
# An Introduction to the Computing Facilities in the Department of Computer Science

Dr. Graham Roberts
email: G.Roberts@cs.ucl.ac.uk
URL: http://www.cs.ucl.ac.uk/staff/G.Roberts
This version: September 8, 2012

# Contents

# Part I
# The Basics

## 1 Introduction

Welcome! This document is a beginners' guide to using the computing facilities in the Department of Computer Science[1] at UCL. The aim is to introduce you to a selection of the essential facilities, while pointing you in the direction of more advanced features and additional documentation.

There is no substitute for actually sitting in front of a computer and using it if you want to become a proficient user. When you first get this guide read through it once fairly quickly and then spend as much time as possible trying everything out on a real computer in the department, referring back to the guide as necessary. Do not worry if parts of this guide are difficult to follow at first! It will quickly start to make sense as you familiarise yourself with the way things work.

The most important things that you need to know about are:

- logging-in to a computer[2] and logging out,
- getting familiar with the user interface (windows, menus and so on),
- organising files in your filestore,
- creating and editing text files,
- printing files,
- reading and sending email,
- and the department's and UCL's web sites.

It is very likely you have done most or all of the items listed above on your own or other machines. However, the purpose of this document is to introduce you to the computing environment in Computer Science, and also to give an introduction to Unix, which you may need to use quite extensively depending on the degree programme you are on.

The computers you will be using are all located in a number of computer labs on the first and fourth floors of the Malet Place Engineering Building (MPEB) where Computer Science is based. The departmental staff are located on floors four to eight, with the Reception Desk on the 5th floor. As soon as you can, visit the labs and have a look around. See Appendix A page 34 for the full list of labs. The Computer Science Helpdesk is also located on the 4th floor. The Helpdesk provides support for the computing facilities and labs run by the department.

### 1.1 Using Your Own Computer

While this document describes the computing facilities provided for you in the department, there is no reason why you can't use your own computer. Note, though, that you are responsible for configuring and maintaining your own computer, as well as keeping it secure and safe. The department can provide only limited help if you have problems.

Wireless network access is available throughout much of the UCL campus via the eduroam service. See `https://www.ucl.ac.uk/isd/students/wireless` for information on how to access eduroam. The department also provides a separate local wireless service for student use in parts of the MPEB, including some of the labs. Contact the Computer Science Helpdesk to find out how to get connected to the departmental network.

[1] Computer Science runs its own independent computing service that can be used only by students on degree programmes supported by the department. A different department (Information Services Division or ISD) runs UCL's campus wide computing service that all students have access to, including Computer Science students. This document is mostly about using the Computer Science service.

[2] Logging-in is techno-speak for telling the computer that you are about to start using it. There are all sorts of bits of computer jargon like this, confusing at first but don't worry in a few weeks you will be using the jargon like a pro!

# 2 Rules and Regulations

We do have rules and regulations covering the use of labs and computers. These can be viewed on the departmental web site. However, rather than listing them here, we introduce the idea of *lab etiquette*. We hope you will cooperate with each other and make sensible use of the facilities in the department, without us having to enforce the rules. For the record, however, if individuals do behave antisocially or misuse the facilities, then there are measures that will be taken, including suspension and loss of access to the labs. In particular, strong action will be taken if anyone is found trying to circumvent the security systems, using our systems to gain illegal access to other computers, or using our systems to access illegal material.

**The use of P2P or torrent software is strictly prohibited and you should always respect the rights of copyright owners by never copying or distributing copyright material without permission.**

## 2.1 Lab Etiquette

The following is an informal list of principles we would like you to observe:

- Keep the labs as quiet work areas (like a library) – don't hold loud conversations or play music out loud.
- Use the computers for academic purposes only.
- If you are not doing anything useful, always give up a computer to someone who has real work to do.
- Don't do a remote login and overload an already busy computer – always check the load first (this will make more sense as you read on!).
- Don't send silly or offensive electronic mail. Don't send junk mail or spam.
- Don't display images on the computer screen that others might find offensive.
- Collect printouts and leave the printers tidy.
- Don't spill food or drink over the machines, especially the keyboard. Don't take food or drink with a strong smell into a lab.
- Be nice!!

The labs are open to all groups of Computer Science students (Undergraduate and Postgraduate), so you will be mixing with students from all the department's degree programmes. Labs can be booked for lab classes, so at times a particular lab will not be available. Otherwise, labs can be used at any time that the building is open for student access.

If there are problems or you have any ideas for improving the arrangements, contact your student rep or the Departmental Tutor. Issues can also be raised via your student rep(s) at the Staff-Student Consultative committee meetings.

Don't forget that resources are finite and should be used for academic work only.

## 2.2 Building Security

**If there is an emergency or an intruder in the building, and you are able to, use an internal phone to dial the UCL emergency number 222. Tell the operator what has happened, that you are in the Malet Place Engineering Building and, if possible, which room you are in.**

Internal phones are prominently located in all labs and teaching rooms. Other than using the fire alarm if there is fire or smoke, the internal phones are the quickest way of getting help, especially if there is a medical emergency, so use them if possible.

Building and personal security is important, especially as we are in a central London location, so always be on the lookout for suspicious people. Use the emergency phone number (222) if you think there is something wrong, or go in person to the Helpdesk or Reception Desk (during normal hours), or to the security office in the Engineering building next door (at the entrance to the Roberts building next to the back gate). When working in a lab take care to watch over your personal belongings (bags, purses, music players, notebook computers, etc.). Also make sure that entrance/exit and lab doors are properly closed after you pass through them.

As you become familiar with the department, you will notice that some doors require an id card for access. If your id card is not authorised for the door (undergraduate and MSc student cards are not authorised for doors within the department) then please do not use these doors or try to follow a card holder through the door without permission.

Many doors in the building are marked as fire doors, using small round blue signs. These must be kept closed at all times other than when people are passing through them. If you find a fire door that is open and no one is passing through, please close it. Note that it is a criminal offence to leave open or prop open a fire door (especially using a fire extinguisher as a prop), so please do your part to make sure that fire doors are properly closed.

If you find a problem with a fire door or find a fire extinguisher is missing please notify a member of staff or the Reception Desk as soon as possible.

## 2.3 Security Cameras

You will notice that around the building, and UCL in general, there are a number of CCTV security cameras. In addition, some labs may have cameras, which will record all activities in the labs. The primary purpose of the lab cameras is to help prevent the theft of valuable computer equipment, rather than to observe people working the labs. The cameras are operated in accordance with provisions of the Data Protection Act and recordings are only viewed if an incident occurs.

## 2.4 ID Cards

When you enrol with UCL you will be given an ID card with your name and photograph on it. You should carry your ID card with you at all times and be prepared to show it if required. To gain entry to many UCL buildings, including the MPEB where Computer Science is located, you must use your ID card, so don't forget to carry it with you. If you lose your card, report the loss as soon as possible and arrange to get a new one.

## 2.5 Fire Safety

**When the file alarm goes off you MUST LEAVE THE BUILDING and quickly make your way to the assembly point. The UCL safety officers will take action against anyone found to have ignored a fire alarm.**

Emergency exits and routes out of the building are marked by green Emergency Exit signs. If the fire alarm goes off, then as quickly as possible leave the building following the green signs. As you become familiar with the building take note of these signs and where the emergency exits are. There are fire alarm buttons and fire extinguishers located in the labs and in many other places around the building.

You should normally leave the building via the nearest emergency exit, following the routes given by the green signs and taking into account any signs of fire or smoke. Do not simply try to leave the building via the way you came in (typically the main entrance) as this may delay your exit and, worse, lead you into a fire or smoke.

You may need to open an emergency exit door that is normally kept closed. Do this by pushing on the opening bar. Don't worry about opening emergency doors; the most important thing is that you exit the building as quickly and safely as possible.

During a fire alarm Fire Evacuation Marshals, wearing yellow safety vests, will search the building and direct people out of the building to the assembly point. Please follow their instructions. Also, inform a marshal if you believe anyone is left behind or trapped in the building.

Once out of the building make your way to the assembly point at the South Junction where the Print Room Café is (turn left outside the building, walk along Malet Place and through the short tunnel to get there). Do make sure you find out where the assembly point is. Please do not congregate immediately outside the building as this will obstruct others and would be dangerous if a fire develops. Also don't go and stand around the entrance gate or in Malet Place – you will get in the way of London Fire Service when the fire engines arrive.[3]

It is important that you do go to the assembly point rather than walk away and go elsewhere. If you are reported missing (you may get separated from a friend who believes you are still in the building, for example), the fire marshals will look for you at the assembly point, while the fire brigade will be searching the building for you.

If you are the one to discover a fire or smoke in the building then immediately press the nearest fire alarm button and exit the building. Do not attempt to fight the fire yourself. While you will see many fire extinguishers around the building and may be tempted to use them, do not do so unless you are absolutely sure there is no risk to yourselves or others. If in any doubt, leave. Note that there are different kinds of extinguisher and using the wrong kind can make the situation worse.

Remember – when the fire alarm goes off your priority is to leave the building as quickly as possible via the nearest emergency exit and make your way to the assembly point.

### 2.6 Opening Times

The building and the labs are normally open for all students from 8am to 7pm on weekdays only. At all other times undergraduate students are not allowed in the building (this is for Health and Safety reasons). If you are in the building at closing time, you should leave before the security patrol arrives; failure to leave is treated as a serious offence by the UCL authorities. If you do meet the security patrol at closing time, please be polite and leave as requested.

At various times during the teaching terms, notably at the end of term when there are many coursework and project deadlines, the building may close later. Late closing may also be arranged on a regular basis, one or two days a week during term time, if there is sufficient demand. These extended opening times will be advertised as and when they are arranged. However, do not rely on them taking place.

Access to the building for students on MSc programmes may be more flexible. The appropriate course director or tutor will provide details. Research students have the same access rights as staff.

## 3 Getting Started with the Computers

### 3.1 Lab Computers

Computers in the department run one of three operating systems: a version of Linux, OS X or Microsoft Windows. Each operating system defines its own version of the now very familiar graphical user interface or desktop metaphor, so be prepared to switch between the different styles. However, all the computers are networked together, meaning you are able to use any computer without having to worry about access to files and resources.

The computers in a specific lab will all run the same operating system by default but some allow the user to choose which operating system to start up with (this is called dual-booting). In addition, the department's *virtual computing service* allows access to *virtual machines* that run in a window on your desktop regardless of what operating system the computer is currently running. A virtual machine can be running one of a large variety of operating systems and

[3] Fire engines always attend if the alarm goes off for real, so watch out!

versions, all from the same physical lab computer.

Later sections in this document will give a more detailed description of using Unix-based operating systems (such as OS X and Linux)[4], as Unix is less familiar to many people than using Microsoft Windows, and you may be taking modules that require the use of Unix. More importantly any Computer Scientist needs to know how to use Unix fluently.

Each computer has a name so that it can be uniquely identified and you will notice many of the computers are named using a theme such as railway station and place names. The name of a computer is usually displayed either somewhere on its visible casing or somewhere on the screen.

The network in this department is connected to the UCL network, the UK academic network (called JANET) and to the Internet (the global network). In fact, all the computers are directly part of the Internet and all the facilities such as the World Wide Web (WWW) are available (more later).

Section A in the appendices of this document lists the computer labs, where the computers are located.

## 3.2 Your User Account

Access to the computer resources in the department are carefully controlled. In order to use a computer you must first identify yourself as an authorised user before you can start work. To become an authorised user, you need to go through the registration process at the start of the academic year so that an account will be created for you. To use your account you are given a username and a password, which is intended to prevent anyone other than you from using your account.

### 3.2.1 Changing Your Password

When you are first given an account you will get a password that has been generated at random. You may want to change this to something that you can remember more easily, but that is not easy for other people to guess. How you change your password depends on what kind of computer you are using. For Unix computers see page 17. For Windows computers you usually use the password dialog, which is accessed using the Setting/Security menu via the Start Button Menu.

