/local/bin/ksh
       Last Login Host : ives
       Last Login Time : 99/12/21.18:48:30
        WasteBasket Uid : 65619
 Printer Usage Status : Post-census Allocation       400
                      : Used                         112
                      : Available                    288
                      : set at 12:28 AM 23/Dec/1999
        Daily IP Quota : 56.0Mb
     Session IP Quota : 280.0Mb
     Session IP Usage : 0.8Mb
```

1.10 Computing at Home

Many people have a computer at home and find it useful to work on that machine rather than coming in to the University to use the labs on campus. If you wish to do this, it is very convenient to be able to log in to the School's computers using a modem and have access from home to your files at University. To do this you need to SSH to the CSE computers. Modem and SSH access are described in the sections below.

In order to be able to work on the assignments which are set in most CSE courses you need to have some specific software — e.g. a web browser and the java development kit. This software is large (many megabytes) and can take some time to download from its original source. For your convenience we have established a web page with links for downloading most of the software you will need from our computers. Using these links will be the fastest way for you to download this software. The URL for the web page is

`http://www.cse.unsw.edu.au/~homecomputing/`

This software is also available on the Homecomputing CD-ROM, which can be borrowed from the Help Desk.

1.10.1 Modem Access

Modem access to the University is via the UNSW Dial-Up Service (UDUS) which is run by the Division of Information Services (DIS). This modem facility is only available to staff and students at UNSW. UDUS provides access to the University's

Wide Network resources and the Internet. The bad news is that this is a 'user-pays' service.

But the good news is that there are no other fees. To find out more about the UDUS service, have a look at the DIS-Connect web page at:

`http://www.disconnect.unsw.edu.au/`

Students enrolled in CSE courses will be given two hours of UDUS access per week per course (paid for by the School) for each week up to and including StuVac. The payment to UDUS is made in two blocks: in the first weeks of session, the amount available is the total for the period up to the HECS census date, after which the rest of the session's UDUS allocations are made. The allocated funds may be used until the end of the exam period.

Now before you complain that the quota is not enough, approximately two hours per course per week is considered adequate for course related activities. There is nothing to stop you from buying extra time from DIS if surfing the WWW is your passion. Our advice is to use the alloted time sparingly because you can never be sure when you'll need it for a major assignment sometime during the session.

All students are provided with a UDUS computer account, which is separate from any account you may have on the School's computers. *To find out your UDUS account password (UniPass) you need to visit the DIS-Connect desk on Level 2 (Room 231) of the UNSW Main Library Building, opposite the Loans Desk.* To use your School account from home you must first connect to your UDUS account and then `ssh` to one of the School's computers (see section 1.10.2 below for information on using `ssh`).

Staff and Research Postgraduate Students are now advised to use UDUS because the school does not provide modem pools any more.

Connecting to UDUS

To connect to your UDUS account you will need a computer (PC with Windows9x or higher, or NT, or XP, or Vista or Linux, or a Mac), a modem and relevant software. UDUS will provide this software (with the phone number included) for you on a floppy disk or CD-ROM if you visit the DIS-Connect desk on level 2 of the UNSW Library. This is also where you need to go to buy extra time from DIS if you find that the allocated credit per course per week is not enough.

1.10.2 SSH Access

Once you have logged on to a computer (using UDUS or some other Internet Service Provider) you need to connect to the School's computers to use your School account (*Note:* Your School account is separate to your UDUS account). To access your

School account remotely you can use the `ssh` command. SSH (Secure SHell) is a replacement for Telnet that provides secure encrypted communications between two hosts over an insecure network (such as the Internet). SSH also has many more options than Telnet.

Note: As of Session 2 2002, the use of `telnet` to remotely access the School's servers is not supported. You must install a SSH client on your computer and connect via SSH. There are a number of free SSH clients available. We recommend OpenSSH (Unix/Linux), PuTTY (Windows), SSHWin (Windows) and MacSSH (Macintosh). Information on where to download these clients is available at: `http://www.cse.unsw.edu.au/faq/questions/ssh-install.html`

The general use of the `ssh` command is:

```
% ssh -l <username> login.cse.unsw.edu.au
```

at a shell prompt. For example:

```
% ssh -l fpoacha login.cse.unsw.edu.au
```

If you are using a GUI client such as PuTTY then enter `login.cse.unsw.edu.au` for the destination.

For more information on using PuTTY to SSH to CSE, please see

`http://www.cse.unsw.edu.au/~helpdesk/documentation/Putty.html`

As usual, there is a `man` page for the `ssh` command which you should read. All of the GUI clients mentioned above support most of SSH's many options and have good documentation to help you configure them correctly.

After running the `ssh` command and connecting to one of the School's host computers you will be presented with a login prompt at which you should enter your username and password as if you were using the computers from a lab on campus. The SSHWin client has a "Connect As" option, so you will only be prompted for a password.

Machine Name	Hardware	Operating System
wagner	PC	Linux
weill	PC	Linux
williams	PC	Linux

Table 1.3: Computers available to `ssh` to.

Please note, this information changes periodically. For updated and further information please refer to: `http://www.cse.unsw.edu.au/help/computing/facilities/index.html`

Editing via `ssh`

When you are working from home, you need to use an editor which won't consume too much bandwidth or things will become very slow. Some popular editors which people use during `ssh` sessions (and at other times) are:

pico A simple, display-oriented text editor based on the Pine message system composer.

emacs A powerful text-based editor, discussed in more detail in section 2.2.7 on page 41.

vi Also a powerful text-based editor, discussed in more detail in section 2.2.7 on page 41.

1.10.3 More on SSH

The School has configured the `sshd` servers on its hosts such that you can simply login as above and do your work; however, you may find it easier to do your work if you explore some of SSH's features. Some of the more interesting features are discussed in the FAQ at

`http://www.cse.unsw.edu.au/faq/ssh.html`

For instance, you can use SSH to do the following:

- run X applications at home (keep in mind that this can be very slow over a modem connection)
- charge your web traffic to your CSE ipquota instead of your ISP
- securely copy files between your home computer and your CSE account
- securely FTP files between your home computer and your CSE account

More advanced users can use SSH's concept of `identity` to generate SSH `keys` and thus move between different CSE accounts and hosts in an authorised and secure fashion. This is particularly useful for tutors and subject administrators who need to access the subject class account.

2 An Introduction to UNIX

This section attempts to help you become familiar with some of the basic concepts of the UNIX operating system, and it attempts to give you a very basic 'toolkit' that will enable you to work in a UNIX environment. Do not expect to become an instant UNIX guru after reading and working through this section; you will only be disappointed for there remains much to learn. It is hoped that this introduction will stimulate sufficient interest and enthusiasm in you to prompt you to seek more information for yourself.

2.1 What is UNIX?

Very simplistically, the UNIX operating system is a program that manages and controls computer resources and peripheral resources for users. The UNIX operating system was originally written at the Bell Laboratories in the United States. Today there are various 'flavours' of UNIX supplied by different vendors. The 'flavour' of UNIX on the School's machines is *GNU-Linux*. Although each 'flavour' is slightly different from the other, and you have used more than one, their differences should be transparent (you should not notice any differences) to you as a user.

2.2 The Basics

The UNIX operating system offers a large array of facilities and tools to its users. For the most part your time will be spent on the following types of tasks:

- Using an editor to type in programs and data and edit them.
- Compiling your programs.
- Running your programs.
- Printing the contents of your files on one of the printers.
- Displaying the contents of files on your terminal.

- Listing the files in your directory hierarchy.
- Reading and sending email.

Some simple UNIX commands and facilities that will allow you to do the above tasks will be covered in this section. What is covered in this section is by no means the definitive set of UNIX commands, merely a set of commands that will get you started without too much aggravation.

Further information relating to the use of shells can be found in the section starting on page 61.

2.2.1 Your Login name

As of late 1999, students' user names changed from using student number following a change in University policy. Now, your login consists of a combination of the letters of your (first, middle and last) names followed by three numbers selected at random. This login name is revealed to you when you use SIRIUS. You are able to change it but you should make note of this login name at least long enough to be able to do so. Fear not if you have forgoteth — the Help Desk can assist you!

Those with existing accounts from previous years will still use the login name consisting of their student number; however, they do have the option of changing it to be consistent with the new policy.

Information on how to change your login name is available at:

`http://www.cse.unsw.edu.au/faq/questions/account-changename.html`.

2.2.2 Logging On and Off

The first thing you will need to know is how to log on and log off using your allocated user (or login) name.

At the login screen (which will appear similar to the one shown in figure 1.6 on page 10) type `<your user name>`, with no spaces (e.g. `pats777`). Remember to use lower case letters, and press RETURN. At the `Password` prompt, type your password, and press RETURN; note that when you type your password the characters you type are not shown on the screen — this is a security feature!

If you make a mistake typing your username and password, you will get the message 'Login incorrect', or something similar. Should this happen, just try again. When you enter your username and password correctly, you will see several windows on your screen. These are explained in more detail in *The X Window System* (chapter 4, page 68). For now, just move the mouse pointer over one of the larger windows (known as `xterm`s) to enter UNIX commands in that `xterm`.

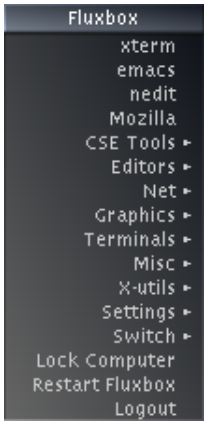

Figure 2.1: Right mouse button menu.

Logging off is simple, you simply move the cursor onto the background window, hold down the right-most mouse button, and select the **Logout** option from the menu bar. The menu bar should look like figure 2.1.

2.2.3 Files

All the world is a file to UNIX; programs, text documents, input and output devices such as terminals and printers are all represented as files in the UNIX world. Files are a very convenient way of storing and managing information. All files have a name, contents that can be written or read, they have an owner, and a place in which they can be stored and from where they can be retrieved.

2.2.4 Directories

A directory is simply a file that contains the names and locations of other files. They are called folders on Windows and Macintosh systems. Since a directory is just a file, it may exist under another directory along with 'normal' files. A directory may contain a directory which in turn contain other directories and so on. A directory that exists under another directory is called a sub-directory. See figure 2.2.

2.2.5 Home Directories

Each registered user is assigned a home directory. This ensures that files that are created by the various users of the system can be distinguished from one another. Placing all the files that belong to a particular user into their home directory allows the operating system to distinguish that particular user's file called `stuff` from any other file with the same name.

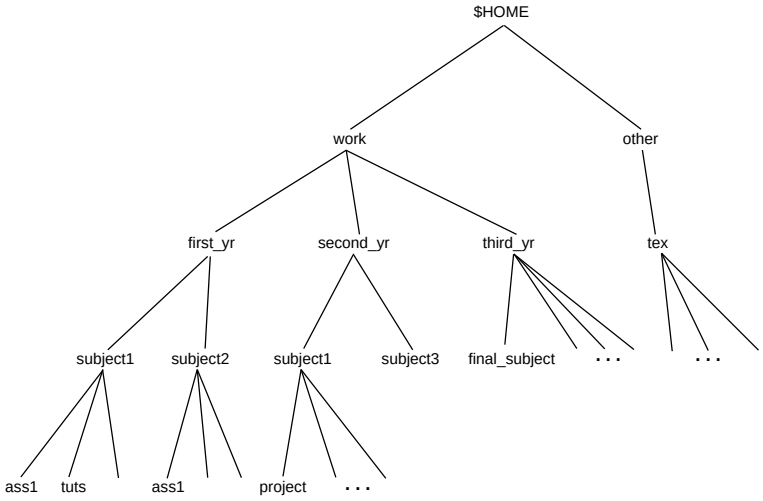

Figure 2.2: Example of what a directory tree looks like.

The name assigned to your home directory is the same as your login name. It is within your home directory that you will create and manipulate your own files and directories.

Your home directory may exist anywhere on our system, but it can always be referred to as `/home/<username>` (eg, `/home/pats777`). **Never** use a name such as `/tmp_amd/bizet/import/1/angies`. It may work today but is unlikely to work in the future. If you need to know where your home directly *really* is located (maybe to track down a problem with one of the fileservers), the `acc` command will tell you.

Most shells also use the tilde character (`~`) to refer to a home directory. `~/file` will be in your home directory; `~zorro/otherfile` is in Zorro's home directory.

2.2.6 The Current Directory

When you log on to a UNIX computer, the operating system sets your home directory as the current directory or working directory. The current directory is the directory under which files and other directories may be created and deleted and is a point of reference for many UNIX commands (this concept will be explored further a little later in this section).

If you ever get lost, the command `pwd` will display the name of the current directory. Unfortunately, it can return confusing names such as `/tmp_amd/bizet/import/1/angies`. Get used to translating this to `/home/angies` in your mind.

The current directory is represented by the character '**.**'. The parent directory of the current directory is represented by the string '**..**'. This may seem strange at the moment, but all will become clear a little later in this section, so read on!

2.2.7 Creating a File

You will normally use an editor to create your files. There are several different editors available for your use on the CSE machines. These include **nedit**, **pico**, **nano**, **ed**, **ex**, **vi** and **emacs**. The editor that you choose to use is entirely up to you and may change depending on the task. For the moment we will use one of the simplest, and you may think most primitive, editors that is available on all UNIX systems, regardless of flavour — **ex**.

To create a file called `stuff` with some text in it, log on as described above, and proceed as follows (NOTE: words within '()' pairs are there as comments to the reader and should not be typed when trying out this example.)

