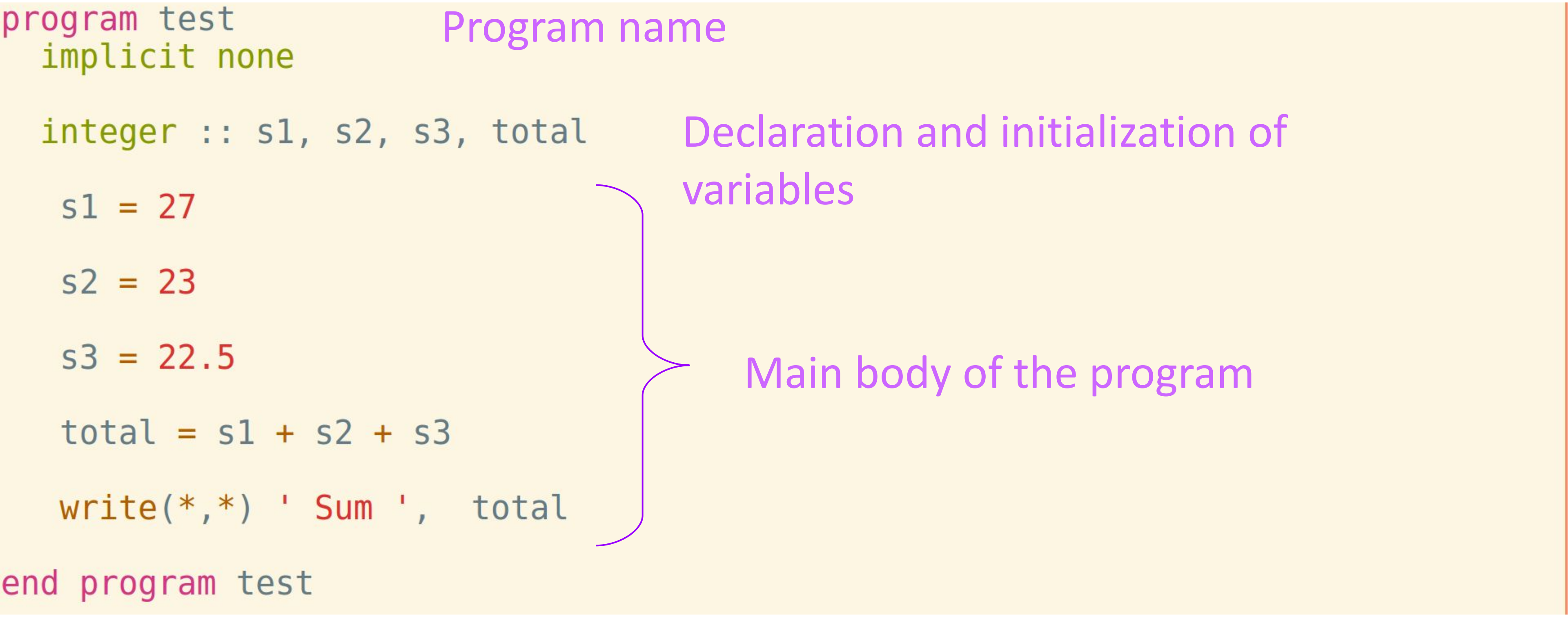
Topic: Fortran Programming

Objectives

- Basics of Fortran programming
- IF conditional statement
- DO loops

Topic: Fortran Programming

FORTRAN program has FOUR elements



```
program test
  implicit none

  integer :: s1, s2, s3, total

   s1 = 27

   s2 = 23

   s3 = 22.5

   total = s1 + s2 + s3

   write(*,*) ' Sum ',  total

end program test
```