Your new password will only be accepted if it has at least six characters (including digits), not too many duplicate characters, no words from any European language, or the names of people, places, pop groups, football teams, swear words[5] or any other word found in a series of special dictionaries. If your password is considered too obvious for one of these reasons, it will be rejected and you will need to find another.

Just in case you are wondering how on earth you will find an acceptable password try something like a combination of two or three short but unrelated words with some numbers added (e.g., the3you7not6), or remember a phrase and take the first letter of each word in the phrase with numbers added to form your password. For example, taking the phrase 'The cat sat on the mat' plus some numbers would give 't3c4s5o6t7m'. Your password should also really include some capital letters and the other non-alphabetic characters that can be typed. For further details about passwords see:
http://tsg.cs.ucl.ac.uk/basics/faqs/general/
It is a very good idea to change your password few months or so. Also, change it immediately if you think that someone else has found out what it is.

If you forget your password visit the Helpdesk and you will be given a new one. To slightly complicate things you will find that you really have two CS passwords, one for use on Windows machines and one for Unix machines. If you change your password via Windows, the Unix one will be left unchanged and vice versa. Your username is the same on both kinds of system.

[4] This document will use the generic term 'Unix' to refer to unix-based operating systems. However, strictly speaking Linux, really GNU/Linux, is not Unix just very similar (GNU is GNot Unix!). OS X v10.7 and 10.8 are real certified versions of Unix. Microsoft Windows is very definitely not Unix of any sort.

[5] Far too easy to guess!

**You must NOT use any other persons account.**
**You must NOT allow someone else to use your account.**
**NEVER tell anyone else what your password is.**
**Any attempt to break the security system by any means, or any attempt to use Computer Science facilities to break into other computers, is treated as a serious offence. Discovery can result in the suspension of your account and, possibly, suspension from your degree course.**

### 3.2.2 CS v. ISD User Accounts

Your user account created by the CS department is for CS computers only. You will have a another and completely separate user account for UCLs Information Services Division (ISD) computing service. Your ISD account will have a different username and password. While much of your academic work for CS can be done using CS computers, you will need to frequently use ISD computers and services as well, in particular the email service, the Portico student information system and the Moodle on-line learning system. So, make sure you remember both usernames and passwords and don't get them muddled up!

Detailed information and documentation about using the ISD services is available on the Web (http://www.ucl.ac.uk/isd) or from the ISD Service Desk on the ground floor of the DMS Watson Science Library (this is the building to the right of the MPEB when facing the MPEB entrance doors).

## 3.3 General Use of the Computers

It is reassuring to know that, during normal use, you cannot damage a computer in any way by pressing the wrong keys, or by making mistakes when using it (typically you will just hear a beep or see an error message).

There are several things, however, which you never need to do and should not try:

- **NEVER switch any computer or printer on or off**. This job should only be carried out by members of the Technical Support Group (TSG). This is in contrast to what you are probably used to, particularly with computers at home, but is essential to ensure that computers are not damaged and information is not lost. When you have finished using a computer you do not need to switch it off. If a computer or printer is switched off there is probably a good reason for it being out of use, so leave it alone!
- **Computers should not be arbitrarily rebooted using hardware resets or software controls**. It is not possible to do this by accident, so if this caution does not mean anything at the moment it does not affect you. This applies to printers as well as computers.
- **Do not change any configuration settings on computers and printers, or unplug any cables**. A computer or printer may have a panel of control buttons or switches, which should not be touched. If a piece of hardware is not responding normally, report the problem to the Helpdesk.

If you discover any faults with any of the equipment in the department, send an email message to request@cs.ucl.ac.uk to report the problem (use of email will be covered later). Describe the symptoms, the location and which item is involved. Alternatively, or in case of a more urgent problem, phone the Helpdesk using the number 37280[6], or visit the Helpdesk in person in room 4.22 on the 4th floor (normal opening hours are 0930-1700 Mon-Fri). Internal phones can be found in all the computer labs.

## 3.4 Logging In

When you first approach a computer you might find that the screen is dark. If so, pressing a key will light the screen up, revealing a login screen displaying a login prompt. In order to access your account you go through the familiar login process of entering your username and password. If you enter your username or password incorrectly without

[6] The Helpdesk is open during normal working hours and is staffed by members of the Technical Support Group (TSG). Further information can be found on the web at: http://tsg.cs.ucl.ac.uk/index/.

realising it, the computer will reject your attempt to login and you will need to try again. If you repeatedly fail to login, the computer may prevent further attempts for several minutes; this is to reduce the risk of someone else trying to guess your password to gain access to your account.

Once you have correctly entered your username and password the computer will initialise your login session and display the appropriate desktop for you to use.

### 3.5 Logging out

After you have finished using a computer you need to logout – the inverse from logging in. This is usually done via a pop-up menu of some sort or via the Start Menu in Windows.

When you logout your desktop will disappear and be replaced by the login prompt display ready for the next user. You don't have to turn the computer off; just leave it running.

**If you find a machine where someone has forgotten to logout, then logout on their behalf (but make sure they have not simply left the room for a couple of minutes!). Please NEVER use another person's account at any time. UCL is obliged to deal strictly with those that abuse computer facilities, especially electronic mail.**

# Part II
# The Web

The Web is an important source of online information and you will need to make regular use of a number of UCL websites, in particular the CS department's site and the services on the main UCL site.

**Please use the Web responsibly. Access is provided to you on the assumption that you will use it for genuine academic purposes. Yes, there are all sorts of bits of dubious information that are easy to get at, some of it illegal in this country. Do not attempt to access it – if you are caught UCL will act quickly to stop you, with suspension likely to be the result.**
**It is possible to waste a great deal of time browsing around the web! Always give up a machine to someone who has real work to do.**

## 4 Using the Web

### 4.1 Browsing

The Web can be accessed using a variety of web browsers. Mozilla Firefox is widely used and supported on all CS computers. When you start a web browser you will see the default page, possibly the CS Teaching Home Page but more likely the standard home page of the browser. You can use the browser preferences to set whatever home page you like.

Usually you will use a pop-up menu to start a browser but on Unix systems you can also type in a command – this will be explored later. Make sure you are proficient at performing web searches and bookmarking. Effective searching

can save a lot of time and frustration. The best general purpose search engine for academic purposes is usually Google at http://www.google.co.uk/.

## 4.2 URL's

Information on the Web can come from anywhere in the world. Each section or page of information is named or located by a URL (Uniform Resource Locator), which is sort of the Web equivalent of a postal address. URLs look like this: http://www.cs.ucl.ac.uk/
The 'http:' element denotes the communication protocol, while the rest gives the location. The location information starts with the name of a web server (a machine that stores web pages) and is optionally followed by a location on that server. For example: http://www.cs.ucl.ac.uk/people.html names the file 'people.html' on the server 'www.cs.ucl.ac.uk', which is a web server in the Computer Science department.

The UCL website can be accessed at: http://www.ucl.ac.uk, where extensive information about all the department's research, teaching, services and student support activities can be found.

Information about the Information Services Division computing service can be found at http://www.ucl.ac.uk/isd/.

Make good use of your browsers book marking system to keep track of the important pages you find as you browse.

## 4.3 Student and Teaching Support Pages

The home page for Computer Science student information is at:
http://www.cs.ucl.ac.uk/students/.
It contains links to all sorts of teaching related information relevant to your degree. This includes timetables, syllabus information, general documentation and lots of other useful stuff. Make sure you have a good look around so that you know where to find things.

You will find that lecturers put a large amount of teaching information onto the web, particularly on the UCL Moodle system (this will be introduced in lectures). Also, use the web to search for other information on the subjects you are learning about.

If you have suggestions about the content of the student support web pages, things that are missing or additional items that should be added, please email the Departmental Tutor (G.Roberts@cs.ucl.ac.uk).

## 4.4 Searching the Local Web

The local Computer Science web has its own indexing and searching service. You can perform a search by using the Search box that appears many pages. Learn to use this effectively as it can save a lot of time.

## 4.5 Portico

The UCL Registry maintains an extensive set of web pages that you should be familiar with as they hold information about many aspects of UCL and its rules and regulations. The main index page is at: http://www.ucl.ac.uk/current-students. On the right hand side of that page is a link to the Portico system. Portico is the UCL Student Information Service and it holds detailed information about your studies at UCL, including your registration information, what modules you are taking and, when available, your results. It is very important you know how to access and use Portico.

Your personal contact details are kept in Portico (address, phone number, email address, etc.) and you need to keep this information up to date. If the department or UCL needs to contact you the Portico information will be used. If it gets out of date you risk missing important communications.

To enter Portico you must first login as directed on the Portico home page. Logging in requires your *ISD username and password*, not your CS ones. Once logged in you will see the main index page listing the various facilities available to you.

If you are an undergraduate, or in some cases a postgraduate student as well, you will need to use Portico to select and register for your option modules (i.e., modules you are allowed to select from a range of choices). You will be given more detailed information on how this works during Registration week. You should also make yourself familiar with the Moodle on-line learning system (see http://moodle.ucl.ac.uk/), as this will be used by many lecturers to support their modules.

### 4.6 Your Own Home Page

If you browse around the local CS web you will find home pages set up by staff and other students. My home page, for example, can be found at:
http://www.cs.ucl.ac.uk/staff/G.Roberts
(it has a specially commissioned photo of me looking nearly normal!) Substitute the names of other staff members to see their pages.

You too can set up your own home page – find the page on the web that tells you how![7]

# Part III
# Email

## 5 Electronic Mail

Electronic mail (usually called email)[8] is a vital communication mechanism within UCL and especially in the CS department.

**It is absolutely essential that you read your email regularly since all important communication is sent this way. In particular, information about tutorials, timetable changes, examination registration, special events and requests for meetings are all sent by email. Since everyone is expected to read email regularly, few people look kindly on failure to read email when it is offered as an excuse for failure to know or do something.**

To use email you need to know:

- how to read your email,
- how to reply to or send new email,
- how to delete old email.

You will be using the UCL Live email service managed by ISD, see:
http://www.ucl.ac.uk/isd/students/mail/live.
It can be accessed via a web-based interface or email clients including Thunderbird, OS X mail and Outlook. Mobile devices (iOS and Android) are also supported via Exchange server access. You should attend an introductory session run by ISD to find out more about using the service.

[7] Hint: try the Helpdesk pages.
[8] Old fashioned letter post is called 'snail mail'.

## 5.1 The Email Message Header

Each email message has a header[9] which tells you:

- the name and email address of who sent you the message (the from: field),
- the email addresses of who else the message was sent to (the cc: field),
- when the message was sent (the date: field),
- what the message is about (the subject: field).

The body of the message, containing the message text follows the header.

Note that all messages include the name of the sender, so it is not possible to send anonymous email in normal usage.

## 5.2 Email Addresses

Every UCL student has an UCL email address. These have the form:
`firstname.familyname.year@ucl.ac.uk`
For example:
`Jane.Bloggs.12@ucl.ac.uk`
The year is the year you start at UCL and remains the same throughout your entire time at UCL. All UCL email addresses have to be unique, so if your firstname.familyname.year is already in use then your address will use a variation based on the same theme.

An email address consists of the email name on the left hand side, followed by an '@' (pronounced 'at') symbol and then the location address on the right hand side. The location is a sequence of domains (places) separated by dots. ucl.ac.uk means UCL within the Academic Community of the United Kingdom.

A number, but not all, of Computer Science staff use CS email addresses of the form `j.bloggs@cs.ucl.ac.uk`, rather than the UCL format for student email addresses. In addition, all staff also have UCL staff email addresses, which look like `firstname.familyname@ucl.ac.uk`. At the current time there is a transition under way whereby all UCL staff will move to using standard UCL email addresses, so you may see some changes during your time at UCL.