```
% ex stuff            (this invokes the ex editor)
"stuff" [New file]    (message to indicate that stuff is a new file)
:a                    (command to ex to add, or append, text to file)

Now you type in any text
that you want, it does not
really matter what it is !

.                     (the '.' signals the end of input)
:wq                   (write out the file and quit from ex )
"stuff" [New file] 3 lines, 78 characters
```

For the moment we shall dispense with an explanation of the `ex` syntax; it will all become clearer should you choose to pursue your knowledge of `vi`.

Another simple and quick way of creating a file is to use a combination of a couple of simple UNIX commands, `echo` and `>` as shown below.

```
% echo "Now you type in any text" > junk
```

The `echo` command echoes its arguments to *standard output*, which is the terminal, by default. The output from this `echo` command is redirected to a file, in this example `junk`, using the `>` character. Input/output redirection is described in more detail in section 3.8, page 65.

You could also use the `cat` command to create a file as shown below:

```
% cat > myfile.txt
Again you type all the stuff that you want to
and then to signal the end of the input
you hold done the control key on the left hand
side of the keyboard, and press the d key.
^d
```

In this example, the `cat` command reads its input from standard input. The output of the `cat` command is redirected to the file `myfile.txt`. The Control-d (`^d`) signals the end of input to the `cat` command.

nedit is a commonly used basic editor with a *point and click, pull down multiple menus* type of interface. It is simple to use but its functionality is limited. **emacs** and **vi**, on the other hand, are full screen text editors which, with experience, will allow the user to work in a much more efficient and productive environment than basic editors offer. They both have a steep learning curve for advanced usage and are extremely useful and powerful once this has been established.

2.2.8 Creating a Directory

This is a very simple procedure. To create a directory called `temp` execute the following command:

```
% mkdir temp
```

Done! It is very easy to do, and with some thought you will be able to keep your work well organised by creating a few appropriate directories. As a 'rule of thumb', if you have more than a screenful of filenames in a directory, it is probably time to divide it into a couple of subdirectories.

2.2.9 Manipulating Files and Directories

Now that you have a few files and a directory in your home directory we will use these to explore some simple UNIX commands that you will grow to know and love.

pwd Print the name of the current working directory

usage: `pwd`

cd Change the current working directory

usage: `cd directory-name`

ls Lists the names of files in the current directory.

usage: `ls` [`filenames`]

cp Makes a copy of a file.

usage: `cp source-filename target-filename`

The `cp` command will copy the contents of the file `source-filename` to the file `target-filename` overwriting the contents of `target-filename` if it exits.

mv This renames a file.

usage: `mv old-filename new-filename`

rm Removes file. (Careful — you can't 'undelete' files. See section 1.9.1 on page 28 for information on restoring files from backups.)

usage: `rm filenames`

To remove a directory use: `rm -rf filename`

rmdir Removes a directory, the directory must be empty.

usage: `rmdir directory-names`

An example session to demonstrate these commands follows:

```
% ls
junk.txt     myfile.txt     session1.txt    stuff.txt
```

List all the files in the current directory, note that files beginning with a '**.**' are not listed. The -a option allows you to list, rather than hide, the entries starting with a '**.**' and the -l option allows you to use a long listing format, shown below.

```
% ls -al
total 20
drwx------   2 cathy        4096 Jan 21 16:45 .
drwx------  32 cathy        4096 Jan 21 16:39 ..
-rw-------   1 cathy         120 Jan 21 16:43 junk.txt
-rw-------   1 cathy         176 Jan 21 16:45 myfile.txt
-rw-------   1 cathy           0 Jan 21 16:45 session1.txt
-rw-------   1 cathy          75 Jan 21 16:40 stuff.txt
```

In this listing, all files are listed. The permissions of each file are listed along with information about the owner of the file, the size of the file and the date the file was last touched. Note that the listing for your account will probably show different files to those in this list.

```
% mkdir temp
% ls
junk.txt   myfile.txt   session1.txt   stuff.txt   temp
```

Now you have a directory called `temp`.

```
% ls -al
total 24
drwx------   3 cathy        4096 Jan 21 16:46 .
drwx------  32 cathy        4096 Jan 21 16:39 ..
-rw-------   1 cathy         120 Jan 21 16:43 junk.txt
-rw-------   1 cathy         176 Jan 21 16:45 myfile.txt
-rw-------   1 cathy           0 Jan 21 16:45 session1.txt
-rw-------   1 cathy          75 Jan 21 16:40 stuff.txt
drwx------   2 cathy        4096 Jan 21 16:46 temp
```

Notice the permissions on the `temp` directory. The first character is a '`d`', indicating that this is a directory. The next three characters make up the owner's permissions. In this case the owner of `temp` has **read**, **write** and **execute** permissions to the `temp` directory, but permissions to group and other users is denied.

The permissions on the file `junk.txt` indicate that it is a file (as the first character in the permissions list is blank). The owner of the file has read and write permissions to the file.

Change directory to the `temp` directory.

```
% cd temp
```

```
% ls -al
total 8
drwx------   2 cathy        4096 Jan 21 16:46 .
drwx------   3 cathy        4096 Jan 21 16:46 ..
```

Notice that this directory is empty (not surprisingly!). The directories listed here are '**.**' which is the current directory, and '**..**' which is the parent directory.

Change directory back up to the parent directory '**..**'.

```
% cd ..
```

Copy the file `junk.txt` to the `temp` directory.

```
% cp junk.txt temp
```

Change directory to `temp`.

```
% cd temp
```

```
% ls -al
total 12
drwx------   2 cathy        4096 Jan 21 16:48 .
drwx------   3 cathy        4096 Jan 21 16:46 ..
-rw-------   1 cathy         120 Jan 21 16:48 junk.txt
```

The `temp` directory contains a copy of the file `junk.txt`.

Now copy the file `stuff.txt` from the parent directory to the current directory, which is still temp. Remember that '**..**' is the parent directory, and '**.**' is the current directory.

```
% cp ../stuff.txt .
```

```
% ls
junk.txt        stuff.txt
```

The directory `temp` now also contains a copy of the file `stuff.txt`.

Use the move command to rename the file `junk.txt` to `other.txt`.

```
% mv junk.txt other.txt
```

```
% ls
other.txt       stuff.txt
```

The directory `temp` has the file `other.txt` now.

Use the `mv` command to move the file `other.txt` up to the parent directory.

```
% mv other.txt ../other.txt
```

Change directory up to the parent directory.

```
% cd ..
```

```
% ls
junk.txt        other.txt       stuff.txt
myfile.txt      session1.txt    temp
```

The parent directory now has the file `other.txt`.

Remove the file `other.txt`.

```
% rm other.txt
```

```
% ls
junk.txt    myfile.txt    session1.txt    stuff.txt    temp
```

The file `other.txt` has been removed.

Attempt to remove the `temp` directory.

Note that you are not able to remove a directory that is not empty.

```
% rmdir temp
rmdir: temp: Directory not empty
```

Use the recursive option of the remove command to remove the entire contents of the directory `temp`, and then remove the directory. (Be very careful with this option!)

```
% rm -rf temp
```

Change to the `temp` directory ...

```
% cd temp
Can't cd to temp: No such file or directory
```

. . . but as the directory no longer exists, you can't change to it.

```
% ls
junk.txt        myfile.txt      session1.txt    stuff.txt
```

The `temp` directory is no longer listed.

Make a directory called `letters`.

```
% mkdir letters

% ls -al
total 24
drwx------   3 cathy        4096 Jan 21 16:58 .
drwx------  32 cathy        4096 Jan 21 16:39 ..
-rw-------   1 cathy         120 Jan 21 16:43 junk.txt
drwx------   2 cathy        4096 Jan 21 16:58 letters
-rw-------   1 cathy         176 Jan 21 16:45 myfile.txt
-rw-------   1 cathy           0 Jan 21 16:45 session1.txt
-rw-------   1 cathy          75 Jan 21 16:40 stuff.txt
```

There is now an empty directory called `letters`.

Use the `rmdir` commands to remove the empty directory.

```
% rmdir letters

% ls -al
total 20
drwx------   2 cathy        4096 Jan 21 16:58 .
drwx------  32 cathy        4096 Jan 21 16:39 ..
-rw-------   1 cathy         120 Jan 21 16:43 junk.txt
-rw-------   1 cathy         176 Jan 21 16:45 myfile.txt
-rw-------   1 cathy           0 Jan 21 16:45 session1.txt
-rw-------   1 cathy          75 Jan 21 16:40 stuff.txt
```

The `letters` directory is no longer listed.

2.2.10 Creating a Symbolic Link

A symbolic link (or symlink) is a file that points to another file. Symlinks are very useful for linking files that continually move about, or to link library dependencies to their appropriate location. As an example, assume there is a file that continually changes in location. Instead of remembering the new location every single time, it is more appropriate to create a symlink and update this link whenever the file's location changes. This way, users don't have to worry about where the file is, or even be aware of it's location change, as the symlink file will point them to the right place.

A symlink is created using the command below. The filename refers to the file you wish to access, while linkname refers to the name of the symlink file.

```
ln -s filename linkname
```

In the example below, a symlink is created in a public directory to point to a file inside a user's account. By doing this, others can view the file via the symlink without physically accessing the file itself:

```
% ln -s ~/public_html/link2dvd.txt mylink
```

A symlink can be identified by running the `ls` command. Everything after the `->` refers to the file the link is pointing to.