Program name

Declaration and initialization of variables

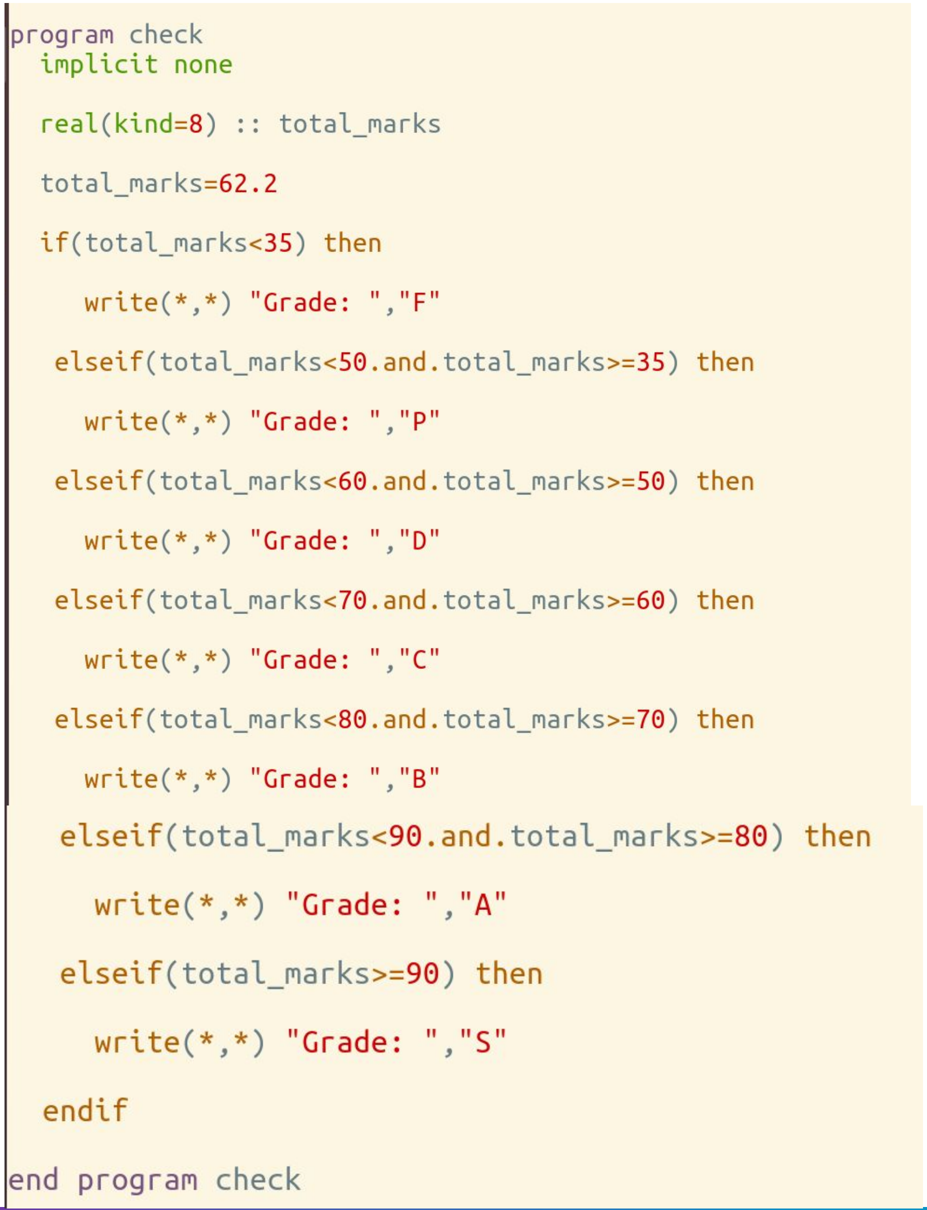
Main body of the program

Subprogram(s)

