### **Objectives**

• Basics of Fortran programming

```
program test
 implicit none
                                                      Output
 real :: s1, s2, s3, total
                                                 Sum 73.5999985
  s1 = 27.2
  s2 = 23.9
  s3 = 22.5
  total = s1 + s2 + s3
  write(*,*) ' Sum ', total
end program test
```

```
program test
 implicit none
                                                       Output
  real :: s(3), total
                                                  Sum 73.5999985
  s(1) = 27.2
  s(2) = 23.9
  s(3) = 22.5
  total = s(1) + s(2) + s(3)
  write(*,*) ' Sum ', total
end program test
```

### Introducing the arrays

```
Syntax: array_name(length_array) (one-dimensional) array_name(array_length, array_length) (two-dimensional)
```

#### How do you read this? 1D or 2D array?

Cartesian coordinates of 9 atoms (particles)

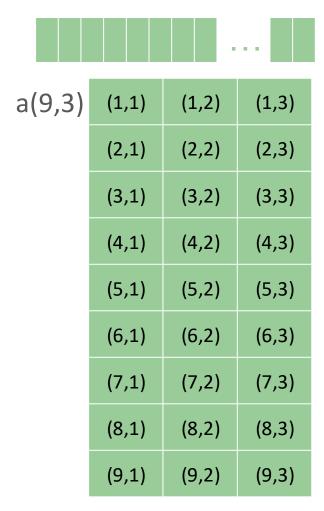
#### • Array operations:

• 
$$a = a + 2.0$$

$$\bullet$$
 a = a + a

• 
$$a(1,:) = a(1,:) * 2$$
 or  $a(1,:) = a(1,:) + 2$ 

a(27) or a(9)



#### **DO Loops**

DO LOOP is used to repeat a block of statements.

```
DO index_variable = start, end, step
---
END DO
```

- Index\_variable must be 'integer' type
- 'step' is optional

#### Example of DO Loops

```
program num
implicit none

integer :: i

do i=1,10

  write(*,*) "num", i

enddo

end program num
```

```
do x = 1, 10
! block 1
enddo
```

```
m=1
n=10
do x = m, n
! block 1
enddo
```

```
m=1
n=10
do x = m, n, 1
! block 1
enddo
```

```
m=1
n=10
x=1
do x= m, n*x
! block 1
enddo
```

#### **CYCLE and EXIT Statements**

- EXIT statement helps in transferring the control outside the DO loop
- CYCLE statement takes the control to the beginning of the next iteration in the DO loop

```
DO index_variable = start, end, step
! block 1 statements
exit
---
! block 2 statements
END DO
```

block 2 statements will not be executed

Usually both CYCLE and EXIT statements are used along with IF condition

### **DO WHILE Loops**

#### **Syntax**

DO WHILE (logical argument)

! block statements

**END DO** 

### Example of DO WHILE Loops

```
program num
 implicit none
 integer :: i
 i=0
 do while (i<10)
  i=i+1
   write(*,*) "num", i
 enddo
end program num
```

```
i=0
n=10
do while ( i < n)
! block 1
enddo
```

```
n=10
i=20
do while (.not. i < n)
! block 1
enddo
```

### Infinite DO Loops

```
Syntax
  DO
      ! block statements
       If (logical expression) then
          exit
       endif
      ! block statements
   END DO
```

#### nested DO loops

```
Syntax
  DO index_variable = start1, end1
     ! block 1 statements
     DO index variable = start2, end2
      ! block 2 statements
      DO index_variable = start3, end3
         ! block 3 statements
      END DO
      ! block 4 statements
     END DO
     ! block 5 statements
  END DO
```

```
program test
 implicit none
 integer :: i
 real :: s(100), total
! initialize the varaible s
 do i = 1, 100
     s(i) = i
 enddo
! main part of the program
 total=0.0
 do i = 1, 100
  total = total + s(i)
enddo
  write(*,*) ' Sum ', total
end program test
```

Output Sum 5050.00000

```
program test
 implicit none
                                                                         Output
 integer :: i
                                                                  Sum 5100.00000
 real :: s(100), total
! initialize the varaible s
 do i = 1, 100
     s(i) = i
 enddo
! main part of the program
 total=0.0
 do i = 1, 100
  total = total + s(i)
! condition
  if(i>50) then
    total = total + 1
  endif
enddo
  write(*,*) ' Sum ', total
end program test
```

• Write a FORTRAN program that computes the average and standard deviation of N real numbers. Use arrays and DO loops in the program.

Standard deviation 
$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$
.

Where

**x**<sub>i</sub> is an individual value

 $\mu$  is the mean/expected value

N is the total number of values

### Tips

- Don't worry about declaring variables initially. Identify the main part of the program and start writing
- All real numbers should be in double precision (add d0 in the end), eg. 10.0d0
- Always use indentation, leave black spaces to improve readability
- Always use 'parameter' in case when assigning the values to integer datatype
- Use internal functions to convert datatypes, eg. real(x)
- Read compiler error messages more carefully
- For debugging, use 'write' statement at several places in the program and check for the output

•

### FORTRAN – Reading material

- Please go through this FORTRAN program for a quick overview,
  - https://learnxinyminutes.com/docs/fortran95/
- Please go through this document for quick overview of FORTRAN
  - https://www.ldeo.columbia.edu/~mspieg/mmm/Fortran.pdf
- For a video on FORTRAN programming, look at
   https://www.youtube.com/watch?v= 2UgFNYgf8
- Book: Computer Programming in Fortran 90 and 95, V. Rajaraman