## 5.3 Virus Checking and Spam

Email is a notorious source of spam, viruses and dangerous attachments. All email messages will be scanned before you receive them to check for viruses or spam (junk mail). Problem emails will be marked with {SPAM?} or {VIRUS?} at the start of the subject line in the message header, and anything dangerous will have been removed. Such messages can then be deleted without having to read them if you make use of email filtering (filtering may on by default so you may not see spam email at all).

If you are emailing staff in the department make sure that you don't include anything that makes your email look like spam, otherwise it will not be read and will most likely be deleted automatically. If you email attachments like Word documents, use a virus checker to check them before you send them. Also when emailing staff please use your UCL email address so it is clear that the message is coming from someone within UCL. Some staff members will automatically delete email from sources like Hotmail or Yahoo and so will never see your message.

If you find that spam email is becoming a problem (for example, due to volume or offensive messages), visit the ISD Helpdesk and ask for assistance.

[9] Email headers can look quite confusing at first sight – this is because they are confusing...

**Neither the Department or UCL will tolerate the abuse of electronic mail. No abusive or obscene mail should be sent either within the department or outside of it. Never try to send email using someone else's account and remember all the email you send has your name attached to it.**

## 5.4 Programme Mailing Lists

A mailing list allows a message to be sent to a group of people without having to type in a long list of email addresses. The list has its own email address and holds a collection of email names and addresses of people who are members of the list. When a message is emailed to the list it is automatically forwarded to all the email addresses held in the list.

There are a number of mailing lists in use in the department, some of which you will be added to automatically. For example, first year undergraduate students will be added to the list `1styr@cs.ucl.ac.uk` and MSc Computer Science students to `msccs@cs.ucl.ac.uk`.

These lists are very useful for sending important messages to a large group of people. For example, if a first year lecture is cancelled a message can be emailed to '1styr and be received by all first year students.

In general you will only receive email from mailing lists and should not send email to them. Remember that a message sent to a list will be received by lots of people. If the message is not important or trivial you will quickly become very unpopular for sending junk email!

**Only send an email message to a mailing list if you are absolutely sure that everyone on the list needs to read it.**

## 5.5 Module Mailing Lists

Each Computer Science module you take will have a Moodle website. Moodle is the UCL online learning space, see `https://moodle.ucl.ac.uk/`. You will be making extensive use of Moodle as nearly all Computer Science modules make use of it. As Moodle is a UCL-wide resource it is not described in detail in this document[10].

Moodle provides its own user management and email list system, and messages about a specific module can be sent to you via Moodle. Such messages can include changes to lecture times or locations, and information about coursework deadlines. You must make sure that you read all the messages as they are taken as definitive statements on what is happening on a module.

When you first access the Moodle web page for a module you will automatically receive email messages sent to the module's News Forum, so you don't have to take any special action to get module emails. The messages are archived on the forum and can be read at any time, so if you lose or delete an email you can still go back to Moodle to read it again.

**It will be assumed you read all messages on all the mailing lists for all the modules you take. No excuses are accepted for not seeing and reading such messages!**

[10] Moodle is much easier to learn by using than by attempting to describe it here! However, there is extensive documentation available via the Moodle website if needed.

### 5.6 Using Email to Report Faults

The department maintains two email addresses that can be used to send an email reporting broken equipment, software problems and other issues, and for obtaining help from the Helpdesk. These are:

request@cs.ucl.ac.uk – this serves two purposes. The first is to report any broken computer equipment. Make sure you include details of the equipment, location and fault in the message. The second is to request help from the CS Helpdesk.

building-faults@cs.ucl.ac.uk – to report any problem with the fabric of the building, broken furniture, faulty lights or safety issues.

Please report any faults as soon as possible, as they will then be fixed sooner. When requesting help from the CS Helpdesk, bear in mind that only problems to do with the departments computing facilities can be dealt with.

# Part IV
# Using Unix

## 6 Getting Started with Unix

A computer essentially consists of a processor together with associated temporary data-storage (memory), a larger amount of more permanent storage (filestore), a network interface and some means of communicating with the user (keyboard, mouse or trackpad, and screen). The processor executes, or runs, programs that control what the computer does. The core program that manages the basic operation of the computer is called the operating system. It provides key services to the user, allowing the user to interact with the computer, to access equipment such as printers, and to run application programs (such as word-processors and so on).

Lab computers used in the department provide access to various versions of the Unix operating system, in particular one of the popular GNU/Linux variants and OS X on Apple Mac machines. Strictly GNU/Linux is not Unix but looks and behaves in a very similar way. We will not worry about the differences in this document and just refer to Unix. The main advantages offered by Unix are its flexibility, robustness, security, ease of management, support for experimentation and ability to support fully networked multi-user computing. As a Computer Science student it is important that you learn about and study Unix as it embodies many core concepts and ideas that you need to know about.

Unix has been around for over 40 years, so has a long history. Many different versions of Unix have been developed and it is supported on many kinds of computer. Many hand-held devices such as smartphones and tablets also run a version of Unix (a variant of the OS X version on iOS devices and GNU/Linux on Android). GNU/Linux is, in Unix terms, a more recent implementation of the operating system designed to run on typical PC hardware. Although we run a number of versions of Unix and GNU/Linux in the department, from the typical users point of view the differences between versions are relatively small, so that all of what follows will be applicable to all of our computers running Unix.

As the Unix operating system was originally designed by programmers for their own use, it has lots of nice features for the experienced user. Unfortunately, it can appear also cryptic and slightly intimidating to beginners; but Computer Science students are tough enough to cope!

Some lab computers run a version of Unix as their default operating system but Linux services can be also be accessed from computers running Microsoft Windows as their default operating system. On a Windows machine you will see an icon on the desktop or a start menu item used to gain access to Linux. Click the icon and a browser window will appear, allowing you to select which operating system you want to access, including Linux. Once an operating system is selected, its desktop will appear within the browser window and can be used in the normal way after logging in.

## 6.1 The Unix Desktop

To make Unix easier to use many versions make use of a windowing system, giving a user interface based on the now universal windowing metaphor made familiar by other operating systems such as OS X and MS Windows. Windows are displayed and managed by a Graphical User Interface (GUI) and desktop manager. Linux-based systems typically run the X Window System[11], often referred to as just X (the letter X not ten) or X11, although times are changing and replacements for X are appearing. OS X has its own GUI and desktop manager but can run X as well.

After you log in, there will be a short delay while the computer sets up the default screen layout and desktop. The appearance will depend on the window manager that is selected as default on the particular machine you have logged into. Whichever version it is, you will see a familiar desktop with windows, menus and so on. An example of a basic desktop is shown in Figure 1 but there are plenty of variations. Modern Unix desktops actually look and behave like much like those of Windows and other operating systems, so no surprises here. Dig below the surface, however, and you start to see the real power of Unix. Experiment and try things out!

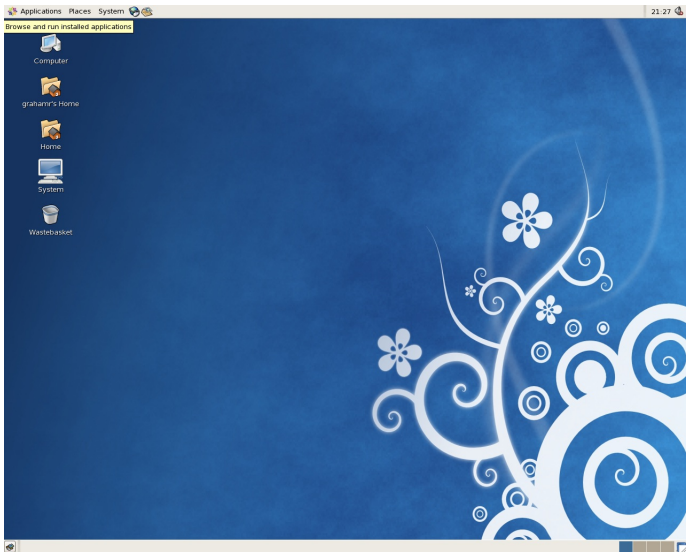


Figure 1: A typical Unix desktop

Figure 2 shows a Unix desktop with a few windows open. Each window you see on the screen effectively acts as an interface to a program running on the computer. The contents of the window and the way it is used depend on the program it is associated with. Notice that the desktop in Figure 2 includes two large windows displaying text. These are *terminal windows*[12]. and, as explained in the next section, they provide one of the most important ways of interacting with a Unix system.

## 6.2 Terminal Windows

One of the key differences when using a Unix-based computer is that a lot of interaction is achieved by typing commands using the keyboard rather than pointing and clicking with the mouse or trackpad. A terminal window displays a *command prompt* ready for the user to type in commands. Typically a prompt consists of the name of the machine you are using (e.g., khone) and possibly your user name.

When you wish to use a terminal window to enter Unix commands the window should be selected and the pointer placed somewhere within the window (it doesn't matter where). The keyboard can then be used to type characters,

[11] Unix gurus insist that X11 is referred to as the X Window System (no plurals, no hyphens). You can have fun annoying them by talking about X-Windows, Windows and other such combinations but make sure you have a good escape route.
While we are on the subject OS X is OS 'Ten' not OS 'X'. The 'X' is a coincidence not a reference to X11.

[12] Terminal windows are also often referred to as xterm windows, as in 'X terminal'. In fact, those shown in the example desktop are another variety of terminal window, not xterms. If you want to see an xterm window just enter the command xterm followed by ¡return¿ and one will appear

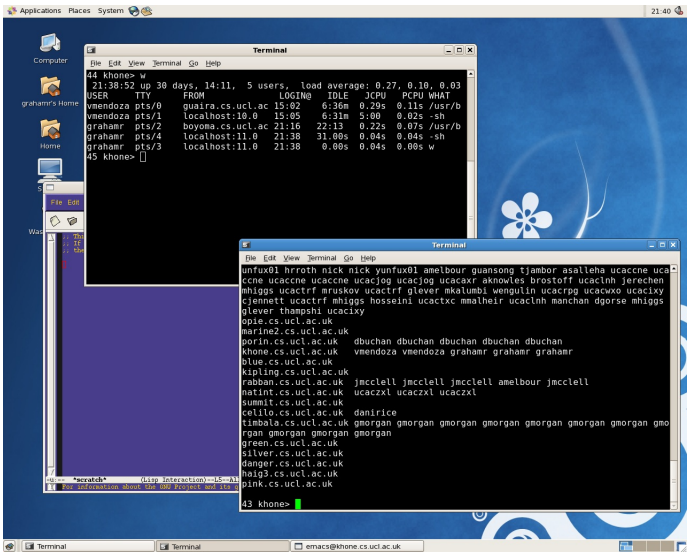

Figure 2: A Unix desktop with open terminal windows

which appear directly following the command prompt. A <**return**> key press will cause the characters typed to be acted on as a command to do something.

Although this guide will continue to refer to the use of terminal windows, they are actually containers for something called a *shell program* or the *Unix shell*, so you may hear these or similar phrases being used. The shell program really does all the work and is quite sophisticated, as well as being configurable in many ways. The default shell program varies on different machines but most use either one called 'csh (C shell) or more commonly another called 'bash' (Bourne Again SHell). You can actually start a new shell within an existing one by typing the command **csh** or **bash** into a terminal window.