```
% ls -al mylink
     0 lrwxrwxrwx    1 angies   angies         47 Mar 12 14:26 mylink
                      -> /import/kamen/1/angies/public_html/link2dvd.txt
```

A symlink can be deleted like a normal file:

```
%  rm mylink
```

2.2.11 Printing

The files that you accumulate and use will vary in their content. Some files will be binary, some text, some postscript, and so on and so forth. Not all file types are printable. All of the School's printers are postscript laser printers. If you send a file to the printer which is not a postscript file, and appears to be a text file, it will be converted into postscript and then printed (more about this in the next section).

Remember that there is a *quota* on the number of pages you are allowed to print. Your print quota is discussed in section 1.9.5 on page 31.

2.2.12 Finding the Nearest Printer

Prior to sending your file to the printer, the `whichprinter` command can be used to determine the name and location of printer that is closest to you. As an example, running this command from the Drum computer lab yields the following result:

```
whichprinter
Printers at drum are:
itch - Elec Eng undercroft printer room
shooter - Elec Eng undercroft printer room
```

2.2.13 Sending a Job to the Printer

To print files on any of the School's printers you use the `lpr` command. This used to be short for *Line PRinter*. The command `pr` (for *PRint*) does something completely different. Check the manpage for details.

A list of available printers can be found in table 1.2 on page 8. The general form of the `lpr` command is:

```
lpr -Pprintername filename
```

For example, to print the postscript file `poem.ps` on the printer `stu` you would type

```
% lpr -Pstu poem.ps
```

Similarly, you could print the text file `poem.txt` by typing

```
% lpr -Pstu poem.txt
```

This will print the file containing a title (in this case `poem.txt`), border, and a heading consisting of your login name.

To print the same file in single sided mode without your login name, append `.sgl` to the end of the printer name:

```
% lpr -Pstu.sgl poem.txt
```

Similarly, appending `.dup` will print out the file using both sides of the paper without your login name being displayed:

```
% lpr -Pstu.dup poem.txt
```

If you want some other layout, you will have to convert the text file to postscript yourself. You can do this using the program `aps`. The general form of this command is:

```
aps [flags] textfile
```

It will print out the text file in postscript format. Given no flags, the postscript will be exactly as it would if you send the text straight to the printer, but in portrait. `aps` puts its output on the standard output, so you will probably want to redirect it to a file, or straight to the `lpr` command.

Here are some examples of how you could print the file `poem.txt` on the student laser printer:

```
% aps poem.txt | lpr -Pstu

% aps poem.txt > poem.ps; lpr -Pstu poem.ps
```

To print a text file with no border or header in portrait form you can use:

```
aps -~B -P -~H file.txt | lpr -Pstu
```

Here, the '~' option is used to turn off the border (B) and header (H). For more information on file printing and viewing, refer to section 2.3.6 on page 58.

2.2.14 Checking the Printer Queue

When you send your job to the printer it goes to a *spool* area where it is put onto the printer queue. To check where your job is in the printer queue, you can use the `lpq` command. The general form of the command is:

```
lpq -Pprintername
```

For example, to check the print jobs on the printer `soup`:

```
% lpq -Psoup
```

To check the only print jobs on the printer soup owned by angies:

```
% lpq -Psoup angies
```

Here is an example:

```
% lpq -Psoup
Queue soup on ives.orchestra.cse.unsw.EDU.AU
  Ready since Jun 29 11:18:44.
Remote printer soup.ps on score.orchestra.cse.unsw.EDU.AU
  Printing (started at Jun 29 15:50:52, attempt 1).
  Rank Owner    Pr  Job Host        Files                   Form   Size Time
active s2191640 X   875 fife03      (stdin)                    - 302.0K 15:47
   1st s2191640 X   876 fife03      (stdin)                    - 146.2K 15:47
   2nd s2191640 X   877 fife03      (stdin)                    - 454.5K 15:47
   3rd s2191640 X   878 fife03      (stdin)                    -   1.1M 15:47
   4th s2191640 X   879 fife03      (stdin)                    - 211.8K 15:47
   5th s2191640 X   881 fife03      wk08.ps                    - 127.2K 15:49
   6th s2172328 X   846 fife13      vis1.ps                    - 785.0K 15:49
```

As you can see, the output is arranged in rows, with one row per print job.

- *Rank* is the job's current rank in the queue. An *active* job is the one which is currently printing.
- *Owner* is the username of the person who sent the file to the printer.
- *Pr* is the job priority, generally not used.
- *Job* is the job number — you will need the job number if you want to remove your job from the queue.
- *Host* is the computer from which the print job was sent.
- *Files* is the names of the files which were sent to the printer. Jobs which have been piped to `lpr` will show as `(stdin)`.
- *Size* is the total size of the print job. Note that this is the number of bytes in the PostScript file, not the number of pages.
- *Time* is the time the job was sent.

2.2.15 Removing a Job from the Printer Queue

To remove any of your unwanted print jobs from the printer queue, you use the `lprm` command. As an example, to remove all your print jobs from the `stu` printer:

```
% lprm -Pstu -a
```

To remove the currently active job (assuming you own it):

```
% lprm -Pstu
```

Note that quite often it is too late to remove an *active* job from the queue.

To remove a specific job you need the job number (get this from `lpq`). To remove job 1234:

```
% lprm -Pstu 1234
```

2.2.16 Permissions

Access to your files, for yourself and for other users, is controlled by setting *file access permissions* on files and directories. The UNIX file system allows file access permissions to be set for the following categories of users:

- The owner

- All users belonging to the same group as the file's group owner. (All users belong to one or more groups. The group is important for security and accounting reasons. To find out which groups you belong to try the command `groups`.)
- All other users

These categories of access permissions can be set to either allow or deny *read*, *write*, and *execute* access to your files.

By default, when you create any files, you, as the owner, are granted the relevant read, write and execute permissions. Other users are given no access permissions to your files.

You can have System Support set up a group directory for group assignment purposes if necessary. Email `ss` to have this arranged.

You will recall from the previous section that the `-l` option of the `ls` command displays the access permissions to a file for all three categories listed above. So, what does it all mean? This is best answered with an example:

```
% ls -al
total 24
drwx------   3 cathy        4096 Jan 21 16:58 .
drwx------  32 cathy        4096 Jan 21 16:39 ..
-rw-------   1 cathy         120 Jan 21 16:43 junk.txt
drwx------   2 cathy        4096 Jan 21 16:58 letters
-rw-------   1 cathy         176 Jan 21 16:45 myfile.txt
-rw-------   1 cathy           0 Jan 21 16:45 session1.txt
-rw-------   1 cathy          75 Jan 21 16:40 stuff.txt
```

Permissions are always listed in the order read, write, execute.

From the above example, you can see that `letters` is a directory, and that the owner has read, write and execute permissions. The owner of the files in this directory, has read and write permissions to the files but group and other permissions are denied.

2.2.17 Changing Permissions

The `chmod` command is used to change the access permissions on a file. Only the owner of a file can change its access permissions (with the exception of the super user). The general format of the command is:

```
chmod mode filename
```

where *mode* has the format:

```
[who] op permission
        or
[0-7][0-7][0-7]
```

who This is a combination of letters **u** for *user's* permission, **g** for *group's* permission and **o** for *other's* permission. The default is **ugo**, this means everyone!

op This can be a + to add permission or − to take away permission.

permission This is any combination of the letters **r** for *read*, **w** for *write* and **x** for *execute*. The permission can also be indicated using a number to indicate the bits to be set.

Permissions may also be specified using the summation of octal (base 8) numbers:

Permissions	Octal
Read permission	4
Write permission	2
Execute permission	1
Permission denied	0

These numbers are added to indicate the permission for each group. Some examples:

Permissions Granted	Number To Use
read, write, execute	7
read and write	6
read only	4

Some examples of manipulating permissions follow:

A listing that shows the current permissions.

```
% ls -al
total 24
drwx------   2 cathy        4096 Jan 21 17:00 .
drwx------  32 cathy        4096 Jan 21 16:39 ..
-rw-------   1 cathy         120 Jan 21 16:43 junk.txt
-rw-------   1 cathy         176 Jan 21 16:45 myfile.txt
-rw-------   1 cathy        2989 Jan 21 16:58 session1.txt
-rw-------   1 cathy          75 Jan 21 16:40 stuff.txt
-rw-------   1 cathy           0 Jan 21 17:00 typescript
```

Change the permissions on the file `junk.txt` to grant read and write permissions to group and other users. It is usually a bad idea to let other users write to your files or directories.

```
% chmod 666 junk.txt
% ls -al
total 24
drwx------   2 cathy        4096 Jan 21 17:00 .
drwx------  32 cathy        4096 Jan 21 16:39 ..
-rw-rw-rw-   1 cathy         120 Jan 21 16:43 junk.txt
```

```
-rw-------   1 cathy         176 Jan 21 16:45 myfile.txt
-rw-------   1 cathy        2989 Jan 21 16:58 session1.txt
-rw-------   1 cathy          75 Jan 21 16:40 stuff.txt
-rw-------   1 cathy           0 Jan 21 17:00 typescript
```

Remove the others permissions from the file `junk.txt`.

```
% chmod o-rw junk.txt
% ls -al
total 24
drwx------   2 cathy        4096 Jan 21 17:00 .
drwx------  32 cathy        4096 Jan 21 16:39 ..
-rw-rw----   1 cathy         120 Jan 21 16:43 junk.txt
-rw-------   1 cathy         176 Jan 21 16:45 myfile.txt
-rw-------   1 cathy        2989 Jan 21 16:58 session1.txt
-rw-------   1 cathy          75 Jan 21 16:40 stuff.txt
-rw-------   1 cathy           0 Jan 21 17:00 typescript
```

Now remove the group permissions from the file `junk.txt`.

```
% chmod g-rw junk.txt
% ls -al
total 24
drwx------   2 cathy        4096 Jan 21 17:00 .
drwx------  32 cathy        4096 Jan 21 16:39 ..
-rw-------   1 cathy         120 Jan 21 16:43 junk.txt
-rw-------   1 cathy         176 Jan 21 16:45 myfile.txt
-rw-------   1 cathy        2989 Jan 21 16:58 session1.txt
-rw-------   1 cathy          75 Jan 21 16:40 stuff.txt
-rw-------   1 cathy           0 Jan 21 17:00 typescript
```

Grant group read and write permission to the file `myfile.txt`.