Structure of the FORTRAN program

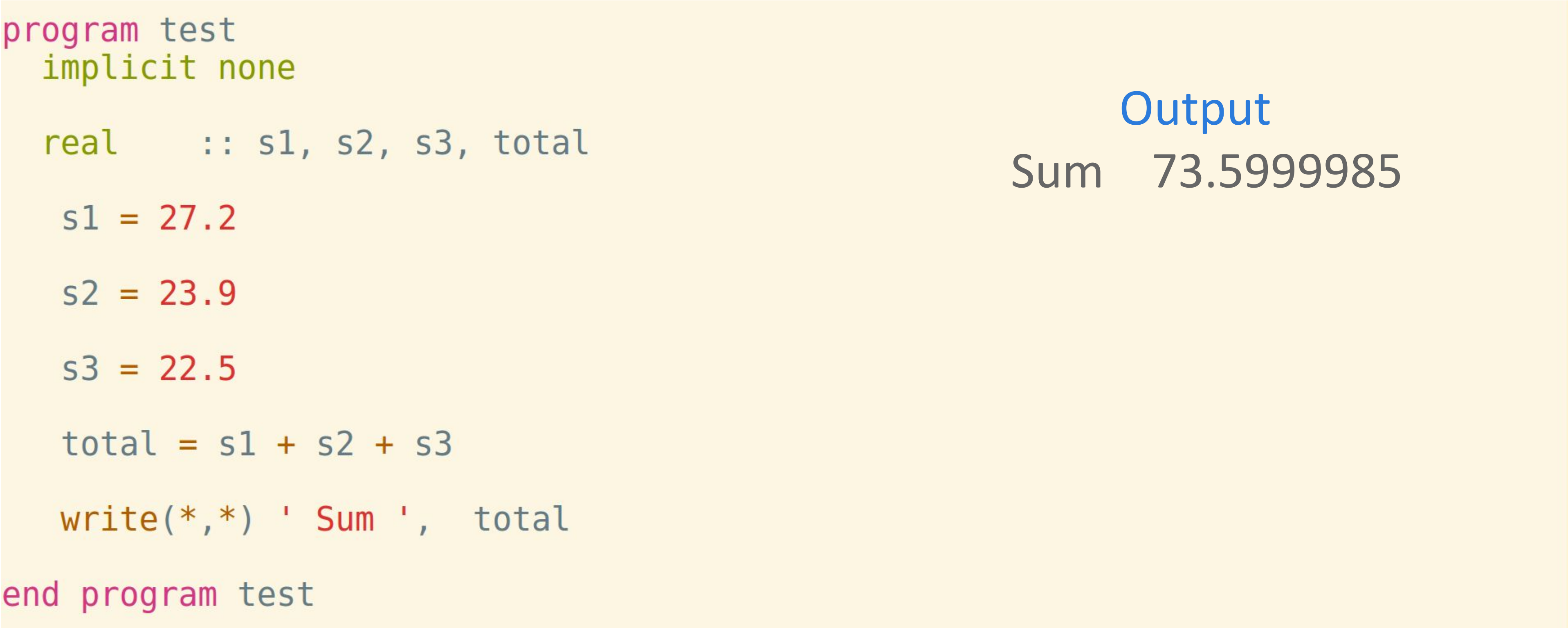
Topic: Fortran Programming

IF conditional statement



```
program check
  implicit none

  real(kind=8) :: total_marks

  total_marks=62.2

  if(total_marks<35) then

     write(*,*) "Grade: ","F"

   elseif(total_marks<50.and.total_marks>=35) then

     write(*,*) "Grade: ","P"

   elseif(total_marks<60.and.total_marks>=50) then

     write(*,*) "Grade: ","D"

   elseif(total_marks<70.and.total_marks>=60) then

     write(*,*) "Grade: ","C"

   elseif(total_marks<80.and.total_marks>=70) then

     write(*,*) "Grade: ","B"

   elseif(total_marks<90.and.total_marks>=80) then

     write(*,*) "Grade: ","A"

   elseif(total_marks>=90) then

     write(*,*) "Grade: ","S"

  endif

end program check
```