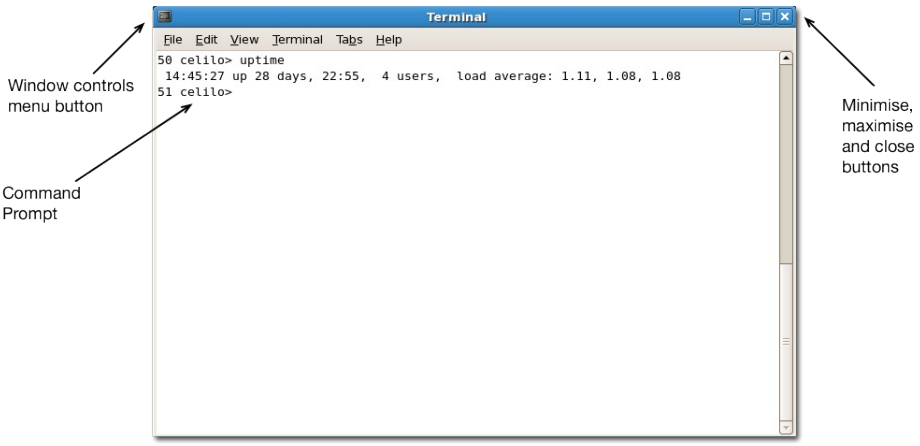


Figure 3: A terminal window

A closer view of a terminal window is shown in Figure 3. Terminal windows can be opened using either one of the pull down menus available from the menu bar or from the right mouse button pop-up menu or equivalent (the exact location and contents of menus will vary between different variants of Unix desktops). A terminal window can be closed permanently by typing the command **exit** at its command prompt and pressing <**return**>, or using the close button on the window itself.

Other programs can be run by typing their name at the command prompt in a terminal window (and not forgetting to press <**return**> of course). For example, the Firefox web browser can be started by typing **firefox&**<**return**>. You can also a menu to run Firefox but often just typing a command is quicker than finding and clicking a menu option.

Note that the ampersand (the '&' character) in the Firefox command name is not a typo and has a specific meaning, which is to run Firefox as a new process and allow the shell program to display the next command prompt. If you omit the ampersand the command prompt does not re-appear until you quit Firefox, and the terminal window remains unusable while Firefox is running; try it for yourself.

### 6.3 Configuration Options

When you have experimented with the desktop for a while and gained confidence with using the computer you might wish to explore what else can be done with it. It is possible to do a great deal to configure the desktop and window manager to your own preferences and there are many people in the department who will help you to find out more about how to do this. In addition, there are a number of other things that can be done using the default configuration, such as moving text from one window to another (copying and pasting). Experiment and swap ideas to find out about these facilities.

Many people are perfectly happy with the defaults and find that they can achieve everything that they want to without ever changing the original set up. Please remember, if you do decide to customise your account configuration, that the department will not accept anti-social use of the machines. In particular, make sure that your configuration leaves the machine as it was when you found it after you logout. If you change the background image please remember that certain images can cause offence, so act responsibly.

At some point you will discover that your account contains a collection of 'dot files' that hold the configuration information for your user account (these are files whose names start with a '.', such as **.uclcs-csh-options**). Do not change these unless directed to do so or you are confident you know what you are doing, as you are likely to discover that programs will apparently stop working, logging in or out fails, or the system becomes unpredictable. The Helpdesk staff will restore the default setup if you make mistakes but won't appreciate the time they waste doing it!

### 6.4 Changing Your Unix Password

Changing your password was discussed earlier but here is how to do it with Unix. Type the command **passwd**[13] and press the <**return**> key. You will then be prompted to type in your current password (to prove you are the correct user). You will find that your password is not displayed on the screen as you type, preventing anyone else reading it (including you, so type carefully). Next you are twice prompted to type in your new password, with the second time being a check against mistakes, as you won't see the characters you type on the screen. Press the <**return**> key after each password you type, in order to let the computer know you have entered the password and are ready to proceed.

# Part V
# Unix Fundamentals

## 7 The Filestore

The most important function Unix will perform for you is to manage your filestore. A filestore can be seen as a large filing cabinet in which all your electronic documents are stored. Each document is stored in a *file* which can be opened up to access the contents. As the computers in the department are all linked in a large network, you will have access to your files from any computer that you use.

Files are contained in *directories*, which are a bit like the drawers of a filing cabinet, but are rather more flexible. Directories can hold *sub-directories*, as if the drawers of a filing cabinet were like miniature filing cabinets in their own right, with new drawers inside them (the analogy gets a bit strained here but hopefully you get the idea!). Sub-directories, in turn, can hold their own sub-directories and this nesting can go to any depth required. The end result is

[13] The spelling may look strange but is correct! You will find that many commands have odd looking names - it is a 'feature' of Unix. Some names are so obscure that no one can fully agree what they originally stood for!

that directories and sub-directories form a hierarchy or tree structure, so you often hear phrases like the 'directory tree.

All files and directories have names so that you can identify and work with them. A name can be any sequence of characters found on the keyboard, excluding some of the punctuation and other symbols. It is best to stick with meaningful names that denote what a file or directory contains, rather than more cryptic names. There is no practical limit on the length of a name but all the names in a given directory must be distinct; having two files with the same name in the same directory isn't very useful...

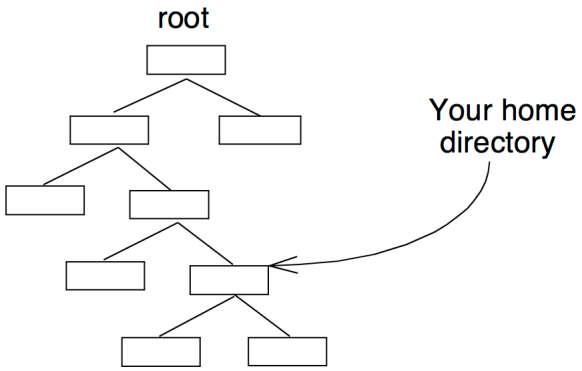


Figure 4: The Filestore Structure

A terminal window is always working from the context of one of your directories[14]. When you first start with a terminal window it will be situated at your top level directory, called your *home directory* (see Figure 4). Your home directory is itself kept in a directory holding all the other home directories for students in the same group as you. This directory in turn is kept inside a larger directory for all students and so on. Therefore, your home directory is already some way down the directory tree. At the top of everything is the *root directory*. Following from this you can see that the entire filestore for the whole department is organised as a single giant structure (an upside-down tree).

The following subsections introduce the core Unix commands that allow you to work with the files and directories in your filestore. Each command is typed in at a terminal window prompt. Don't forget that each command you type should be followed by a press of the Return key, shown by <**return**> in the text. If you don't press <**return**> nothing will happen!

When looking at Unix command descriptions, note that components given in angle-brackets (⟨**...**⟩) are arguments that should be replaced with the specific text you want to apply the command to, typically a file or directory name. Also make sure the pointer is pointing inside the window you want to use. If no text appears as you type (and the machine keeps beeping) the cursor is probably in the wrong place.

## 7.1 Listing File and Directories

To see the contents of a directory use the **ls** command which stands for list. If you type:
**ls** <**return**>
then the names of the files and sub-directories (if any) stored in the current directory for that terminal window will be listed. Notice that different terminal windows can have different current directories, but any changes you make to your filestore will affect the filestore for all the terminal windows, since they are all viewing the same single global filestore.

When using commands like **ls** you will find that Unix is not very verbose in its response to your commands, and will generally reply with the minimum amount of information. If no reply is needed at all, none will be displayed and you will simply see the command prompt immediately reappear.

The command:
**ls -l** <**return**>

[14] Strictly speaking the shell program running inside the terminal window is actually in control and determines the current directory. However, we will simply stick to referring to the terminal window.

uses the **-l** *flag* to tell the **ls** command to list the directory contents in long format. This means that information such as the size of a file, the date it was last changed or created and the file's owner are displayed. All files have an owner identified by their username, so all the files you create should have your username attached, as you will be the owner. In general you can only view or edit the files that are owned by you (file ownership will be explained in Section 7.10 page 24).

If you apply **ls** to a directory name, the files in the directory will be listed. For example:
**ls docs** <**return**>
will display the contents of the docs sub-directory (assuming it exists).

By default **ls** doesn't distinguish between files and directories when it lists them. If you use the '-p' flag then a '/' character will be added to the end of each directory name:
**ls -p** <**return**>

## 7.2 Changing Directory

To move between directories, you use the command **cd**, which stands for change directory. For example:
**cd coursework** <**return**>
would change to the directory called 'coursework', providing that directory exists and is a sub-directory of the current directory. **ls** will then list the contents of the 'coursework' sub-directory. If the sub-directory does not exist you will see a simple error message similar to this:
**cd: coursework: No such file or directory**
(The exact form will vary with different versions of Unix.)

If you type:
**cd** <**return**>
with no argument, you will be taken to your home directory, useful if you are lost in your sub-directories and just want to get back to home.

If you wish to move from a directory to its containing or parent directory type:
**cd ..** <**return**>
The dot-dot stands for 'the parent directory of the current directory'. Dot-dot is an example of the slightly cryptic notation used by Unix that is important to learn about to use Unix effectively.

While on the subject of dots, a single dot can be used as a shorthand for the current directory, so:
**ls .** <**return**>
is the same as **ls** on its own, and:
**cd .** <**return**>
means change to the current directory a null operation!

## 7.3 Creating Directories

As you start creating and using files, you will also need to create directories to organise your files, before your file-store becomes unmanageable (like a filing cabinet with all the files stuffed in one drawer). This is done using **mkdir**, which stands for make directory. You need to give this command the name of the directory you want to make, and the directory will be created as a sub-directory of the current directory (which might be a sub-directory of your home directory if you have moved down through your filestore). The **mkdir** command has this form:
**mkdir** <**new directory name**> <**return**>
For example:
**mkdir myDirectory** <**return**>
will create a sub-directory called 'myDirectory'.

Note that the notation <**new directory name**> means substitute a suitable directory name at this point – you don't type the angle brackets or the name inside them. As mentioned earlier, the descriptions of many commands will use this style to show the additional arguments a command requires. Arguments always appear on the same line as the

command name, so don't press <return> until the entire command line has been entered.

The new directory name may consist of letters and numbers, and possibly the underscore character but must start with a letter. While you can use spaces in a file or directory name, it makes life difficult, since, by default, UNIX treats spaces as breaks between parts of a command, and you have to enclose names with spaces in double quotes to avoid your commands being misinterpreted.

## 7.4 Directory Path Names

If you want to know which directory a terminal window is currently accessing, the Unix command **pwd**, short for print working directory, will tell you. The reply will be a sequence of directory names with slashes to separate them (a slash is the forward-slash character '/'[15]). This is how Unix refers to a long chain of directory names all the way from the filestore root. For example, the home directory of a user called 'fred' might be at:
**/cs/student/ug/violet/2012/fred**
This is called the *full path name* and you can use a similar path name to access any directory in the filestore that you have permission to see. A user's home directory name is always the same as that user's username.

Following the same idea you can also have partial path names that locate a directory relative to your current location in the filestore (as we have already seen with the **cd** command). For example, if you are in a sub-directory of your home directory called 'programs' and you wish to list the contents of a sub-directory called 'program_docs' inside a sub-directory 'documents' of your home-directory, the command:
**ls ../documents/program_docs** <**return**>
will achieve it, by telling Unix that the directory to list is to be found by going up one level, the intial '..', and then down into 'documents' and from there into 'program_docs'.

Full or relative pathnames for files and directories can be used anywhere a simple file or directory name is expected, as illustrated by the **ls** example above. This means that you don't have to change to a directory to affect the files it contains; pathnames can be used instead to reach the right location.

A path name displayed in response to **pwd** may include something like '/a/marine' at the front[16], for example:
**/a/marine/cs/student/ug/violet/2000/fred**
These additions are not a real part of the path name and can be ignored (you should never use them when typing a path yourself). For normal use, the path always starts with '/cs'. Most of the parts of a path name are obvious, but you will notice a few parts, such as 'violet' or 'orinda', which are the names of machines where parts of the filestore are kept[17].