```
% chmod 660 myfile.txt
% ls -al
total 24
drwx------   2 cathy        4096 Jan 21 17:00 .
drwx------  32 cathy        4096 Jan 21 16:39 ..
-rw-------   1 cathy         120 Jan 21 16:43 junk.txt
-rw-rw----   1 cathy         176 Jan 21 16:45 myfile.txt
-rw-------   1 cathy        2989 Jan 21 16:58 session1.txt
-rw-------   1 cathy          75 Jan 21 16:40 stuff.txt
-rw-------   1 cathy           0 Jan 21 17:00 typescript
```

It is a matter of personal preference as to which of these two formats you decide to use. Whichever format you choose to use, **make sure that you never make your home directory unreadable by you!** This will result in you not being able to login again! If you do happen to have a problem, go to the Help Desk to have the problem resolved.

2.2.18 Finding and Killing Your Processes

When you run a program, a copy of the program is loaded into the computer's memory and is run from there. A running program is called a *process*. It is easy to run many processes at once, and to lose track of some of them, especially when you are writing your own programs and when these programs spawn further processes. This can be troublesome as it can consume excessive amounts of computer resources and cause you to breach the terms and conditions of your account usage. See *Why is my Account Access Denied?* (section 1.6.2, page 23) and *The Yellow Form* (section 1.6.3, page 24).

You can find out what processes you have running using the `ps` command, and you can stop runaway processes using the `kill` command.

To use the `ps` command in a fairly simple way, type

```
% ps -u <username>
```

This will produce an output something like:

```
ives:/home/www/lib/html> ps -u tchambers
   PID TTY      S       TIME CMD
  1853 ??       S     0:03.61 /usr/local/bin/xdvi -paper a4r -geometry 1050x
  4803 ??       I     0:00.16 (dns helper)
  9118 ??       S     0:11.43 /usr/local/bin/.netscape4 -no-about-splash
 11742 ??       I     0:00.24 /bin/ksh /import/bizet/3/tchambers/.xsession
 12307 ??       S     0:00.10 xclock -digital
 17697 ??       S     0:00.91 /usr/local/X11/bin/twm
 30481 ??       I     0:00.06 /usr/local/bin/checkmail
 32565 ??       I     0:00.65 gs -sDEVICE=x11 -dNOPAUSE -q -
   200 ttypa    I +   0:03.24 pine -i
 25844 ttypc    S     0:08.04 emacs primer.tex
 32093 ttypc    I +   0:00.48 ae
 16503 ttyq2    S     0:00.44 ae
```

The number in the 'PID' column is the *Process ID Number* which is used to identify the process and is what you need to know to be able to kill a process. The 'TTY' column shows which terminal (or x-window) the process output is displayed in. The 'S' column is the process status. The 'TIME' column shows how much cpu time the process has consumed. This is a good indicator of how resource hungry the process is. The 'CMD' column lists the command which launched the process, or the name of the program.

The `ps` command has many more capabilities. You should read the manual (type `man ps`).

Note: `ps` *can behave differently depending on the architecture you are using.*

To kill a process, you type

```
% kill <PID>
```

The `kill` command won't tell you whether or not it was successful — you should run the `ps` command again to verify that the process you wanted to kill has indeed been killed. If you have a process which won't die, you can use a more insistent form of the `kill` command:

```
% kill -9 <PID>
```

You should avoid using this version if you can because it does not give the process a chance to 'tidy up', and can therefore lead to problems.

2.3 Useful Commands

passwd Change your current password. You should change your password when ever you feel that it may have been compromised. It is a good idea to change it every few months anyway.

usage: `passwd`

cat Display the contents of the files listed as arguments. The named files are concatenated to the terminal in the order in which they are listed as arguments to the command.

usage: `cat file1 [file2...]`

more This command allows you to browse the contents of a file a page at a time. To move forward through the file a page at a time press the **f** key. To move back a page press the **b** key. When you have had enough, press the **q** key to quit.

usage: `more filename`

less This command is an extension of the `more` command, as it allows foward and backward traversal using arrow keys.

usage: `less filename`

grep This command searches for all occurrences of the pattern in the named files, or standard input.

usage: `grep pattern file1 [file2...]`

This is an incredibly powerful command, especially when combined with regular expressions. It is often used in a 'pipe' to sort out the lines of output from another command which contain (or do not contain) a certain word or pattern. Refer to section 3.8 on page 65 for more information on the pipe character (|).

As an example, to search for the word hello in a text file, you could use the 'pipe' and `grep` command as follows:

```
more file.txt | grep hello
```

diff This command finds all of the differences between the two files `old-filename` and `new-filename`. The output of this command is a list of `ed` commands that could be executed on `old-filename` to produce `new-filename`. (This command can be very useful if you ever get into a situation where you have confused yourself with several versions of the one file!)

usage: `diff old-filename new-filename`

date This command displays the current system date and time.

usage: `date`

2.3.1 Checking your Account Details

acc This command checks account details from account status to print and IP quota details.

usage: `acc`

rquota This command summarises disk and file quota allocation and how much is being used.

usage: `rquota`

disk_guess This command will give you a detailed disk usage summary, listing your 6 biggest files and 6 biggest directories as well as advice on how to remove unnecessary files to save disk space.

usage: `disk_guess`

For more information on Disk Quotas, refer to section 1.9.4 on page 30.

2.3.2 Print Commands

mlpr This command manipulates a .ps file to print several logical pages onto a single physical page. The number of logical pages printed per physical page must be an order of 2.

usage: `mlpr -n -Pprintername filename.ps`

where n is the number of logical pages (eg, 2, 4, 8)

2.3.3 Floppy Disk Commands

mdir This command will display your disk contents.

usage: `mdir`

mcopy You can use this command to copy files between your floppy disk and your home directory.

usage: `mcopy a:/filename .` This will copy `filename` from your floppy disk to your current working directory, represented by '.'

usage: `mcopy filename a:` This will copy `filename` from your current working directory to your floppy disk a:

mdel Is used to delete files from your floppy disk.

usage: `mdel a:/filename`

mformat This command will format your floppy disk.

usage: `mformat a:`

For more information on floppy disk commands, refer to the man page for `mtools`.

2.3.4 Compressing Files and Directories

gzip This command will compress a single file.

usage: `gzip file`

This will produce a compressed version of `file` called file.gz, taking up much less disk space.

gunzip This command will uncompress a compressed file.

usage: `gunzip file.gz`

This will uncompress your `file.gz` to give you your original file.

tar This command will archive a whole directory, and its subdirectories and all the files in those directories, into a single file.

usage: `tar cvf - dir > archive_name.tar`

Where `dir` is the directory you want to collect and archive_name.tar is the name you want to give to the resulting compressed directory.

That file can then be compressed with `gzip`, creating `archive_name.tar.gz`.

untar This command will uncompress a compressed directory to give you your original directory structure.

usage: `tar -xvf archive_name.tar`

2.3.5 Viewing and Printing Microsoft Office Documents

Microsoft Office documents can be viewed, edited, and printed using `openoffice`. Printing files from `openoffice` is a two step process which involves converting the file to PostScript (`.ps` format) and then printing the newly created `ps` file.

To convert a file to `ps`:

1. Open the file in either `openoffice`.
2. Click on `File -> Print`.
3. A Print dialog box will appear. Select the `Print to File` option and click `OK`.
4. A second Save As dialog box will appear. Enter a filename with a `.ps` extension and click `Save`

Information on printing your newly created `ps` file can be found in the next section. For more detailed instructions on the steps outlined above, consult the Help Desk web site at `http://www.cse.unsw.edu.au/~helpdesk`.

2.3.6 Viewing and Printing PDF and PS files

To view PostScript (`.ps`) files, you need to use a program called `gv`. PDF (`.pdf`) files can be viewed using `acroread` or `xpdf`.

By default, `xpdf` will save the `.pdf` file to a postscript format, which you can then print. When you choose the print option, you will see a destination for the file to be saved rather than a print command. You can subvert this by piping the output of the save operation to a printer. To do this, you will need to replace the existing path in the print box with the print command below.

lpr -Pprintername This command will print a `.ps` file from `gv` and a `.pdf` file from `acroread`. From `gv`, this will be the complete print command. However, from `acroread`, you will need to replace the existing text in the box with this command.

|lpr -Pprintername This command will print a `.pdf` file from `xpdf`.

2.3.7 Basic LaTeXCommands

LaTeXis a language for typesetting documents. It is widely used in `unix`, being portable and powerful, particularly for scientific and mathematical papers. For an introduction to LaTeX, type

```
%info latex
```

Some useful LaTeX-related commands are:

latex This command processes the formatted source file `file.tex` producing (at least) 3 output files: `file.dvi`, `file.log` and `file.aux`.

usage: `latex file` where `file` is a file of extension .tex

xdvi This command launches `xdvi`, a viewer which views `.dvi` files so you can see what will be printed.

usage: `xdvi file` where `file` is a file of extension .dvi

dvips This command will convert `.dvi` files to postscript files.

usage: `dvips file` where `file` is a file of extension .dvi

You can print the file by typing

`dvips file |lpr -Pprintername` where `file` is a file of extension .dvi

pdflatex This command allows you to create PDF files from LaTeX files. It behaves almost identically to `latex`.

2.4 The Online Manual

The first place to start start searching for information is the online manual. For almost all of the commands on the system there is a corresponding online manual page which contains information about the command such as its syntax, what options are available for its use (these are like fashion accessories, most commands can be dressed up or dressed down), known bugs, and if you are lucky some examples of how to use the command.

Access to the online manual can be gained using the `man` command. The general form of the command is:

```
% man command-name
```

As an example the command:

```
% man man
```

would display the manual entry for the `man` command. The manual pages are displayed using the command `less` as described above.

You will notice that to use the online manual in this way, you have to know the name of the command that you are interested in. If you wish to look for manual entries on a particular topic, try the commands:

`% man -k` *keyword*

`% apropos` *keyword*

Some topics have more than one manual entry, for example:

```
% man -k uuencode
uuencode (5)            - format of an encoded uuencode file
uuencode, uudecode (1c) - encode/decode a binary file to transmit by mail
```

The number in parentheses is the section of the manual in which the entry is located. `uuencode (1c)` contains information about the program `uuencode` this manual entry can be read by typing `man 1c uuencode`. Whereas `man 5 uuencode` will get you information about the *uuencode* file format.

2.4.1 tkman

Another useful tool is the manual browser called `tkman`. You can start up this manual browser by running the command at the shell prompt:

```
% tkman &
```

You enter the name of the manual page you wish to read in the box at the top of the window. `tkman` enables you to select which manual page you wish to read if there are several with that name. You can also search for text within a manual page. You can also do apropos searches. Have a look at the manual page for `tkman` for more details.

You will probably find it tough going reading the online manual. It takes time to get used to the language and format in which the information is presented. Try to persevere, it will become easier as your knowledge and understanding develop.

2.5 Info

Another primary source of information are the `info` documents, viewed either with the `info` command, or via from info-mode within `emacs`. All GNU programs and much of the rest of our system is well documented with info, easily navigable but as much detail as you might want.

`info` uses similar commands to `emacs`. Typing '`h`' from within `info` invokes comprehensive documentation about `info`, including a tutorial on using it.

Typing '`q`' gets you out of `info`.

3 Introduction to Shells

3.1 What is a Shell?

When you type commands in an `xterm`, you are actually interacting with a piece of software called a *shell*. The shell interprets your command line and causes your program (or programs) to be executed in an appropriate way. The shell is also a fully-featured programming language in its own right.

There are a number of common Unix shells:

Bourne Shell (sh) The traditional Unix shell (written by Steve Bourne) and as such is available on all Unix systems. Somewhat 'unfriendly' for interactive use.

C Shell (csh) Has C-like syntax and some useful interactive features (job control and command aliases).

TC Shell (tcsh) Adds Tenex-like command-line editing to the C-shell (see *'The T in* `tcsh`*'* in the tcsh manpage). Not related to Top Cat.

Korn Shell (ksh) The Korn shell (written by David Korn) adds the interactive features of tcsh to the Bourne shell.

Bash (bash) The Bourne Again SHell, generally similar to the Korn shell.

Z Shell (zsh) A further enhancement of the Bourne shell: it corrects your spelling, it makes your coffee, ...

For further information refer to the various `man` pages for each shell.

3.2 Initialisation Files

A number of files are used to configure and customise aspects of your login session and the behaviour of your programs (see the manual pages for the relevant programs for full details). The primary initialisation files that you will use are:

3.2.1 .xsession

If the file `.xsession` file exists in your home directory, it is used when you log into a computer console to:

- Start any standard X-based applications, such as `xclock`, `xterm` or `xscreensaver`.
- Invoke the user's preferred window manager (see *The Window Manager*, section 4.6, page 71).

3.2.2 .profile

If the file `.profile` file exists in your home directory, it is used when you log into a computer to:

- Define any required environment variables such as `$PATH` (see *Shell and Environment Variables*, section 3.5, page 64).
- Start any standard non-X-based applications.

This file is also used whenever you log into a computer via `ssh` (in addition to logging into the console).

Some shells use different filenames. For example, C-shell uses the file `.login`.

3.2.3 .shellrc

Some shells also use an initialisation file for each instance of a shell (eg, `xterm`). Typically these define command aliases and shell functions which cannot be passed via the environment. `bash`, for example, uses the file `.bashrc`. Check the manpage for your shell for full details.

3.3 Changing Your Default Shell

The default shell for CSE users is `bash`. To change to another shell:

1. Type `chsh` at the command prompt
2. Enter your password
3. Enter the name of the shell you wish to use

This new login shell will be available for use the next time you log in. All of the shells mentioned in *What is a Shell?* (section 3.1, page 61) are available.

3.4 What is a PATH?

The PATH variable controls where the shell searches for commands when you type them at the prompt (see *Shell and Environment Variables*, section 3.5, page 64). When you type a non-builtin command to the shell, the shell searches for a program to execute. The programs are simply executable files somewhere in the file system; they are executable either because they are compiled programs or because they are scripts that may be executed. They also need to have their 'execute permission' enabled (see *Changing Permissions* in section 2.2.17, page 51).

Since we don't want to go through all the files on the file system, we have a path of directories where the program may be stored. This path is given to the shell as a colon-separated list of directories stored in the environment variable PATH. This means that any executables or other files within that directory can be accessed by typing only the filename without having to specify the entire directory.

For example, if the PATH variable has been set to `/usr/local/java`, then in order to access the java compiler, the user would only need to type:

```
% javac <filename>
```

However, if the PATH variable had NOT been set or did not include `/usr/local/java`, the user would need to specify the location by typing:

```
% /usr/local/java/javac <filename>
```

The PATH variable for a user is usually set in the user's `.profile` file. You can check your PATH at any time by typing:

```
% echo $PATH
```

It will look something like the following colon-separated list of directories.

```
/import/bizet/B/angies/Bin:/usr/local/bin:/usr/bin:/usr/etc:
/usr/sbin:/usr/local/bin:/usr/local/X11/bin:/usr/local/etc:
/usr/openwin/bin:/usr/openwin/demo:/sbin:/etc:/usr/local/
X11R6/bin:/usr/local/share/bin:/bin
```

Caution: It is generally a **bad** idea to include '.' (the current directory) in your PATH. This makes you vulnerable to Trojan horses and other mistakes.

The `which` command is a command locator. It will find which version of the command in your PATH will be run. Some shells have a `where` command which will find all the versions of the command in your PATH. Within `bash`, the command '`type -a`' does the same thing. You can then choose to modify your PATH or just type in the full path of the version of command you want. See the manpage for your shell for advice on changing your PATH by typing:

```
% man <shell>
```

3.5 Shell and Environment Variables

All Unix shells come complete with variables. In the Bourne shell family of shells (that is, all other than csh and tcsh), these are set and used such as:

```
% STUFF="winds light to variable"
% echo The forecast is: $STUFF.
The forecast is: winds light to variable.
```

Notice that shell variables are traditionally (though not necessarily) in UPPER-CASE, and are dereferenced by a prefix '`$`'.

Variables can also be inherited from the *Environment*. Environment Variables are a useful way of passing information on to child processes. Variables can be *exported* into the environment with:

```
% FRED=frog
% export FRED
```

or more simply with:

```
% export FRED=frog
```

It is often useful to change the environment of just one process with:

```
% FRED=frog programname arg1 arg2
```

There are a number of 'standard' environment variables, such as:

`$PATH`	Where to find programs (see *What is a PATH*, section 3.4, page 63).
`$PRINTER`	The name of your default printer. If this is set, then you can omit the '`-Pprintername`' from commands such as `lpr` and `dvips`.
`$ARCH`	The current hardware/software platform. Typically such values as `pc.i86.linux` or `sun.sparc.solaris`.

and many more.

3.6 *stdin, stdout* and *stderr*

Most UNIX programs read their input from a 'file descriptor' known as 'standard input' (*stdin*) and send their output to 'standard output' (*stdout*). Any error messages are sent to 'standard error' (*stderr*). Usually *stdin* is the keyboard, and *stdout* and *stderr* are the screen. Sometimes it is useful to redirect these so that a program can take its input from somewhere else, or send its output elsewhere. You have already seen this used when we looked at printing files (see section 2.2.13, page 48). There are more details about this in the manual pages for `bash`, but some of the possibilities are explained in the following sections.

3.7 Exit Status

All Unix programs exit with an *exit status* which generally denotes success or failure of the program. This is useful, for example, with the '`&&`' and '`||`' constructs in the *Unix Shell Metacharacters* section following. See also *Shell Scripts* (section 3.9, page 67).

The exit status is a numeric value: `0` denoting success and any positive integer denoting failure. Although somewhat counter-intuitive, the simple rationale for this is that there is usually only one way of succeeding, but a number of ways of failing.

You can visually check the exit status of a program by typing '`echo $?`' or '`echo $status`' (depending on your shell) immediately after running the program.

3.8 UNIX Shell Metacharacters

These are special characters that are recognised by the shell. Here are some that you will use often:

`*` This is a 'wildcard', it matches any string of zero or more characters, except a leading '.' An example of its use:

```
% ls *.mod
```

This will list all the files in the current directory that end with a `.mod`

```
% ls ca*
```

This will list all the files that commence with the letters 'ca' and have zero or more characters afterwards.

`?` This will match any single character An example of its use:

```
% ls tut?.m
```

This will find files such as `tuta.m`, `tut1.m`, etc.

`[`*ccc*`]` This will match any single character from the string *ccc*. Ranges such as 0–9, a–z, and A–Z are also valid. An example of its use:

```
% ls tut[0-9].m
```

This will find files such as `tut1.m`, `tut2.m`, etc

```
% ls *.[ch]
```

This will list all files in the current directory that end in either `.c` or `.h`.

`>` Redirect standard output (the terminal) to a file. An example of its use:

```
% echo "Hello world" > hello.txt
% cat hello.txt
Hello world
```

`>>` Appends standard output to a file. An example of its use:

```
% echo "Hello again" >> hello.txt
% cat hello.txt
Hello world
Hello again
```

`<` Take standard input from a file An example of its use:

```
% mail bruce < myfile.mod
```

This will mail the file `myfile.mod` to the user `bruce`. Electronic mail is discussed in more detail in section 5.1 on page 76.

`|` This is the pipe character. **P1** | **P2** connects standard output of **P1** to standard input of **P2**. An example of its use:

```
% cat poem.txt | lpr -Psoup
```

This will print the contents of the file `poem.txt` to the printer called `soup`.

```
% cat duckpond | tr ' ' '\n' | grep -i frog | wc -l
```

This will count the number of frogs in the `duckpond` file. Check the manual pages for `cat`, `tr`, `grep` and `wc` yourself.

`;` **P1 ; P2** executes **P1** and then **P2**.

```
% cd Temp ; lpr -Psoup file.txt
```

This will change your working directory to `Temp` and print the file `file.txt` in that directory to the printer called soup.

`&` This starts a process in the background. **P1 &** will cause **P1** to start running, and return you to the shell you were typing at, whilst **P1** keeps going. This is useful for programs which start up their own xwindow.

`&&` In **P1 && P2**, if **P1** *succeeds*, then **P2** is executed. This is like a logical *and*: both **P1** and **P2** have to succeed for the compound statement to succeed. See *Exit Status* (section 3.7, page 65).

```
% latex file && dvips file
```

If the `latex` command succeeds, then `dvips` will be run.

`||` In **P1** || **P2**, if **P1** *fails*, then **P2** is executed. This is like a logical *or*: only one of **P1** and **P2** have to succeed for the compound statement to succeed. See *Exit Status* (section 3.7, page 65).

```
% ./assign1 || echo "bugger"
```

The program `assign1` (in the current directory) is run. If it fails, then a helpful error message is printed.

Your initial use of shell metacharacters will be for simple interactive use, as in the above examples. Remember, however, that Unix shells are fully-featured programming languages and that all these constructs can be used to powerful effect in shell programs or scripts. See the following section on *Shell Scripts*.

3.9 Shell Scripts

As noted earlier, Unix shells are all fully-featured programming languages as well as simple ways of invoking programs. A file containing a number of shell commands is called a *shell script*.

Simple shell scripts are simply a sequence of simple commands (such as are many `.profile` files). However, a full set of iterative and conditional constructs are also available. The manual pages for each shell will give a complete description of the features available; though they do tend to be fairly cryptic and often a book from the book shop is a more useful approach to learning shell scripting.

The very big advantage of shell scripts is that they provide a simple way of automating a number of otherwise manual tasks. If you need to type a sequence of commands in more than twice, you probably want to write shell script. Remember, we programmers are all tool makers; we like making things easier for ourselves.

For a shell script to be usable as a program:

- It must live in a directory in your `$PATH` (see *What is a PATH*, section 3.4, page 63).
- It must have an appropriate execute permission set (see *Permissions*, section 2.2.16, page 50).
- The first line must consist of the string: '`#!/bin/sh`' (or pathname of your shell).
- It must not be called `test` (strange things will break).

Finally, it is traditional, though not at all compulsory, for shell scripts to be written in Bourne (or compatible) shell; this makes portability easier.

But keep an open mind. For a number of problems, other tools such as `perl`, `tcl/tk` or `awk` might be a better solution. It is worth being familiar with a number of tools so that you can use the right tool for each task.

4 The X Window System

4.1 Some Basics

The Display

This is the name given to the terminal (screen and keyboard) that you sit in front of.

The Server

This is the piece of software that actually controls the display. It is only concerned with drawing text and graphics onto the screen.

Clients

These are a kind of software known as *X applications*. Clients make a connection to the server and send it all their drawing requests. The server then translates these into the actual graphics on the screen. When you, as a user, click a mouse button, or type in a window, the server sends back events to the client which owns that particular window. Clients can connect to a server on the same or a remote machine.

X session

This refers to the time period from when a user first logs onto an X display to the time they log off.

xterm

An *xterm* is a window on your display in which you can type UNIX commands and run programs.

4.2 Getting Started

The login box on the terminal is displayed by a program called `xdm`. When it has verified that you are a valid user of the system from your login and password, it will then set up your environment for the session. It checks to see if there is a `.xsession` file in your home directory; if there is not, `xdm` sets up the default environment. If it

does find a personal `.xsession` file `xdm` executes the commands in this file instead of using the defaults.

An `.xsession` file is a shell script; you can have a look at an example `.xsession` file in:[1]

