Chapter 3

A bit of C

3.1 File I/O in C

Consider inputting a large amount of data to a program. Eg find the average of 1000 numbers.
We store the numbers in a separate file and input the file name.
C can open/close, read/write to files.

```
#include<stdio.h>

main(void)
{
FILE *ifp;

ifp=fopen("my_file","r");

}
```

FILE *ifp;	declares ifp (infile pointer) to be a pointer to FILE FILE is defined in stdio.h and contains info. about the state of the current file ie is it open, closed?...
fopen	is a function, it takes 2 arguments: -filename -mode in which file should be opened fopen returns a pointer to FILE

file modes are:	”r” for read
	”w” for write
	”a” for append

So in our example the file was opened for reading ie getting info from the file.

3.1.1 Read/write to files

once a file is opened use:
fprintf() to write
fscanf() to read
ie file versions of the functions:
printf()
scanf()

again fprintf() and fscanf() require #include <stdio.h >
they work just like printf and scanf with 1 extra bit of information; the file-name.
Eg.
fprintf(ifp,"this file contains %d numbers ",x);
or
fscanf(ifp,"%d",&a);
When,
a file opened for writing and doesn't already exist – it is created.
opened for writing and exists – writing starts at the beginning of the file ie you lose what is written previously.
We use "a" append to write to a file and keep existing data.
How do you know when you reach the end of a file?
EOF, end of file, is a symbolic constant defined in stdio.h in strings 'ø'
We use the command fclose(ifp);