A useful shorthand in specifying path names is the character '~' (called tilde) that can be used to specify the home directory of a user. For example, '~maria' is the pathname for the user with the username 'maria', so:
**cd ~maria** <**return**>
would, in principle, take you to maria's home directory. In practice, however, another user's home directory and its contents will be protected from access by other people, and you will not be able to change to that directory. This, of course, means that your home directory and its contents are protected in turn, so can only be accessed by yourself. Such security is an important feature of Unix and allows you to keep your files private.

Using '~' on its own refers to your home directory, so you can use it with **cd** to get to your home directory:
**cd ~**<**return**>
However, **cd** <**return**> does the same with less typing! More usefully, '~' allows you to refer to files or directories via your home directory, for example:
**cd ~documents** <**return**>
means change to the sub-directory 'documents' of your home directory, regardless of the current location.

[15] Don't confuse forward-slash '/' with back-slash '\'. Backslash is used for path names on non-standard operating systems like Microsoft Windows, but not on Unix systems.

[16] *'/a/marine'* specifies that an on-the-fly connection to part of the file store, using a *mount point*, has been made – the departmental filestore is so big that an individual machine only wants access to small parts of it at any one time. Hence, as you access a file or directory the relevant part of the file store is made available.

[17] These machines are called file servers and there are a large number of them. The whole filestore is not only big but distributed between the file servers/index file server.

**cd** / <**return**>
will change to the directory at the top of the filestore (the root directory).

## 7.5 Wildcards

Various useful short-hands or wildcards, such as '*', are available when referring to files or directories. '*' can be used to match any number of characters in a name, so:
**ls *prog*** <**return**>
will list all the files and directory names which have the sequence 'prog' somewhere in their name (in the current directory). The first star matches any characters before 'prog' and the second star any characters after. This means names such as 'aprog1', 'aprog2' and 'xyzprogqwe' will all be matched.
**ls prog*** <**return**>
will list files and directories that begin with the sequence 'prog'.
**ls *** <**return**>
will match all names in the current directory and gives the same results as just using **ls** on its own.
**ls -l *** <**return**>
will do the same in long format, illustrating that command flags can be included on any command line. Long format means that more information is printed about each file, including access permissions (see section 7.10), who owns the file, the date the file was last changed and other information. For example:

-rw-r–r– 1 jon staff 1383 2 Oct 2011 graphics.c
-rw-r–r– 1 jon staff 350 2 Oct 2011 graphics.h
-rw-r–r– 1 jon staff 370 1 Oct 2011 q1.c
-rw-r–r– 1 jon staff 717 2 Nov 2011 q10.c

Another wildcard is '?' which matches any *single* character. For example: **ls prog?** <**return**>
will match 'prog1', 'prog2', 'prog3', and so on.

## 7.6 Filestore Quotas

Each user is allocated a finite amount of filestore space, known as your *filestore quota*. When you log in you may see details of your current quota, including how much has been used, appear on the screen and also in the console window. It is important that you manage your filestore carefully to remain within your quota. If you use up your quota you will not be able to create or edit any more files. The command:
**quota** <**return**>
can be used show your current usage at any time.

The total amount of space you are currently using is found by adding up the space used by each file and directory. Filestore space is measured in terms of bytes, where one byte is equivalent to one character. File sizes are given in kilobytes or KBytes, where 1 kilobyte is 1024 bytes or characters. Use the command **ls -l** to list files and directories along with their sizes.

If you run out of filestore space and cannot work out what is using it up, then visit the Helpdesk and ask them to take a look. Web browsing, email and programming tend to create a lot of hidden files that use space without you really noticing.

## 7.7 Deleting Files and Directories

Deleting files and directories is an important activity if you want to manage your filestore and not exceed your quota. The **rm** command (short for remove) is used to delete files, while the **rmdir** command deletes directories. There is no undelete command for recovering deleted files, so once you delete a file or directory it is gone. Make sure you

use **rm** and **rmdir** carefully[18].

Note, however, if you have a disaster it is usually possible for the Helpdesk to recover deleted files from the daily backup provided they existed on the day *before* you deleted them – contact the Helpdesk to see what can be done. Remember, however, any files created or edited during the current day will be permanently lost.

The command:
**rm** <**file1**> <**return**>
removes (deletes) the file named at position <**file1**> from the current directory. No query is made as to whether you really want to delete the file, it just gets deleted. So, again, be careful how you use this command, it gives you no second chances!

Multiple files can be deleted at one go by listing several file names on the same command line:
**rm** <**file1**> <**file2**> <**file3**> <**return**>
will delete all the named files. Wildcards can also be used to delete multiple files using a single command, so:
**rm *.txt** <**return**>
will delete all files with names ending '.txt' in the current directory (sub-directories are not affected). When using a wildcard make sure that you really want to delete all the files that match, and you don't accidentally match files you wanted to keep. Again, no second chances!

Note that the command:
**rm *** <**return**>
can be very devastating, as '*' matches all file names in the current directory, and they will all be deleted. A common mistake is the intention to use a wildcard, for example:
**rm *.class** <**return**>
but accidentally type an extra space and end up with:
**rm * .class** <**return**>
The extra space (between the '*' and '.class') means that all files get matched and deleted.

The command:
**rmdir** <**directory_name**> <**return**>
will delete a directory if there are no files in it. To delete a directory and all the files it contains, first change to the directory, delete all the files in the directory using **rm**, change back to the parent directory and then use **rmdir**.

## 7.8 Other Useful Unix Commands

This section summarises some further Unix commands that you might expect to use as part of your everyday use of a Unix machine. There are also many other commands not covered here, the majority of which you will not need to make use of for some time, if at all[19]. Some people will be quite satisfied with the basic commands listed in this document but we would really encourage you to explore the command set and become as proficient as possible with using Unix.

- **date** – gives the current date and time.
- **w** – lists who is currently logged in to the computer you are using and also gives numbers indicating how busy the machine is.
- **whoami** – returns the username of the current user (i.e., your username).
- **cp** <**file1**> <**file2**> – copies <**file1**> to a new file called <**file2**> (remember, you replace <**file1**> and <**file2**> by the actual file names you are using). <**file1**> is not destroyed, but <**file2**> will be if it already exists (with no warning, so beware). The file names can be specified with a full or relative path name if the source or destination are not in the current directory. If a path is specified that gives a directory but not a filename as a

[18] Always think twice before pressing <**return**> when using **rm** – a good tip is to sit on your hands and carefully check what you have typed before pressing <**return**>!
[19] Asking more experienced users is a good way of finding out whether a specific task that you want to carry out can be done easily using unfamiliar Unix commands.

destination, the file is copied into the directory using the same name. For example:
**cp ~june/programs/example_program .** <**return**>
will copy the file 'example_program' from june's sub-directory 'programs' into the current directory, giving the copy the same name (the dot is shorthand for the current directory, where you were when you typed the command).

- **mv** <**file1**> <**file2**> – the move command changes the name of <**file1**> to <**file2**>. If a path name is given then the file is also moved (not copied) between directories. If the destination specifies a directory, but no name, then the file is moved with its current name. If <file2> already exists it will be replaced and the original will be lost.

- **more** <**file**> – allows you to display the contents of a text file in the terminal window. You can page through files using the space bar and return key. Although this command will attempt to display the content of any file it is only useful for files that actually contain plain text. If you try to display the contents of a file that doesn't contain text (and contains binary data) you will see lots of strange characters but no harm is done. Hit the 'q' key to return to the command prompt. Data represented as plain text is widely used in Unix, so the **more** command is more useful that you might think.

- **cat** <**file**> – the **cat** command is like **more** but displays the contents of a text file without pausing. It basically just copies the content of the file to the screen, so for a long file the content rapidly scrolls out of the window. **Cat** is actually a venerable Unix command that is very useful when you need it.

- **clear** – clears the terminal window, leaving only the prompt displayed at the top.

- **du** – this command will display details of your filestore usage by displaying a list of all your directories and the amount of space the files contained within them are using. It is useful for finding out where all your filestore space has gone to.

- **df** (or **df -k** on some machines) – this will tell you how much space is in use or still available on areas of the filestore as a whole that you have access to, rather than just your personal filestore. These areas are shared amongst many users, so don't assume all the spare space will go unused! Typing the command:
**df .**
will display the amount of space on the local filestore available to your particular user group.

- **find** – this command is used to search for files and directories. It has many flags and options, so we'll provide only a short introduction here. The structure of a find command can have up to three parts:
**find** <**where to search**> <**selection criteria**> <**action to take**>
Not all parts need to be used, so this use of **find** prints out the contents of the current directory and all sub-directories:
**find .**
Or you can specify any directory to start in:
**find ~/coursework**
This will list everything in the 'coursework' directory, plus sub-directories, in your home directory. Rather more useful is to search for a file with a specific name. This is done with the **-name** criteria flag:
**find -name ~/coursework ex1.c**
This will list any files with the name 'ex1.c' found in the 'coursework' directory. There are many more things you can do with **find** so check the documentation or search for a tutorial on the web. As one last example, here **find** is used to find all the files that have been changed in the last hour:
**find ~/coursework -mmin -60**

As well as commands like those just listed, it is also possible to type combinations of keys that have interesting effects. In particular, the technique of holding down the Control key[20] and typing another character is often used. For example:

- **Ctrl-C** – will immediately terminate any command running in a terminal window (Ctrl-C means hold down the control key and type the character 'c', in lower case). Once a command is terminated it cannot be restarted; you have to retype the command again. Typing Ctrl-C can be useful in an emergency if you type the wrong command. However, interrupting a command can leave things half done.

[20] The Control Key is usually marked as Ctrl on the keyboard.

- **Ctrl-Z** – will suspend a command running in a terminal window. A suspended command can be restarted later by typing the command:
  **fg** <**return**>

Beware of typing **Ctrl-S**. This will cause output to the terminal window to be suspended and so it looks as though it may have stopped working. Typing **Ctrl-Q** should get it going again. If a terminal window doesn't appear to be responding it is often worth remembering to try **Ctrl-Q**.

## 7.9 Useful Short-cuts

There are numerous useful short-cuts that can make using Unix commands quicker and easier. One of the most useful is what is known as *filename completion*. This means that you can type part of a filename or pathname, press the Tab key[21] and an attempt will be made to fill in the rest of the name automatically, saving you the effort of having to type it. For example, if you have a file called 'myFile' in your current directory, then you could type:
**ls my** <**Tab**>
Providing no other file name starts with 'my', the result of pressing Tab will be the characters 'File' appearing on the command line, to complete the name 'myFile'. If there are other names starting with 'my', then as many additional characters as possible are added provided they are common to all the names, leaving you to finish off by typing the rest of the characters for the name you actually want. This means that if another file called 'myFirstFile' exists then pressing Tab will result in 'myFi' but not a complete name, as 'myFi' is common to both names.

You can use Tab as many times as you like when trying to complete a name. So, in the case above, after you had got to 'myFi', you could type 'r' followed by Tab to proceed. This would result in 'stFile' being added to complete the file name 'myFirstFile'. If no completions are possible you hear a beep. Experiment with this to become familiar with it.

Another couple of useful short-cuts are associated with the **history** command. When you use a terminal window, the last 15-20 commands are remembered in what is known as the *history list*. Typing the command **history** will display the list and you will see a numbered list of the commands you have been using.

If you want to repeat the last command you typed (when you are programming this is something you quite often need to do), then typing:
**!!**<**return**>
will repeat the last command on the history list. This can save a lot of fiddly typing!

You can also use **!**<**number**> where <**number**> is the number of the command in the history list you want to perform again, so you are not limited to repeating just the last command used.

A more useful variation on this is **!**<**character(s)**> where <**character(s)**> are one or more characters forming the initial part of a previous command. When used, the first command matched by working up the history list will be performed again. For example, **!l** might match the last **ls** command you used. A final touch is that you can append additional arguments to a repeated command. For example, if you have used **!!** to repeat the last **ls**, then you could type **!! -l** to actually perform **ls -l**.