```
/usr/local/X11R6/lib/X11/xdm/Session-Script
```

or

```
/usr/local/X11/lib/X11/xdm/Xsession
```

4.3 Logging in without .xsession

When you first start to fiddle with your own environment, much can go wrong! If you do something silly in your `.xsession` file, your session may not start up successfully.

To get around this problem, login as normal BUT instead of pressing the RETURN key after your password, press the F1 key. This will cause `xdm` to completely ignore your `.xsession` file. This will enable you to log in successfully and correct the problems in your `.xsession` file. Your `.xsession` file is just a shell script, so once you have fixed the problems you can execute the file like any other.

This is very useful for testing the changes you make as you go, and any error messages will be displayed on the terminal.

As well as the F1 key, there are several other options you can use:

F1 Opens an xterm window

F2 Opens the `Fluxbox` window manager with standard settings

F3 Opens the `twm` window manager with standard settings

F4 Opens the `KDE` window manager with standard settings

4.4 Setting the X Display Resolution

At the login screen of some machines you will see a small text box in the top right hand corner which allows you to adjust the resolution of your X display. Figure 4.1 is an example of what the text box looks like. Some resolution selection boxes offer higher bit depth than the one shown.

Applications use several colours when they are displayed on your screen. If you select 8-bit colour you will have 256 colours available. Firefox, for example, can use

[1] The name of the file and the location of the file may vary depending on the system you are using.

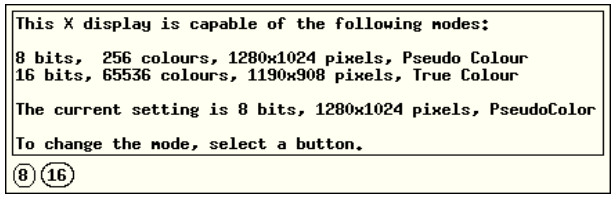

Figure 4.1: Resolution selection box.

many colours and you may find that running a text editor like nedit at the same time will force it to have a white background as opposed to grey. This is because Firefox has grabbed the colours it needs to display its window first. Further applications then use what they can. If you use the 16-bit colour option (or higher) before you log in to a machine, this will not occur.

It is strongly recommended that you select 16-bit colour (or higher) if you are using the WindowMaker or KDE window managers. Being more detailed than the default fluxbox window manager they tend to use many colours. For more information on window managers see section 4.6 on page 71.

4.5 Mouse Buttons

Inside an xterm

Inside an `xterm`, if Button 1 of the mouse is held down and dragged, the text is highlighted. Pressing Button 2 while the text is still highlighted causes the text to be pasted into the window. This is a handy way of transferring text from one `xterm` to another.

Inside an `xterm`, each mouse button is also associated with a menu. If you press Ctrl–Button 1 you will get the *Main Options* menu, which can be useful if the program you are running in that `xterm` runs into trouble or you want to get rid of that `xterm`. Ctrl–Button 2 brings up the *VT Options* menu, which controls various ways the `xterm` behaves such as line wrapping and scroll bars. The third menu, appearing when you press Ctrl–Button 3, is the *VT Fonts* menu, which allows you to change the font size of that `xterm`.

These menus are available independent of the window manager you are using.

Outside an xterm

Outside an `xterm`, holding down the mouse buttons will give you different, and not to mention very useful, menus. These menus vary slightly and are accessed differently depending on the window manager you are running.

4.6 The Window Manager

The *window manager* is what controls the 'look and feel' of your screen. It adds the titlebars and borders to all of your windows. It is also responsible for any menus that appear when you click on the background. Different window managers decorate their windows with different icons used to manipulate the windows.

Some window managers support 'virtual desktops', which means you can have more than one screen full of windows. This often makes it easier to organise the way you work.

You can also customise the way your window manager looks. This can range from simply modifying the colour of the titlebars and the background image, to changing the borders and adding extra buttons to the titlebar.

The following sections describe four of the window managers currently available on the School's system.

4.6.1 Fluxbox

When you log in at a terminal, the default window manager *Fluxbox* is already running in order to handle the windows on the display.

Fluxbox is a window manager that is based on the code from BlackBox. It is pretty lightweight and tries to leave you with as much screen space to yourself as possible. It provides multiple desktops, *tabs* (a useful feature if you get to know it) and many other bits and pieces.

Fluxbox allows you to switch between workspaces, and move windows between workspaces.

The following is a brief list of some useful shortcuts you can use in Fluxbox:

Alt–F*num*	Switch to workspace number *num*.
Alt–Tab	Cycle through windows in current workspace.
Alt–Shift–Tab	Cycle backwards.
Alt–leftmouse–drag	Move the window around by dragging any part of the window.
Alt–rightmouse–drag	Resize the window by dragging any part of the window.
middlemouse–drag	Drag a tab onto another window to put it into the other window's tabs. To pull it out again, simply drag the tab (using middle button) away from the window.
Ctrl–Tab*	Cycle through tabs.

Ctrl–Shift–Tab* Cycle backwards through tabs.

*These are CSE-specific keybindings. Please see `http://www.cse.unsw.edu.au/faq/questions/other-fluxbox.html/` for information on how to modify these yourself.

There are many more options associated with the use of Fluxbox. For more information on the Fluxbox window manager and its features see `http://www.cse.unsw.edu.au/faq/questions/other-fluxbox.html`.

4.6.2 Twm

This is the no-frills window manager, it is fast and stable.

With twm, you have the option to configure or customise the menus in your window manager. Copy the file system.twmrc in /usr/local/X11/lib/X11/twm/ into your home directory as .twmrc and edit it.

You can do this by typing:

```
% cp /usr/local/X11/lib/X11/twm/system.twmrc  /.twmrc
```

And then you can edit your `.twmrc` file from your home directory.

4.6.3 WindowMaker

WindowMaker is a window manager designed to emulate the 'look and feel' of the NeXTSTEP(tm) GUI. It is quite elegant, and comes with a graphical configuration tool to help you customise it. There are also 'themes' available which you can use to quickly change WindowMaker's appearance. It is relatively fast, looks good, and easy to configure and use.

There are items in the Applications menu unique to WindowMaker which allow you to test window managers as well as adjust the appearance of your screen. It also comes with 'themes' installed which you can experiment with.

You can customise how WindowMaker looks by using the 'Appearance' submenu where, among other things, all the colors of the titlebar and window borders can be changed. WindowMaker has a special tool that allows you to configure the behaviour of the window manager with several options. Double-click on the 'Preferences Utility' icon to bring this up (the square icon with a circular image inside). Your configuration files are stored the in your `GNUstep/Defaults` directory. You can directly edit these files to customise WindowMaker. You can also change your background picture from this menu and any background images should be put into your `GNUstep/Library/WindowMaker/Backgrounds` directory.

When you are working, chances are that you are going to have more windows than can comfortably fit on the screen. Using WindowMaker, you can 'hide' windows, cycle through a clutter of windows and have several virtual desktops on which to

arrange your windows. The default number of desktops is four but this is easily changed to suit your needs.

You can find more complete and up-to-date information on WindowMaker by visiting the website. The address is `http://www.windowmaker.org`.

4.6.4 KDE

KDE stands for 'K Desktop Environment'. It has a very similar 'look and feel' to Windows95/98/2000. If you have used Windows, then you should be familiar with how things work in KDE.

A Panel of buttons is a key feature of KDE. It is used to launch applications as well as manipulate virtual desktops, which are useful in organising the way you work. Holding your mouse over any of the buttons in the Panel will launch a small 'tooltip' (or 'Balloon Help' for MacOS users) if you ever forget what a button's function is.

Other KDE features include window manipulation and finding lost windows in the clutter of the desktop. KDE also comes with a graphical file manager to allow you to navigate your home directory. If you have used the Windows or MacOS file managers, you should be familiar with the operation of the K File Manager.

KDE allows customisation to a certain degree. All of the options available for changing can be accessed through the 'KDE Control Center' — similar to the 'Control Panels' from Windows and MacOS.

From the Control Center you can modify things such as

- The buttons that appear in the titlebars.
- The colour of your windows.
- Your desktop backgrounds.
- Add extra applications to the panel.

The KDE Control Center is accessed from the Application Starter menu (click on the 'K' on the panel), or via the fourth button from the left on the panel, which opens up the control center straight away.

For more complete and up-to-date information on KDE, visit their website. The web site address is `http://www.kde.org`.

4.7 Changing your Window Manager

`chwm` stands for 'Change Window Manager'. It is a script which helps you change your window manager permanently. At the basic level, it will save a back up copy

of your `.xsession` file called `.xsession_old` (if you already have one), and then create a new `.xsession` file, configured for your new window manager.

To run `chwm`, you can select 'Change Permanently' in the 'Switch' item of your main menu (most window managers support this feature) or you can type the following at your prompt:

```
% chwm
```

`chwm` gives you a choice of 7 window managers for use within the CSE system. They are:

1. `fluxbox` — Modern, sleek and lightweight. It tries to leave you with as much screen space to yourself as possible. It is the default window manager described further in section 4.6.1 on page 71.
2. `twm` — A simple Tab Window Manager for the X Window System, described further in section 4.6.2 on page 72.
3. `fvwm2` — Version 2 of fvwm, which provides a 3-D look to window frames and a simple virtual desktop.
4. `ctwm` — twm with several virtual screens (workspaces).
5. `wmaker` — X11 window manager with a NeXTSTEP look described further in section 4.6.3 on page 72.
6. `startkde` — A windows-like GUI for Xpwd described further in section 4.6.4 on page 73.
7. `gnome-session` — The GNOME Desktop Environment works in concert with a window manager to provide extra functionality.

When you start up `chwm`, choose the number of the window manger you wish to use. `chwm` will then set up all the files needed to run that window manger.

Though `chwm` will only change your `.xsession` file, some of the window managers might require resource files to be created.

Feel free to experiment with the different window managers available. Do note that gnome-session (GNOME) and startkde (KDE) are large window managers which require a lot of disk space and CPU time to run. So, if you are Disk or CPU limited, we would recommend you use another window manager.

If you would like to switch back to your old .xsession, type:

```
% rm .xsession
% mv .xsession_old .xsession
```

This will remove your current .xsession and re-instate your old .xsession. Please not that the above command will *not* remove any settings files or directories created

by your current window manager. In most cases this will not prove to be an issue except when the `KDE` window manager is used. To remove all `KDE` directories and settings files, run the command:

```
% cleankde
```

For more information on changing your window manager see `http://www.cse.unsw.edu.au/faq/questions/account-windowmgr.html`.

4.8 XScreenSaver

A lot of emphasis is placed on security of accounts in the School. There will be many times you will need to leave your machine unattended to pick up a printout or take a short break. There is a program called `xlock` which will lock your machine while you are away, but often enough, you will forget to use it or underestimate how long you will be gone and leave your account unattended.

To prevent the tampering of accounts left unattended, a program called **XScreenSaver** will automatically lock your screen after it has been inactive for 10 minutes. The screen will appear black. In order to access your account again you will need to enter your password.

Please note, this is for your account security only, misuse of xlock to reserve facilities for yourself is unacceptable. Having your machine locked in excess of 15 minutes at any one time is considered too long and is not good lab etiquette. In many cases a public logout buton will be displayed after ten minutes providing other students with the opportunity to utilise a machine which has been locked for a large period of time.

5 Internet Tools

5.1 Electronic Mail

Electronic mail, or *email*, is an extremely convenient way of communicating with various users on the network. Each time you log in, a program called `checkmail` is started. It checks the spool area for any new email items addressed to you. If it finds any it places the items into the `.incoming-mail` file in your home directory where you can read them.

Each user has a unique email address allowing email messages to be sent to that particular user. The general form of this address is `username@sitename`, where `username` is the user's login name and `sitename` is the name of a site. Undergraduate users have usernames of the form `s<student number>` (eg. `s2123456`) or a combination of the letters of their (first, middle and last) names followed by three numbers selected at random (eg. `pats777`). The CSE School site, where your account is, has the sitename `cse.unsw.edu.au` so your full email address is

`username@cse.unsw.edu.au`
(eg. `s2123456@cse.unsw.edu.au` or `pats777@cse.unsw.edu.au`)

When sending email to local users (other students and staff in CSE) you can leave out the '`@sitename`' part of their address and just use their login name. There is no need to supply a sitename because the address will be resolved locally.

There are a variety of email readers available, `mail`, `pine`, and `elm` are three popular ones. You can also read mail using `Mozilla` or `Thunderbird`.

Different mail programs read email from different files. For example, `pine` and `elm` read email from your `.incoming-mail` file. If you then read your email from `emacs`, a file called `INBOX` will be created, into which all your email items from the `.incoming-mail` file will be moved. Reverting back to your original mail reader will result in a `no mail` error message or similar. If this happens, just move your `INBOX` file to your `.incoming-mail` file and all should be fine.

Different mail programs also store mail in different directories. For example `mail`, `elm` store mail in `/Mail` and pine stores it in `/mail`.

The best idea is to choose one email program and stick with it!

There are many Frequently Asked Questions documented on CSE's faq web page. For any mail related problems, you should refer to `http://www.cse.unsw.edu.au/faq/mail.html`.

mail

To run `mail`, type '`mail`' at the shell prompt thus:

```
% mail
```

and you should see something similar to:

```
% mail
4 messages:
No. From              Date         Len Subject
  1 lenka@cse.unsw.ed Apr  2 1993   21 Printer
  2 neilb@cse.unsw.ed May 20 1993   40 Re: NFS file locking
  3 geoffo@cse.insw.e May 26 1993   25 emacs v19
  4 markm@cse.unsw.ed May 28 1993   14 boat subnet
?
```

The email items that are found are listed as shown above.

To read a particular email item, type the number of the item, as listed, at the `?` prompt.

pine

To start pine, type

```
% pine
```

`pine` has built-in help, has a list of commonly used keys at the bottom of the screen and is very easy to use. Like `mail`, `pine` is a text-based reader. Although originally designed for inexperienced email users, `pine` has evolved to support many advanced features. There are an ever-growing number of configuration and personal-preference options which you can experiment with.

Redirecting Mail

It is simple to set things up so that email sent to your account at the School (eg. `pats777@cse.unsw.edu.au`) is redirected to another account, such as your UDUS account (see section 1.10 on page 33 about using UDUS). All you have to do is to type:

```
% mlalias -C <your CSE address> -a <destination address>
```

For example, if you want to redirect email from the CSE account `pats777@cse.unsw.edu.au` to the matching UDUS student account `pats@unsw.edu.au` you would type the following:

```
% mlalias -C pats777 -a pats@unsw.edu.au
```

This will set up a 'mail alias' with the same name as your CSE login name. In order to receive email at your CSE address *as well as* redirect it, you should type the following instead:

```
% mlalias -C pats777 -a pats777 pats@unsw.edu.au
```

As always, see the manpage for more information.

5.2 News

News is a public messaging system originating with USENET. In many ways it is similar to *email*, but instead of sending a message to an individual or to a small group a news message is broadcast more or less to the entire country or planet, depending on its topic.

To control just which portion of this cacophony you want to hear, news articles are classified into 'newsgroups', each group representing a different topic. There are several thousand newsgroups, and more are being created all the time. Most people read only a few of these.

5.2.1 Reading News

Before you start you must first create a `.newsrc` file. This file is used by newsreaders to keep track of which articles you have already read and what order you wish to read the newsgroups in. Most newsreaders will set this file up for you automatically the first time they are run.

To invoke `trn` type:

```
% trn
```

After doing the initial setup (the first time only), `trn` will present you with a prompt something like this:

```
********   1 unread article in news.answers--read now? [+ynq]
```

for each newsgroup in your `.newsrc` file. Newsgroups are presented to you in the order they are found in your `.newsrc`, so once you've decided what interests you it's a good idea to edit the file to move those groups to the top.

At this point you can do several things. The likely choices are shown in the '`[`' and '`]`' brackets as a reminder, but there are usually more choices than these. At any `trn` prompt pressing:

h shows a help listing of what you can do at this prompt.

space chooses the first option within the `[ ]` brackets.

Let's look at that prompt again:

```
********   1 unread article in news.answers--read now? [+ynq]
```

There are five common choices here:

+ or space Enter thread selection mode, where you pick which topics within the newsgroup you actually want to read. This is a win when you're only interested in some of the discussions.

q Quit trn.

y Enter the old rn-style article selection mode, where each article is presented in turn. This is usually only a win for low volume newsgroups.

n No, skip to next newsgroup.

U Unsubscribe from this group. It will not be offered to you again. Until you have cut your subscription down to something manageable this will likely be your most used choice.

5.2.2 Writing News

Of course, while there's a lot of interesting stuff out there, it's both fun and useful to add to the torrent yourself. *Before* you do so, remember that the network population is numbered in the millions.

Some guidelines:

- The net is a co-operative anarchy. Courtesy is all that keeps it usable. Courtesy comes in many forms, from being polite in wording your articles (usually a good idea) to being careful in deciding which group to post to and, indeed, whether to post at all.
- Be familiar with a group before posting to it. Read it for at least a week to get a feel for its intended subject matter and style.

- Compose your article in an editor. Run a spell checker over it. Proof read your article.
- Read the newsgroup `news.answers`. Much useful information is posted here.
- If your posting is a query, make some attempt to find the information locally before you resort to posting news.
- Don't cross-post! Posting one and the same news item to multiple news groups is frowned upon. Choose only *one* of the most appropriate groups to post to.

Ok, so you want to post news. There are two basic methods of posting:

1. Posting a followup. This means writing an article in response to another article, continuing a discussion. From within `trn` typing 'F' will do this. There will be equivalent methods in the other newsreaders. Try to read the entire thread before posting a followup, it's likely that someone else had a similar thought and beat you to it.
2. Posting a new article, starting a new thread. This is done with the `Pnews` command:

```
% Pnews
```

Remember to think carefully about which newsgroup should get the article.

Here are some more guidelines:

- Don't post requests such as 'where can I find this program?' to those news groups devoted to discussions of source code (`comp.sources.misc`, etc). There are ancillary groups for requests like this (`comp.sources.wanted`, etc).
- Don't post to a world-wide group when a local (Australia wide) group will do. For example, try `aus.ads.wanted` rather than `misc.wanted`.
- If your request is more regional than the group, try to restrict its distribution with the `Distribution:` header (such as '`Distribution: syd`' for just Sydney).
- Again, if your request is more regional than the group, make this clear in the subject line. For example, a for sale notice should go in `aus.ads.forsale`, but people in Perth won't be that interested in an alarm clock for sale in Sydney. Mark the subject line like this:

```
Subject: [Sydney] alarm clock for sale
```

to save them the effort.

5.2.3 Reading News with a Web Browser

In order to read news from a web browser, you will have to tell the browser where the news (or nntp) server is. The University's nntp server is at `nntp.unsw.edu.au`. With `Mozilla` and `Thunderbird`, this is done from the `Window:Mail and Newsgroups` dialog box.

To get a list of all available newsgroups, you can open the URL `news:*`. The results of this vary with the browser. `Mozilla` sorts the groups into alphabetical order and joins groups with the same prefix together. You can subscribe to newsgroups either by viewing the group and clicking *Subscribe*, or clicking the box to the left of the group in the list of groups and pressing *Subscribe to selected newsgroups*. `Mozilla` labels the list of subscribed newsgroups as `newsrc://nntp.unsw.edu.au/`. It will save the list of newsgroups in `.newsrc-nntp.unsw.edu.au`

5.3 FTP

FTP stands for **F**ile **T**ransfer **P**rotocol. It is used to transfer files to and from remote machines (such as from your computer at home to your account at CSE or vice versa). It is far better to use `ftp` to transfer files than to use `email`. `ftp` is much quicker, and `email` runs into problems when large files clog up people's `.incoming-mail` files and/or push them over quota. Also, many mail programs refuse to deal with very large email messages.

In general to start an `ftp` session you would use:

```
ftp remote-hostname
```

where `remote-hostname` is either the full name of the machine or the *IP number* of the machine (or simply the machine name for local machines).

At CSE, there is an ftp server, so the host to connect to is `ftp.cse.unsw.edu.au`.

`ftp` will then attempt to make a connection with the specified machine. If it is successful in making a connection, you will be prompted to log in, as shown below.

```
% ftp xanthic Connected to xanthic.spectrum.cs.unsw.OZ.AU.
220 xanthic FTP server (Domain/OS FTPD) ready.
Name (xanthic:cathy): cathy
331 Password required for cathy.
Password:
230 User cathy logged in.
ftp>
```

Anonymous Ftp is often used to download software from public ftp servers. For Anonymous Ftp, log in with the username `anonymous` or `ftp` (both usually work) and use your full email address as the password.

5.3.1 FTP Commands

open *remote-hostname* — This will attempt to open a connection with the specified remote host.

close — This will close your current connection without exiting ftp.

help — This will give a listing of valid ftp commands.

help *command-name* — This will give a brief description of what the command does.

dir or **ls** — This will give a listing of the contents of the remote directory.

lcd *directory-name* — This will change your local working directory.

cd *directory-name* — This will change your remote working directory.

binary — This is used to set the transfer type to binary (or image 'I') mode. By default, ftp transfers files in *ascii* mode, this is unsuitable for some files eg. executables or compressed files (ie. those files with a `.Z` suffix). This also applies to *Microsoft Word* and *Excel* files.

To transfer such files you will need to explicitly specify that transmission is to be in *binary* mode. All subsequent transfers for the ftp session will be in binary mode until you set the transmission mode back to *ascii*.

ascii — This sets the transmission to ascii mode. You will use this mode when transferring text files. It automatically handles any file conversion that is necessary for example between MS-DOS and UNIX file formats.

get *filename* — This will transfer the specified file from the remote machine to the local machine. The file is placed in the current working directory on the local machine.

mget *filenames* — This will transfer the specified files from the remote machine to the local machine. The files are placed in the current working directory on the local machine. The files can be specified using patterns eg. `mget *.mod` will fetch all files with a `.mod` suffix from the client machine.

put *filename* — This will transfer the specified file from the local machine to the remote (or client) machine. The file is placed in the current working directory on the remote machine. Note that you must have write permission to this directory.

mput *filenames* — This will transfer the specified files from the local machine to the remote machine. The files are placed in the current working directory on the remote machine, note that you must have write permission to this directory. The files can be specified using patterns eg. `mput *.mod` will copy all files with a `.mod` suffix from the client machine to the local machine.

quit — This will exit ftp.

5.4 SCP

A more secure replacement for `ftp` is **SCP** (Secure Copy). SCP operates through an SSH (Secure SHell), and as a result, provides for a more secure data transfer method than ftp. SSH is discussed in more detail in section 1.10.2 on page 34.

Transfer the file `file.txt` from home to to a CSE account (user `fpoacha`).

