City University
School of Engineering and Mathematical Sciences
Common part 1

COMPUTING

Fortran Lecture 1.1

1. Fortran Language

Fortran is a high level computer language first invented in the 1950's. Fortran IV, invented 1970, Fortran 77 used until Fortran 90/95. The latest version we have is SALFORD F90/95. Some features of Fortran were needed for older machines without VDUs, keyboards or mice. If you read old text books for Fortran 77, not all of the features will work with Fortran 90/95. Nowadays Fortran is very powerful when linked with Windows. You can write very sophisticated programs, including computer games.

The course is an introduction to the language. We will not be using much graphics. All input and output is TEXT, i.e. keyboard and screen.

2 Structure

All Fortran programs need three different kinds of files.

1) Source File. We can read these files with a text editor, such as NotePad. All our source files have extension .F90 or .F95.
2) Object File
   The compiler will translate our source file into Binary machine code which our particular computer can understand. If we compile the source file, prog1.f95, the compiler will generate a file, prog1.obj. We may get some error messages from the compiler, if so the Fortran language errors must be corrected.
3) Executable file.
   The last stage in program preparation is to link or build the object file with the dynamic link libraries (*.DLL files) for our particular machine. We will use Windows XP. The Builder will produce an executable (*.EXE) file.

3 Getting Started

To use Salford Fortran properly, one of the .DLL files must be in the same directory as the .EXE files. Since you should store your source files on your personal space on the U: drive, the linker will automatically generate *.OBJ and *.EXE files in your own U: drive disk space. The first time you use this machine, you should setup the directory \Fortran in your personal disk space on the U: drive, then use MYCOMPUTER icon twice to copy

C: \Program Files\win32app\salford\salfibc.dll
Into U:\profile\personal\fortran\salfibc.dll

Please ask your tutor if you don't know how to do this.

4 Fortran Help

From the desktop double click the SALFORD PLATO IDE icon. IDE stands for Integrated Development Environment. You should have a screen called PLATO2. Select File, New, then type the three lines

```
Program Prog1
Print *,'Welcome to FTN95'
End Prog1
```

You should save the program source file in your own disk space on U:. To do this, select File, Save AS, then U:\Profile\Personal\Fortran.
You should now have a new program called prog1.f95 saved on your own personal space. Return to the PLATO2 window, select Project, then Compile; Project, then Build; Project, then Run. Your file Prog1.exe will then start executing. Prog1.exe should create a new window; print out the message "Welcome to FTN95".

Lecture 1.2

1. Variable types

All the data in a computer memory is stored in 8 bit BYTES. This means that there is a limit to the biggest and smallest numbers the computer can store in one byte. For larger and smaller numbers more than one byte must be used to store each number. Also the computer will store Characters, i.e. printable letters in bytes. We can define how many bytes the computer uses to store each number. In Fortran (and most other languages) we can have names for each variable and we can define how many bytes of memory each variable occupies. Types of data are:

CHAR *1	This is one character stored in one byte
INTEGER	This may be either two or four bytes of memory
REAL	Usually four bytes
COMPLEX	usually two INTEGERS (can be different)

INTEGERS are whole numbers, either positive or negative. Integer arithmetic is exact , with no errors. REAL numbers have a decimal place. REAL arithmetic is subject to rounding errors i.e. 10.0/3.0 = 3.3333…. .This is not exact.

We can have more than one character stored as a string. The string will need one byte of memory storage for each character, including spaces and punctuation marks.

Converting INTEGES to REAL and back again must be done by the programmer. Also we can't do arithmetic with CHAR variables (why not?).

2.Questions

Identify the following constants as REAL, INTEGER,CHAR or COMPLEX

3.14159
356
-2
qwerty
3+4i
L890
8L90
+3459.0
13.0 E+9
-4.67E+6
[,,kk]

3.Fortran variable names

Normally names beginning with I,J,K,L,M,N are used to store INTEGER numbers and all the other legal variable names are used for REAL numbers. When we need COMPLEX or characters we must declare them with CHAR or COMPLEX. We can also use INTEGER and REAL.

4.Fortran Input and Output

We must INPUT data into the computers memory, perform some calculations, the OUTPUT the answers. Fortran does this by defining the input and output devices (e.g. keyboard, printer,vdu screen) and also the FORMAT of the data. In PROG1 we used PRINT *, 'welcome to FTN95' . The PRINT is a fortran language word, the * means the vdu screen and 'Welcome to FTN95' is a CHARACTER string of 16 bytes.

Sometimes we want to input data from , say a disk file, or the keyboard. We may want to write data to a disk file. Fortran uses READ and WRITE statements with a FORMAT to do this Type in Prog2 and Prog3 and see what they do. Ask your tutor to explain the format.

```
 PROGRAM Prog2
  INTEGER i,j
  READ *,i
  j=2*i+3
  PRINT *,j
END PROGRAM Prog2
```

The next example is similar but uses REAL numbers:

```
PROGRAM Prog3
  REAL x
  WRITE(*,"(a)",ADVANCE="NO") "Enter a real number:"
  READ *,x
  PRINT *, "The result is:", x/3.0-0.9E6
END PROGRAM Prog3
```

5. Mixing INTEGERS and REAL arithmetic

```
      PROGRAM Prog4
        PRINT "(a,i2,4x,a,f6.2,4x,a,e10.3)", &
             "Integer:",77,"Float",77.0,"Exponent",77.0
    END  Prog4

      PROGRAM Prog5
      L=7
      K=2
      M=L/K
      PRINT *,"Integer division",L,"divided by",K,"equals",M

      X=7.0
      Y=2.0
      Z=X/Y
      PRINT *,"Real Arithmetic division",X,"divided by",Y,"equals",Z

END PROG5
```