Usually the history list can also be accessed using the arrow keys. Pressing the up and down arrows will move up and down the history list. Also a command line can be edited by using the left and right arrow keys to move backwards and forwards along the command line. Characters can be added and deleted. This is useful if you have a long command line and make a typing mistake, as you don't have to retype the whole command.

Again, spend some time experimenting with these features to get used to them – they do save a lot of time.

## 7.10 File Access Permissions

Each file and directory has an owner and set of access permissions. You can access a file or directory (i.e., read or write to it) only if you are the owner or if the owner grants you the required access permissions. If you try to read or

[21] On some machines you may need to use the Escape key (Esc) rather than Tab.

write a file that you don't have access to, you will see an error message saying 'Access denied'.

Access permissions are split into three categories:

- permissions for the owner – 'u',
- permissions for people in the same group as the owner – 'g',
- permissions for everyone else (the others) – 'o'.

Users are put into groups (such as 'staff' and '1styr'), which identify the kind of user they are. You can use the **groups** command to find out your group(s). There are three principle kinds of permission in each category[22]:

- **read** permission, allowing the file to be read,
- **write** permission, allowing the file to be written to,
- **execute** permission, allowing the operating system to try and run the contents of the file as a command or program, or when dealing with a directory, allows the user to change to that directory.

If you use the **ls -l** command you can see the access permissions for the files and directories. For example:
-rw-rw-r– 1 derek 64 Aug 19 16:50 myfile
At the beginning of the line you can see '-rw-rw-r–'. The 'r' and 'w' characters indicate read and write permissions. Looking at the permissions more closely they can be broken down as shown in Figure 5.

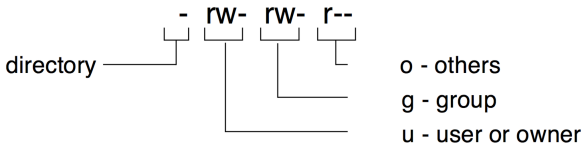


Figure 5: File and Directory Permissions

The first character is used to indicate a directory if the character 'd' is present, otherwise the entry is for a file. The next three groups of three characters are used to denote the read, write and execute permissions for the different categories of owner, group and others.

It is possible to set any combinations of permissions (including denying yourself read or write permission!). Permissions can be changed using the **chmod** command:
**chmod o+r** <**file or directory**> <**return**>
makes the selected file or directory readable by others but not writeable. The 'o+r' argument means 'others add read permission'. 'g+rw' would mean 'group add read and write permission'.
**chmod o=rw** <**file or directory**> <**return**>
using = simply sets the permissions exactly as specified, rather than adding them. The 'o=rw' means that others will be able to read and write the file or read or write to files in the directory (subject to the permissions on the individual files).
**chmod o-r** <**file or directory**> <**return**>
makes the file or directory unreadable to others (hence, the - character). chmod o+w <file or directory> <return> will allow other users to write to the file or directory.
**chmod u+x** <**file or directory**> <**return**>
adds execute permission to the file for the owner (you) or, for directories, allows access to the directory contents.

Check the online manual pages (see section 7.11) for all the other options.

If a directory is readable it means that its contents can be seen (e.g., listed by the **ls** command). If a directory is writeable it means that new files or directories can be created in the directory. It is a good idea to make sure that your home directory is only accessible (readable and writeable) by yourself. This can be done by changing to the directory above your home directory (use **cd ..** to get there from your home directory) and then doing **ls -l** to see all the directory permissions. Your home directory will have the same name as your login name and you should be able

[22] There are actually more than three but the others are less often used.

to see the access permissions at the start of the line. The permissions look like 'drwx–x–x'. Notice the 'd' for directory and the 'x' that a directory needs to allow access to it. Be very careful if you change these permissions, as you risk giving others access to your files.

### 7.11 Manual Pages

A manual page gives a detailed explanation of a command. You can use the **man** command to display a manual page in a terminal window and find out more about a command. All commands have many more options than are discussed in this guide, so it is always worth a look.

**man** <**commandname**> <**return**>
gives information about a particular Unix command (the manual page) if it is available. If you don't know the name of a command then try:
man -k <idea> <return>
'-k' is another example of a flag and in this case it modifies the behaviour of the man command. Many commands have flags that haven't been described in this document. They can be discovered by looking at the manual page for the command. In this case, the argument <idea> is any word that might be relevant to the behaviour of the command you want. For example:
**man -k print** <**return**>
would give a long list of commands relevant to printing.

The **man** command gives access to a large amount of information. Try using **man man** to find out more about the manual command itself.

It is worth the effort of learning how to use man and other tools, as most of the information you need about using Unix is available online. There are also many web-based tutorials and guides to Unix.

# Part VI
# Editing Text Files

## 8 Editors

One of the most important things you will use the computers for is to create and edit text files, in particular the source code for all the programs you will be writing. All this typing and editing means that you really need to be proficient at using both the keyboard and all the different editor programs you will come across.

When programming you will often be using a specialised editor embedded within an integrated programming environment (IDE) such as NetBeans or Eclipse. These will be introduced in the taught modules as needed, so won't be covered in this document. Instead, we will focus on more basic text editing, as it is an important activity when using Unix. You will find that a lot of the editing commands introduced in the following sub-sections also work on the command line when entering commands, and are also supported by editors in the IDE's.

### 8.1 Text Editors

A text editor allows you to enter text, rearrange and correct it, store it in a file and later retrieve it for further editing. One of the most widely-used general purpose editors on Unix systems is called Emacs, [23] and its basic operation will be described next.

[23] Emacs is really a Lisp programming system disguised as a Unix editor. It is very powerful and very much worth the effort to learn to use. It is a matter of great debate about what Emacs is an acronym for but some say it means: Emacs Makes All Computing Simple (this is a computer science joke – make sure you understand why before you graduate!).

There are other text editors available, including Vi[24], that may be familiar if you have used Unix before. If you are used to using editors or word-processors on PCs or similar machines you will find that Emacs looks very different and feels quite basic. But appearances can be deceptive. Emacs is a typically Unix tool, designed for experienced Unix users with a huge amount of power hidden below the surface.

## 8.2 Emacs

We recommend that you immediately start using Emacs as your default text editor. There are several distinct versions of Emacs available including GNU Emacs and XEmacs. It has also been ported to run as native OS X and Windows applications. Some versions of Emacs supports the use of the mouse or trackpad and include features such as a menu bar at the top of the window, scroll bars and cut/copy/paste. However, many expert users will control Emacs entirely via the keyboard as their preferred way of using it.

Emacs is definitely the 'real thing' – a very powerful editor capable of doing just about any editing job and a lot more besides. If it doesn't do what you want, you can modify or add to its code. If you like Emacs you will find that it provides facilities that are very useful for the programming modules you will be taking.

You can run Emacs by simply typing the command **emacs&**[25] at a terminal window prompt or by using pop-up a menu item. Emacs appears in its own window (see Figure 6), allowing you to carry on using the terminal window to type commands (the exact appearance of the Emacs window will differ depending on the precise operating system on the machine).

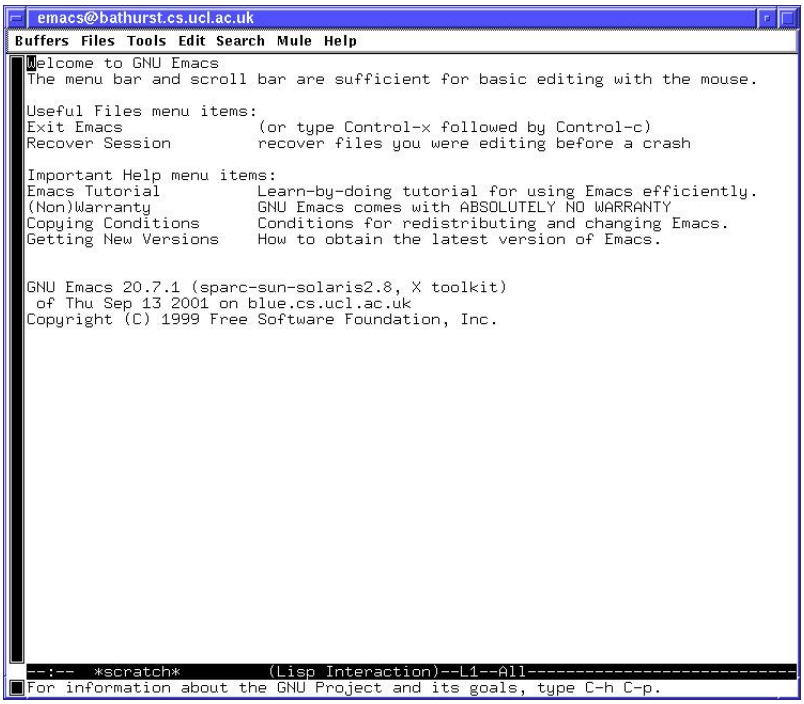


Figure 6: Emacs

You can find out a great deal about using Emacs by using the help menu (right end of menu bar) to access the help files and tutorials.

### 8.2.1 Basic Editing

To use Emacs you need to know about the following actions, and the commands used to make them happen:

[24] There is great rivalry between Emacs and Vi users. You are at UCL. You will use Emacs.

[25] The ampersand ('&') following the name emacs is not a mistake. It means 'start Emacs as a new task', allowing the terminal window to carry on working and accepting commands. Leave out the ampersand and the prompt will not reappear until you quit Emacs.

- opening a file,
- entering and deleting text,
- moving around a file,
- copying, cutting and pasting text,
- searching,
- saving files and exiting the editor.

All commands can be given by using various combinations of key strokes. The following sub-sections will focus on the use of key commands, as using the mouse or trackpad and menus with Emacs is straightforward and doesn't need detailed explanation. You may want to skip over this material for now, in favour of trying out Emacs and getting a feel for what it's like.

Commands given via the keyboard fall into two main groups, control commands and escape commands. To enter control commands you need to hold down the Control key while typing the appropriate command letter. These commands are written using the form:
**Ctrl-V**
which means hold down the Control key and type the 'v' character. Note the capital 'V' does not mean typing Shift-V (i.e., the upper case 'V')[26].

Escape commands require you to press the Escape key (Esc) before typing the command letter or letters. These commands are written using the form:
**Esc-V** which means press the Escape key, release it, and then press the 'v' key, using lower case.

A partially completed command can be cancelled at any time using:
**Ctrl-G**

You will probably find the command sequences rather odd at first, but you will quickly get used to them. If you think a better editor exists it is up to you to go and find it!

### 8.2.2 Opening a File for Editing

The best way to use an editor is to start it up and then load files for editing as you need them. Try to avoid starting a new editor every time you want to edit a file. Typing the command **Ctrl-X Ctrl-F** (hold down the Ctrl key and type 'x' then 'f' as separate characters) will prompt you to type the file name (and path, if needed) in the command line at the bottom of the editor window.

Name completion is supported by typing a space after you have typed the first few letters of the filename (not Tab as used in the terminal window). As long as those letters identify a unique file name, the name will be automatically completed, saving you the typing effort.

If the file you want to open already exists in the current directory, the text in it will be displayed. If the file doesn't yet exist, a new file is created and you start with an empty window.

When a file is open a status bar is displayed at the bottom of the window and a further line of text below it. The status bar displays information about the file, while the line below displays messages and acts as an input buffer for some commands that you type.

To enter text simply start typing. Text inserts at the text cursor position, which is the block shape that appears in the editor window (don't confuse this with the desktop cursor or pointer). If there is text in front of the text cursor on the same line, it will move along in front of the text cursor as new text is added. The backspace key will delete characters to the left of the text cursor.

[26] If a capital 'V' is required you would see Ctrl-SHIFT-V.