```
scp local_file username@hostname:remote_file
```

```
% scp ~/file.txt fpoacha@williams.cse.unsw.edu.au:file.txt
```

Retrieve the file `file.txt` from a CSE account and save it as `localfile.txt` at the destination (your home computer for example):

```
scp username@hostname:remote_file local_file
```

```
% scp fpoacha@williams.cse.unsw.edu.au:file.txt localfile.txt
```

In both examples, the server `williams.cse.unsw.edu.au` is used to perform the transfer, however any CSE server that supports SSH can be used for this purpose.

As usual, there is a `man` page for the `scp` command which describes **SCP** and its features in more detail.

Use

```
% man scp
```

to read the manual page for this program.

5.5 The World Wide Web

The World Wide Web began in 1992 at CERN, the European Laboratory for Particle Physics, as a means of distributing and annotating scientific research.

It now contains squillions of documents containing information on an enormous number of varied topics. The protocols that define the web specify three sets of rules for creating, publishing, and finding documents:

HTML (HyperText Markup Language)

Web documents are ordinary text files that can be created with any word processing program or text editor. There are also HTML editors

available. Simple ones are included in the latest versions of `Mozilla` and `Internet Explorer`.

HTML documents include *tags*, or markups, which control the appearance of the text in the document. For example, to emphasise a word in your HTML document, you might write something like this:

```
This is <EM>emphasised</EM> text.
```

Which might appear in a web browser something like:

This is *emphasised* text.

More accurately, the tags describe the structure or meaning of the text. Any web browser can then make the best choice of how to represent the text on a particular output device. For example, a browser/device pair that cannot display *italic* text may display the above example as underlined text. Indeed, the tag `<EM>` means **emphasised**, not *italic*.

Another sort of tag can define something as a *link*, more technically referred to as an *anchor*. Selecting a link lets the user go to another document (or to another section of the same document).

HTML documents (often called *pages*, or *web pages*) can contain other sorts of data, such as graphics, audio, video and even interactive programs such as java applets. The ability to view such data depends on the browser being used.

HTTP (HyperText Transfer Protocol)

Users of the web retrieve documents from *servers* (or *web sites*). HTTP allows a networked computer to listen for and respond to incoming requests for files (*hits*).

URL (Uniform Resource Locator)

A URL is the address for a web document or other file. The URL for the School's *home page* looks like:

```
http://www.cse.unsw.edu.au/index.html
```

This requests a file called `index.html` be retrieved from the server `www.cse.unsw.edu.au` using HTTP. The location specified between the first // and the next / is your internet host.

Other sorts of URLs you may encounter start with:

`file:` Identifies a file which the browser must find, which is often from your machine (the `//hostname` may be missing).

`ftp:` Identifies a file that can be downloaded from an FTP server.

`gopher:` Identifies a file on a gopher server.

`mailto:`	This will enable you to send an email to the specified address.
`news:`	Identifies an item from a news group.
`https:`	Secure HTTP (encrypted connection).

5.5.1 How to Look at Web Pages

There are many different web browsers (programs for looking at web pages). Some names of web browsers are: `Lynx` (a text only browser), `Firefox`, `Mozilla` and `Internet Explorer` (both graphical browsers). The browser which you will be using in the labs will be either `Mozilla` or `Firefox`. You will probably want to run `Mozilla` in the background (by typing `mozilla &` in an `xterm`) or `Mozilla-Firefox` by typing `firefox &`.

Microsoft's *Internet Explorer* is not available for the UNIX operating system so we will not be discussing it here. It behaves in a similar way to *Mozilla-Firefox* and *Mozilla*.

There are quota limits associated with web usage. See section 1.9.6 on page 32 for further details.

5.5.2 Common Browsers

Listed below are some of the common browsers you might consider using:

Mozilla	This is the default and most supported broswer used. It has a similar interface to `Netscape` whilst being inherently more stable.
Mozilla-Firefox	This is a very fast, lightweight browser with several intuitive features. It is closely related to the `Mozilla` browser broswer and is available in both Windows and Linux/ Unix versions.
Netscape	This is one of the most famous, and commercially successful browsers of all time. It is no longer in use within CSE due to a lack of support by `Debian` Linux after `Netscape` version 4.77 was released.
Galeon	The current release of `Galeon` can only be used on the `Gnome` window manager. Until a generic version is available for use, it is unlikely to gain support within CSE.
Konqueror	`Konqueror` is an inbuilt browser distributed with the `KDE` window manager. It isn't as stable as `Mozilla` and as such is not recommended.
Opera	A very small, efficient browser that has gained popularity in recent times. `Opera` can struggle on web pages with unusual applets or features, however it continues to enjoy a loyal following of users due to its speed.

You are welcome to use any browser you wish, including those not listed here. Please note, however, that browsers aside from the standard `Mozilla`, `Mozilla-Firefox` and `Konqueror` will need to be manually installed and administered yourself.

5.5.3 Common Problems with Browsers

In order to access external sites on the World Wide Web, you will have to use the web proxy. To use the School's proxy server you need to tell your browser which proxy to use. The simplest way for `Mozilla` and `Mozilla-Firefox` users is to set the proxy to `NetscapeProxyConfig.pac`.

This is done differently depending on the browser you are using. Details are available on the web at:

`http://www.cse.unsw.edu.au/faq/questions/www-proxysetup.html`

If you find your browser crashes often, you can minimise the chance of this occuring by setting your preferences as follows:

Turn off JavaScript in `Edit:Preferences:Advanced`. There is a lot of badly written JavaScript out there which your browser may have trouble coping with. More rarely, you might need to turn off Java as well.

Use a disk cache of **ZERO**. There is no advantage in using a local disk cache as `Mozilla` uses a cacheing proxy server that caches files within CSE anyway. It also stops `Mozilla` using all your disk quota.

Use a small memory cache of 1 MB.

If you get stuck on a page that keeps crashing your browser, clear the disk cache (if you are using it) and memory cache and turn off JavaScript, before attempting to load the page again.

Other Frequently Asked Questions relating to browser support are available at:

`http://www.cse.unsw.edu.au/faq/www.html`

5.5.4 What is a Home Page?

A *home page* is a web document. Your home page is set in your browser settings is the first page you see when you start the browser, or press the *Home* button.

In `mozilla` and `Mozilla-Firefox` your *home page* can be set up under `Edit:Preferences:Home Page`.

Unless you change it to something else, the School web page will be your default home page.

5.5.5 Creating Your Own Web Pages

There are lots of good (and even more bad) web pages that tell you how to write HTML. Now that you know how to look at web pages, you might want to look at `http://www.cse.unsw.edu.au/web/weblore.html#authoring`. It contains a list of several on-line tutorials and guides to HTML, among other things. Be warned that some of the information there may be outdated. An updated version is under construction.

Note that in the *yellow form* (discussed in section 1.6.3 on page 24) which you signed before getting an account on the School's computers, there is a section which states, in part:

Be advised that the following are not allowed:

- *Using the facilities for commercial purposes, or for preparing, storing or displaying racist or other offensive material.* This includes displaying pornographic or offensive images as screen backgrounds.
- *Storing copyright material in a publically accessible place.* Note that anything written by somebody else is copyrighted to them and may not be reproduced without permission. This includes graphics.

You should keep this in mind when designing web pages. Your web pages should not contain offensive or copyright material, or be used for commercial purposes. If your web pages have a detrimental effect on the School's network or the image of the University, you will be asked to remove them. You may also be in breach of the rules relating to the student use of electronic facilities as well as the yellow form. See section 1.6.3 on page 24 for more details.

To create a web page, you just need to make a text document containing HTML commands and text. The HTML source code for a very minimal web page is shown in figure 5.1. You can use your favourite editor to compose a web page or you can use a program such as `composer` to create or modify it.

You can expand your knowledge of HTML by examining the way other people have written their pages. To see the source code (HTML) of the web page you are looking at in `Mozilla` (and `Mozilla-Firefox`), simply go to `View:Page Source`.

For your web pages to be visible to you and to the rest of the world, they will need to be placed in a directory called `public_html` in your home directory. This directory (and your home directory) must be *world-executable*, and the HTML files must be *world-readable* (see section 2.2.16 on page 50). Note also that if your HTML files do not end with one of the suffixes `.html` or `.htm`, they may not be recognised as HTML files by all browsers.

Figure 5.2 shows a directory listing from the `public_html` directory of a user called *markn*. This listing illustrates the required permissions for your `public_html` directory and the `.html` files you put there.

```
<HTML>
<HEAD>
<TITLE>A Very Simple HTML Document</TITLE>
</HEAD>
<BODY>
<H1><CENTER>
A Heading For This Page
</CENTER></H1>

Some sample text.

<p>Some more text in a separate paragraph.

</BODY>
</HTML>
```

Figure 5.1: A very simple HTML document.

```
% pwd
/home/markn/public_html
% ls -al
total 49
drwxr-xr-x   4 markn    markn       8192 Feb 13 17:39 .
drwxr-xr-x  19 markn    markn       8192 Feb 12 16:59 ..
drwx--x--x   2 markn    markn       8192 Jan 12 12:33 docs
-rw-r--r--   1 markn    markn       4099 Feb  8 16:55 humour.html
-rw-r--r--   1 markn    markn       1657 Feb 12 13:53 index.html
-rw-r--r--   1 markn    markn       7328 Nov 10 14:39 iso-latin.html
-rw-r--r--   1 markn    markn       1684 Jan 19 14:47 links.html
```

Figure 5.2: Example `public_html` listing.

The URL for a file `file.html` in the `public_html` directory of an account belonging to the user `pats777` would be:

`http://www.cse.unsw.edu.au/~pats777/file.html`

Useful URLs

The pdf and html versions of the 2007 Concise Unix Primer can be found online at:

`http://www.cse.unsw.edu.au/help/doc/`

For all academic information and announcements the School of Computer Science and Engineering homepage is at:

`http://www.cse.unsw.edu.au/`

The first place to look when you have a CSE account or system enquiry is the Frequently Asked Questions webpage:

`http://www.cse.unsw.edu.au/faq/`

`https://csg.cse.unsw.edu.au/~csg/`

For information on the School's Computing Facilities including labs, printers, servers and architecture information:

`http://www.cse.unsw.edu.au/help/computing/facilities/`

The CSE Help Desk webpage contains information about the services and facilities offered to CSE students, as well as useful links to various student-related resources:

`http://www.cse.unsw.edu.au/~helpdesk/`

Your local mirror for *free* software downloads such as Linux, GCC and Mozilla as well as links to other local and externa;l mirrors:

`http://mirror.cse.unsw.edu.au/`

The CSE Student Representative group are your contact with the School, representing students at various events:

`http://www.cse.unsw.edu.au/~stureps/`

For information regarding social events, seminars and student support visit the CSE affiliated Societies and Clubs webpage:

`http://do.cse.unsw.edu.au/students/clubs/`

The Home Computing Software CDROM webpage provides help with installing and using the software on the CDROM, available for purchase from the CSE Help Desk:

`http://www.cse.unsw.edu.au/homecomputing/`

Important advice relating to the responsible use of laboratories and information on academic misconduct:

`http://www.cse.unsw.edu.au/~studentoffice/policies/yellowform.html`