CY40014 Introduction to Computational Chemistry
Autumn 2010-2011

Module 1: Introduction to FORTRAN programming

Worksheet 2

Elementary FORTRAN: Part II

Program layout

```
Declaration statements
Executable statements
End
```

Example

```
implicit real*4 (a-h, o-z)

read(*,*)x
y=x+1.0
write(*,*)y
stop

end
```

Note: Blanks are generally not significant and are used to improve the readability of different segments of the program.

Example of other declaration statements:

```
implicit none
integer*4 x
character*3 amino
real*8 imap
```

Aim of the present worksheet

1. To read **data from a file**
2. Introduction to **arrays**
3. To use **do loops**
4. To use the **if statement** with IF-THEN-ELSE-ENDIF

Important instruction

You are expected to complete as many worksheet problems as you can within the stipulated class hours. The rest can be solved outside the class hour. The completed solutions to Worksheet 2 are due in TWO weeks.

Reading/Writing data from/to a file

W2_1. Write a simple FORTRAN program to read the values of the real variables *x*,*y* and *z* from standard input and write these values to standard output.

W2_2. Write a simple FORTRAN program to read the values of the real variables *x*,*y* and *z* from a file named '*fort.12*' and write these values to standard output. Your program may look like the code given here.

```
implicit real*4 (a-h, o-z)
read(12,*)x,y,z
write(*,*)x,y,z
stop
end
```

W2_3. Write a simple FORTRAN program that

(i) reads the values of the real variables *x*,*y* and *z* from a file named '*data.in*' and

(ii) write these values to standard output. Your program may look like the code given here.

```
implicit real*4 (a-h, o-z)

open (unit=12, file='data.in', status='old')
read(12,*)x,y,z
close(12)

write(*,*)x,y,z

stop
end
```

W2_4. Write a simple FORTRAN program that

(i) reads the values of *x*,*y* and *z* from a file named '*data.in*' and

(ii) writes these values to a file named '*result.out*'.

You may follow the code given here.

```
implicit real*4 (a-h, o-z)

open (unit=12, file='data.in', status='old')
read(12,*)x,y,z
close(12)

open (unit=14, file='result.out', status='unknown')
write(14,*)x,y,z
close(14)

stop
end
```

W2_5. Write a simple FORTRAN program that

(iii) reads the values of *x*,*y* and *z* from a file named '*data.in*' and

(iv) writes these values to a file named '*fort.10*'.

Note: implications various arguments specified in the **open** statement -

unit a reference number that is used in the code to pinpoint the file

file name given to the file

status old: code looks for a file that is already present in the working directory

new: code opens a new file

unknown: opens the file if it already exists, otherwise creates a new one

Use of do-loops

W2_6. Consider a real variable *a*. Write a program that tells you the results of adding the numbers 1.0, 2.0, 3.0, …,10.0 to this variable *a*.

Program

```
implicit real*4 (a-h, o-z)
read(*,*)a
do i=1,10
    x=a+float(i)
    write(*,*)i,x
end do
stop
end
```

W2_7. Consider a real variable *a*. Write a program that tells you the results of multiplying this variable by 5.0, 10.0, 15.0, ……, 50.0.

[Hint: At the *i*-th step, you need to compute $x = a + (5.0 * i)$]

W2_8. Evaluate the factorial of an integer *n*.

[Hint: factorial = $\prod_{i=1}^{n} i$]

W2_9. Suppose you have a data file named '*data.in*' that contains 10 values of the variable *x*. Write a program as shown here to

(i) read these values from '*data.in*' and
(ii) write these values to the standard output.

Note: Here, x is not a simple real variable. It has been declared as an '**array variable**'. As shown in the program, x can take up to *100* values in the present problem.

```
implicit real*4 (a-h, o-z)
dimension x(100)

open (unit=12, file='data.in', status='old')
read(12,*)n
do i=1,n
   read(12,*)x(i)
end do
close(12)

do i=1,n
   write(*,*)x(i)
end do

stop
end
```

W2_10. Now write a program that reads in the *10* values of *x* from '*data.in*' and calculates the (i) sum and (ii) product of these numbers. The new program segments that you would now need are as follows:

(i) to calculate the sum $\sum_{i=1}^{n} x_i$

```
sum=0.0
do i=1,n
   sum=sum+x(i)
end do
```

(ii) to calculate the product $\prod_{i=1}^{n} x_i$

```
prd=1.0
do i=1,n
   prd=prd*x(i)
end do
```

Use of if construct

W2_11. This is an exercise where you may determine a condition using the '*if-then-else-endif*' construct. Write a program as shown and decide if the input value of the variable x is positive or negative.

```
read(*,*)x
if(x.lt.0.0) then
  write(*,*)' x is negative'
else
   write(*,*)'x is positive'
endif
```

W2_12. Write a program that

(i) reads in the values of two real numbers *x* and *y* from standard input,
(ii) writes to the standard output the larger of the two numbers.

W2_13. Write a program that

(i) reads in the values of two real numbers *x* and *y* from standard input,
(ii) calculates the sum and product of these numbers and
(iii) writes to the standard output the sum OR the product whichever is smaller.

W2_14. From a given set of numbers $x_1, x_2, x_3, ..., x_N$, find the (i) greatest and (ii) smallest numbers.

Parts of two sample programs are given here. The algorithm for the problem (i) is as follows. You start by assuming that the greatest number, represented as *xmax*, is a very small number. Here it has been assumed to be equal to -10^{-6}. Then you have to examine one by one each the numbers given to you and replace the value of *xmax* by the number that turns out to be larger than *xmax* .

```
xmax= - 1.e+6

do i=1,n
   if(x(i).gt.xmax) then
     xmax = x(i)
   endif
 end do

write(*,*)xmax
```

Hint: Write the numbers $x_1, x_2, x_3, ..., x_N$ in a file named 'xdata.in' and read them from there.

```
xmin=1.e+6

do i=1,n
   if(x(i).lt.xmin) then
     xmin = x(i)
   endif
 end do

write(*,*)xmin
```

Note: 1. While using the 'if' construct, you may use the following:

lt : $<$ *gt* : $>$ *eq* : $=$ *ne* : $\neq$

ge : $\geq$ *le* : $\leq$

2. The structure of an *if construct* usually consists of *if-then-else-endif* as shown. But simplified versions without the *else* option can also be used as and when necessary.

Assignment 2

A2_1. Given a set of real numbers $x_1, x_2, x_3, ..., x_N$. Find (a) the average and (b) the standard deviation of these numbers.

A2_2. Evaluate $\exp(x) = 1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + \frac{1}{24}x^4 = 1 + \sum_{m=1}^{4} \frac{x^m}{m!}$ for $x = 2.0$. Compare the accuracy of your estimate using an intrinsic library function.
[Hint: Use a do loop to evaluate the sum]

A2_3. Evaluate the following using 10,100, 500 and 1000 terms in the series expansion with $x=0.8$.

(i) $\frac{1}{1-x} = 1 + x + x^2 + x^3 + ... = \sum_{m=0}^{\infty} x^m$

(ii) $\ln(1+x) = x - \frac{1}{2}x^2 + \frac{1}{3}x^3 - \frac{1}{4}x^4 + ...$

(iii) $\cos(x) = 1 - \frac{1}{2}x^2 + \frac{1}{24}x^4 - \frac{1}{720}x^6 - .....$

How will you decide if your estimates with 10,100,500 and 1000 terms are accurate?

A2_4. Write a simple program to find out if a given number is even or odd.