As you type in text you need to remember to press <**return**> at the end of each line. By default, when the end of a line is reached, new text simply appears on the next line, but you will see an extra line continuation character (a '\' or backslash character) at the end of the previous line. This will be treated as a special character and won't affect your editing, although you will find you cannot edit it yourself.

### 8.2.3 Moving the Text Cursor Around

The text cursor can be moved around the Emacs window, in order to position it so that text can be inserted, deleted, or corrected.

The most basic movements are one character up, down, left or right. These are achieved using:

- **Ctrl-B** – move one character back.
- **Ctrl-F** – move one character forward.
- **Ctrl-P** – move up to previous line, remaining in the same column if possible.
- **Ctrl-N** – move down to next line, remaining in the same column if possible.

On most machines it is possible to use the arrow keys to move around, as well. Emacs also lets you point and click with the mouse or trackpad and use the scroll bar. At times the movement of the cursor may seem strange; where the cursor cannot remain in the same column because the destination line is not long enough, it will move to the end of the line, but if you then move to another line, which is long enough, the cursor will move out to the correct column again.

Other useful text cursor movements are:

- **Ctrl-V** – move vertically down one page (equivalent to the number of lines the editor window can display).
- **Esc-V** – move vertically up one page.
- **Ctrl-A** – move to start of current line.
- **Ctrl-E** – move to end of current line.
- **Esc-**< – move to start of file.
- **Esc-**> – move to end of file (note that typing '<' and '>' require the shift key to be used).
- **Esc-]** – move forward one paragraph.
- **Esc-A** – move to start of sentence.
- **Esc-E** – move to end of sentence.
- **Ctrl-D** – delete the character in front of the text cursor.
- **Ctrl-K** – delete all text to the end of the line. This is the 'kill' rest of line command.
- **Esc-D** – delete the next word.
- **Esc-C** – delete the previous word.

It is also possible to jump to a particular line in a file, assuming each line is numbered sequentially starting from one: **Esc-x goto-line**, then enter the line number.
The current line number is also displayed at the bottom of the Emacs window. Note that a single line may wrap around and look like multiple lines in the Emacs window.

For a full list of similar key combinations (or *key bindings* as they are called) use the command:
**Esc-X describe-bindings**
This will display the list in the editor window.

### 8.2.4 Searching and Replacing Text

Two commands allow the text to be searched for words and patterns:

- **Ctrl-S** will search forwards for a text starting from the current cursor position. The text to search for is entered at the prompt that appears at the bottom of the window.
- **Ctrl-R** will search backwards in the same way.

It is also possible to replace one piece of text with another. To do this, type **Esc-X replace-regexp**, then enter the text to be replaced (which will appear in the input line at the bottom of the window), type <**return**>, then enter the replacement text and type <**return**> again. The replacements happen throughout the text after the text cursor position.

### 8.2.5 Killing and Yanking

Killing and yanking[27] allows text to be moved around in a file by deleting the text to be moved, moving the text cursor to the destination and then re-inserting the text.

To delete a large chunk of text, move the text cursor to the first character and use **Ctrl-K** to 'kill' the rest of the text on the same line. Repeated use of this command will delete a sequence of lines. These lines are stored in a temporary store called the 'kill buffer'. It is important to note that if you move to a different place and delete further lines, then these newly deleted lines will replace the old lines, rather than be appended to them. Only lines deleted in a single sequence of commands will be appended together in the kill buffer.

Once you have deleted the text you wish to move, move the text cursor to the destination and then insert it by typing **Ctrl-Y** to yank back the text. The same text can be yanked back as many times as desired and in as many places as desired, so this offers a good way to copy text.

### 8.2.6 Marking Regions

A region is a chunk of text with a beginning and an end. The beginning of a region can be marked by typing **Ctrl-Space** (that is hold down the Control key and press the space bar). The end of a region is marked by simply moving the text cursor onto the character following the last character.

Once marked a region can be deleted by typing **Ctrl-W** or copied by typing **Esc-W**. In both cases the text in the region is copied into the kill buffer and can be inserted at another location using **Ctrl-Y**.

### 8.2.7 Inserting Files

It is often useful to be able to insert the text from another file into the file that is currently being edited. This is done using the command **Ctrl-X Ctrl-I**, which then prompts for the name of the file to insert. The text from the file is inserted at the cursor position. Note, that the text is copied so the original file will remain unchanged.

### 8.2.8 Saving and Exiting

When the document you are editing is complete, or you are ready to finish your current editing session, you may want to quit Emacs (but remember you can also leave it running and just load a new file to edit). At this point it is important to realise that the work you have done is only stored in a temporary store and if you quit Emacs without making the storage permanent then the work is lost. This can be useful if you have made changes that you subsequently decide you don't want to keep, since exiting without saving the changes will leave you with the original contents of the file.

To save a file type the command **Ctrl-X Ctrl-S**. It is often good practice to save a file every minute or so, in case of problems (e.g., power failure).

[27] Unix terminology quite often relies on 'killing', 'yanking', 'aborting', 'terminating' and 'executing'. This is because Unix developers are strange people. Don't go into labs in the dark – if the grues don't get you the Unix programmers will.

To exit Emacs type **Ctrl-X Ctrl-C**. If you have not saved the document you are editing the editor will display a message asking you to confirm that you really want to leave without saving. Respond with 'y' or 'n' appropriately.

And don't forget, if you want to move on to edit another file you don't need to quit Emacs. Instead, just load in the new file as explained earlier using **CtrlX-Ctrl-F**. You don't have to quit Emacs until you finish using the computer.

#### 8.2.9 Buffers and Panes

Several files can be open for editing at any one time, by simply opening as many files as you want. Each file is placed in a separate *buffer*. A 'buffer' is the place where the file contents is stored in Emacs. There can be multiple buffers each holding the contents of a different file other content being used by Emacs.

By default only one file buffer is visible at one time in the Emacs window but you can switch between buffers using the command **Ctrl-X B** (type Ctrl-X followed by the letter 'b' without the control key held down). To see a list of buffers use **Ctrl-X Ctrl-B**.

Emacs also supports multiple *panes*, where the Emacs window is split into two or more sections, each displaying a different buffer. Hence, two of more files can be visible at once. The command **Ctrl-X 2**, will split a window pane into two, while **Ctrl-X 1** will remove all panes and display the normal window. The cursor can be moved between panes by clicking with the mouse or trackpad, or typing **Ctrl-X O**. Each pane effectively acts as an editor window in its own right, so you can load, save and do everything you would normally expect. As well as panes, Emacs allows you to open a complete new editor window. Explore the pull down menus to find the 'Make new frame' menu item and try it out.

It is well worth getting used to using buffers and all the other features of Emacs as they will save a lot of time and effort during your programming modules. Another tip worth remembering is that the desktop allows you to have any number of windows open at one time. This means you can dedicate one or more windows to an editor, while you type Unix commands into a separate terminal window. This is much more productive than starting and quitting your editor every time you need to edit a file. Get into the habit of using multiple windows.

# Part VII
# Printing

## 9 Printers

Having edited files and documents you may have a need to print them on paper (e.g., for handing in coursework). The department provides a general purpose laser printer service, available from any computer. The following subsections will give an overview of printing from Unix, see http://tsg.cs.ucl.ac.uk/basics/printing/ for further information and printing from Windows.

### 9.1 Printer Quotas

As the printers have limited capacity, you are given a quota for the number of pages you are allowed to print. A page is a printed side of a sheet of A4 paper. The quota varies according to your degree course and year of study but once you have used it up, you won't be able to use any printers in the department. However, it is possible to buy additional printer quota by going to the Reception Desk.

The command **printerquota** will tell you how much you have printed and what is left of your quota. First year undergraduates, for example, might see that they get a quota of something like 200 laser printer pages during terms 1 and 2 (quotas are renewed at the start of each term).

The standard quotas should be sufficient for any work you are required to do and additional quota is provided for activities such as printing project reports. Do remember to be economical with what you print (don't go mad!) and that printing is intended only for supporting the various modules you are taking.

## 9.2 Printing Files

The usual way to print a file is to use the **lpr** command which will send your file to your selected printer:
**lpr -P**<**printername**> <**filename**>
The **-P** flag is used to select the printer, so the command:
**lpr -Pps105** <**filename**>
will print your document on the printer named ps105, which is located in the 1st floor lab 1.05. The list of available printers is given in Section A.2 page 34 of the appendices or can be seen on the web at: http://tsg.cs.ucl.ac.uk/basics/printing/.

Most printers are mono laser printers that will print on both sides of the paper (duplex mode). There is also a colour laser printer on the 4th floor.

**Printing is an expensive facility to provide. Only print files that are required for the modules you are taking and the coursework you are asked to produce. Also keep the printer room tidy and always collect printouts promptly.**

If you try to print what are known as binary files (e.g., programs or specially formatted data files) the print job will probably be rejected, but may end up with the printer wasting lots of paper or becoming confused. When you collect printouts, do not press any buttons on the printer unless there are printed instructions on what to do and always try to leave the pile of printouts waiting to be collected in a tidy state.

## 9.3 Printer Queues

As the printers are shared by many people, each printer has a queue of 'print jobs' waiting to be printed. When you use the **lpr** command your print job is added to the queue.

There are a further two commands that it is useful to know about to use the printers effectively. The **lpq** command allows you to examine the queue of files waiting to be printed. It is useful to do this to confirm that your document really is waiting to be printed and to see how long you are likely to wait. Use this in preference to sending another copy of your document to the printer when you have had to wait a long time for anything to happen – it is very likely your job is waiting in a queue, which is either rather long or blocked waiting for a printer fault to be corrected (e.g., to load more paper)[28].

The command **lpq** lets you see the current queue on the default printer. To see the queue on any other printer use:
**lpq -P**<**printername**>
Each print job has an associated print number. If you want to stop a file being printed (it must belong to you) use the **lprm** command:
**lprm** <**job-number**>
or
**lprm -P**<**printername**> <**job-number**>

[28] If you find a printer has run out of paper then add more. Extra paper can usually be found near the printer. If there is no paper available or the printer is jammed, contact the Helpdesk and ask them to fix it. Do the same if the printed text is faint and the printer needs new toner. Don't attempt to fix printers yourself if they seem broken!

# 10 Summary

This guide has described the main set of tools which you will need to be familiar with in order to make use of the Unix computing facilities in the department. There might appear to be lots to learn but you will find that much of it can be treated as a reference at first, and that you will quickly remember the commands that you use frequently. For those who wish to explore the facilities further, there are various books, documents and information sources available – see the appendix.

Don't forget you can always ask your tutor for help, as well as your fellow students or the Helpdesk. Never be embarrassed or reluctant to ask for help, members of staff will always try to sort out any problems. Finally, become familiar with using the online help and manuals, and have fun!!.

If you have any suggestions for additions or improvements to this document[29], please email me at:
G.Roberts@cs.ucl.ac.uk
or access my home page at URL: http://www.cs.ucl.ac.uk/staff/G.Roberts/

This document has been formatted using LaTeX, a very powerful Unix document processing application, which is very different to the word processors you are used to.

Learn to use LaTeX(or else).

[29] Why does this document have so many footnotes? Answer: Historical inertia. Favourite question from previous years: What's a grue? Answer: that information cannot be divulged without a suitably large donation of zorkmids being made (real Unix users will understand what this means and if you don't there is a whole lot of computing history you need to catch up on).

# Part VIII
# Appendices

These appendices provides a bit more information about services in the department and the UCL Information Services (ISD) computing service.

## A Computer Labs in Computer Science

The following labs are available in Computer Science:
**1st floor**
Room 1.05: PCs running Windows
Room 1.21: PCs running Windows (Financial Computing lab)

**4th floor**
Room 4.06: Macs running OS X.

Note that all machines provide access to a variety of alternative operating systems via the virtual machine service.

All labs have a combination lock using the same combination - please keep the combination secret.

The Helpdesk is located in Room 4.22 on the 4th floor.

### A.1 Wireless Access

The CS Department provides a wireless access service within parts of the CS building, in particular within the computer labs. See http://tsg.cs.ucl.ac.uk/basics/connectivity/wireless_access/ for more information.

The UCL Information Services Division (ISD) provides a campus wide wireless access service called *eduroam*. This is accessible in most lecture theatres and class rooms, as well as many locations around UCL. See https://www.ucl.ac.uk/isd/students/wireless for information on how to access the service.

Note that any machine you connect to the network must be properly secure and have good virus checking software if you have to run Windows. If in any doubt talk to the Helpdesk.

P2P software, or any other software that causes unnecessary load on the wireless network, must not be used under any circumstances.

### A.2 Printers in Computer Science

Lab 1.05: Mono laser printer. Use name ps105 for double-sided printing, ps105-s for single-sided printing.
Lab 1.21: Mono laser printer. Use name ps121 for double-sided printing, ps121-s for single- sided printing.
Lab 4.05: Mono laser printer. Use name ps405 for double-sided printing, ps405-s for single-sided printing.
4th Floor: Colour laser printer. Use name cps405.

Remember that you have a printer quota that limits the number of pages you can print.

See http://tsg.cs.ucl.ac.uk/basics/printing/printers_in_the_department/ for more information.

### A.3 Opening Times

The labs are always open during normal weekday working hours: 8am to 7pm. At certain times during the year, and for certain groups of students, there may be extended opening hours. Watch out for announcements via email. There

is no weekend access.

# B ISD PC services

ISD stands for 'Information Services Division'. It is responsible for providing computing services to the UCL as a whole (Computer Science provides its own separate service to students in this department). As well as core services like email, ISD provide an extensive range of other computing services to students that you are free to make use of, including printing services.

Around the UCL campus you will find a number of computer workrooms, also known as cluster rooms, where computers are located. These are largely running Windows. Information about ISD, including the equipment available and opening times, can be obtained from the ISD Service Desk in the DMS Watson Science Library (next door to the MPEB), or via the ISD web pages at http://www.ucl.ac.uk/isd. The closest cluster rooms are also located in the Science library. See http://www.ucl.ac.uk/isd/students/workrooms information about the location of all the workrooms and their opening hours.

Before you start using ISD computers you will need to have been registered for an account. This should have happened during your normal UCL registration process but if you miss out go to the ISD Service Desk in the DMS Watson Science Library.

You will find that ISD operates a booking system and you will need to book a computer before you can use it. ISD also has a printing service with a print quota. Additional quota can be purchased, see http://www.ucl.ac.uk/isd/students/workrooms/printing.

If you have problems using ISD machines then contact the ISD Service Desk. The Computer Science Helpdesk will NOT be able to deal with these problems.

# C Using your own computer for coursework

It is possible to use your own Mac or PC, if you have one, to do at least some of your coursework, particularly for programming modules. Lecturers will tell you what is possible for specific modules. Note, however, that the responsibility for maintaining your own machine and making backups of your important data is all yours.

It is very strongly recommended that you copy your important files onto your Unix filestore in the department, as this filestore has very reliable backup procedures. Failure of your machine leading to loss of data is not accepted as an excuse for not completing work on time.

## C.1 Mac OS X

If you have a Mac running OS X Snow Leopard or Lion, then it is already running a version of Unix. The X Window system (XQuartz) is available as an optional install. Many members of staff use Macs, so they are very suitable for use on your modules.

## C.2 GNU/Linux – Unix on your PC

For those with their own PC's it is possible to run a version of GNU/Linux. This will convert your Windows PC into a system very much like the Unix computers in the CS department. GNU/Linux is available in a wide variety of distributions, such as Ubuntu. Ubuntu is popular and easy to install. It, and other varieties of GNU/Linux, have a number of options including the ability to simply run it as another program on a Windows machine, so you don't have to worry about damaging your existing version of Windows.

GNU/Linux also has the major advantage of low cost – you can download it for free. A typical installation comes with many of the tools that you will be using in the CS department and provides an excellent way of learning about Unix and becoming proficient in using it. Note, however, installation of GNU/Linux is entirely at your own risk and the department will not be able to help if you run into trouble.

As an alternative to running GNU/Linux, you might also look at Cygwin (http://www.cygwin.com) with Microsoft Windows. This gives you a basic Unix-like environment without having to install a new operating system.

## C.3 Remote Access to CS Computers

As all machines in the department are networked, it is possible to remotely login to a CS department machine from ISD machines, from halls or via commercial networks and ISPs from home (this can be useful is CS is full-up or closed).

Remote login is allowed to a limited number of gateway machines in the department. The list can be obtained from the Helpdesk or the departmental web site (see http://tsg.cs.ucl.ac.uk/basics/connectivity/remote_access/). To remote login you must use **ssh**, the secure shell protocol.

CS services can also be accessed using the web-based interface to the Remote Worker virtual computing service. See http://tsg.cs.ucl.ac.uk/basics/connectivity/ for further information.

# Index

# Practice Worksheet

This worksheet gives you a list of things to try out when you first start using the department's computers and, in particular, Unix. Make sure you are able to do all the basic operations, including logging in and out, using the keyboard and mouse or trackpad, using windows and menus, using files, using emacs, using Unix commands, sending and reading email and accessing the Web.

It is most important that you are able to use email properly, so make sure you get some practice now. It is also a good idea to practice your typing skills. The better you can type, the easier it will be to get things done.

If you have problems or get stuck then don't hesitate to ask for help – there should be a helper present at the lab sessions (they may be in one of several labs) or visit the Helpdesk on the 4th floor.

Have fun!

***** When you have had enough practice please do the following *****
**1. Send me an email to let me know that you are here and that you are able to use email. In the message put your name and what course you are on. Feel free to ask questions if you have problems or queries about Unix. My full email address is: G.Roberts@cs.ucl.ac.uk or graham.roberts@ucl.ac.uk**
**2. Send your tutor an email message to tell them you are here. Your tutor's email address will be like mine above but using their initial and surname. (If I am your tutor send me both messages!)**

Don't forget that the introductory document describes many of these operations in more detail. Check it out if you get stuck or ask a helper.

**Logging in**
Start by logging in – you should have got your Computer Science username and password when you registered for your computer account in the department.

**The Web**
Start up a web browser, such as Firefox, and look around the department's and UCL's web pages. The department's home page is at www.cs.ucl.ac.uk. Find the the student pages as these include lots of useful stuff, such as timetables and full details of each module.

My url is: http://www.cs.ucl.ac.uk/staff/G.Roberts - this will display my home page. All members of staff have a similar url pointing to their home page if you substitute their initial and surname. You too can have a url and your own home page. But I'm not going to tell you how to set it up!! (Im sure that other students will let you know :-) Hint: try searching.

**Email**
It is important that you are able to read and send email, so make sure the UCL Live email system is working for you. You access your email account via your web browser following the instructions you got from ISD (or see http://www.ucl.ac.uk/isd/students/mail/live). Don't forget that you use your UCL ISD computer account (name and password) to access email.

Send some email messages - start by sending messages to yourself (don't worry it's not as bad as it sounds!!). Old email messages you no longer need can be deleted.

**Unix**
Practice using Unix. If using a Windows machine you get access to Unix via clicking a desktop Icon, via the Start menu or use this URL in a web browser http://tsg.cs.ucl.ac.uk/basics/connectivity/cs_remote_worker/ to access the Remote Worker service. You can actually use the Remote Worker service from any up-to-date web browser from any machine with Internet access including your own machine from outside of UCL.

When Unix is running get used to using the desktop, moving the windows around and using the pop-up menus. Use a terminal window to run some basic Unix commands like **ls** and **pwd**. Don't forget to press <**return**> (that means the return or enter key on the keyboard) so that Unix will try to interpret what you typed as a command.

Take note of the 'helpful' error messages when you get a command wrong. Deliberately make some mistakes to see what you get!

**ls** will list any files in your current directory – it may be empty when you first start.

**pwd** will tell you the pathname of your current directory.

You can use the Unix command **passwd** <**return**> to change your Unix password. Choose your new password carefully – use a minimum of 6-8 letters, digits and other characters, and avoid common words and names that others might guess. Don't tell anyone else what your new password is. Note that this will not change your Windows password on Computer Science machines or your ISD password.

Dont forget your new password!! If you do, contact the Helpdesk.

Using Unix you can start Firefox by typing the command **firefox&**<**return**> in a terminal window, or find the firefox item in a pop-up menu.

**Editing with Unix**
Try using the Emacs editor. Emacs is started by typing the command **emacs&** <**return**> in a terminal window or using the root menu. Also, if it is installed, try xemacs using **xemacs&** <**return**>.

Using the notes in the introductory document, experiment with Emacs and get used to using the commands and editing text. It is important to become familiar with the various menus and command key combinations (which will seem strange at first). You might want to start by typing in part of the first paragraph of this sheet to get used to typing and editing. Here is a summary of the more useful keystrokes for moving the cursor and editing text - try them all:

- **Ctrl-B** – move one character back.
- **Ctrl-F** – move one character forward.
- **Ctrl-P** – move up to previous line, remaining in the same column if possible.
- **Ctrl-N** – move down to next line, remaining in the same column if possible.
- **Ctrl-V** – move vertically down one page (equivalent to the number of lines the editor window can display).
- **Esc-V** – move vertically up one page.
- **Ctrl-A** – move to start of current line.
- **Ctrl-E** – move to end of current line.
- **Esc-**< – move to start of file.
- **Esc-**> – move to end of file (note that typing '<' and '>' require the shift key to be used).
- **Esc-]** – move forward one paragraph.
- **Esc-A** – move to start of sentence.
- **Esc-E** – move to end of sentence.
- **Ctrl-D** – delete the character in front of the text cursor.
- **Ctrl-K** – delete all text to the end of the line. This is the 'kill' rest of line command.
- **Esc-D** – delete the next word.
- **Esc-C** – delete the previous word.

Save what you have typed to a file using the **Ctrl-X Ctrl-S** command. Notice that when you try to save a new file, Emacs will ask you to give it a name. Type in a file name and press <**return**>.

You can also start Emacs by typing **emacs** <**filename**>**&**, if you want to edit a file that already exists. Use **Ctrl-X Ctrl-F** to load a new file into Emacs.

**Using files and directories**
Use Emacs to create three or four small text files and then move to a terminal window and try using some Unix commands such as **ls -l**, **cp**, **mv**, **rm** and more (see the Introduction to Unix document for details).

There is online help available to tell you more about the various Unix commands. Try a manual look-up such as:
**man ls** or **man rm**
to learn more about a command. If it is installed a more comfortable way to read the manual pages is to use the program **xman** (type **xman** <**return**> in a terminal window). Or simply do a web search.

You may have created all your files so far in your home directory. Create some new directories using the **mkdir** command and use **cd** to change directories. Copy files into the new directories.

Experiment more with **cd** to navigate around the directory tree. After you change to a new directory use **ls** to see what files and directories it contains.
**cd ..** – will move you to the next directory up in the tree.
**cd** / – will move to the root of the tree.
**cd** – will move to your home directory (if you get lost, just use cd to get home).

**Where next?**
Over the next few weeks you will be starting various Computer Science taught modules, in particular programming modules. You will need to use the computers on a daily basis, so keep practising the basic skills of typing, using Unix and using email.

It is a good idea to read through a Unix text book or find a good tutorial on the Web.

The best thing you can do is to keep exploring and trying things out – that's the whole point really!!

For a challenge see if you can actually set up your own Web home page on the Computer Science web site – email me if you succeed. This will be a good way to learn quite a bit about Unix and the Web formatting language called HTML.

Don't forget to check for new email messages at least once a day.

**REMEMBER TO LOGOUT when you are finished.**