Topic: Fortran Programming

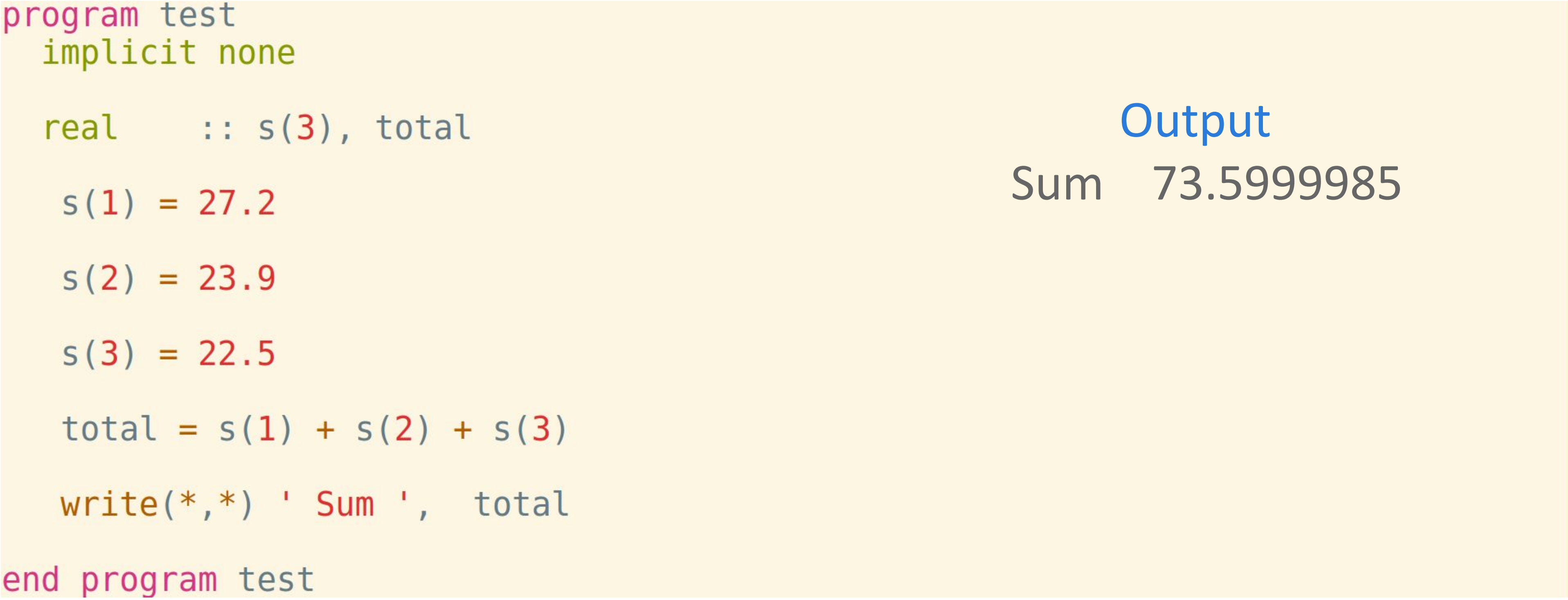


```fortran
program test
  implicit none

  real    :: s1, s2, s3, total

   s1 = 27.2

   s2 = 23.9

   s3 = 22.5

   total = s1 + s2 + s3

   write(*,*) ' Sum ',  total

end program test
```

Output

```
Sum   73.5999985
```

Topic: Fortran Programming



```
program test
  implicit none

  real    :: s(3), total

   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
```

Output

```
Sum   73.5999985
```

Introducing the arrays

Syntax: array_name(length_array) (one-dimensional)

array_name(array_length, array_length) (two-dimensional)

Topic: Fortran Programming

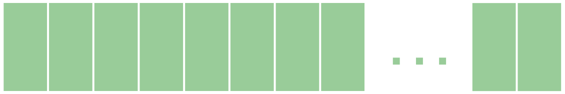
How do you read this? 1D or 2D array?

Cartesian coordinates of 9 atoms (particles)

1.2361419	1.0137761	-0.0612424
0.5104418	0.8944555	0.5514190
1.9926927	1.1973129	0.4956931
-0.9957202	0.0160415	1.2422556
-1.4542703	-0.5669741	1.8472817
-0.9377950	-0.4817912	0.4267562
-0.2432343	-1.0198566	-1.1953808
0.4367536	-0.3759433	-0.9973297
-0.5031835	-0.8251492	-2.0957959

- Array operations:
- a = a + 2.0
- a = a * 2.0
- a = a + a
- a = a * a
- a(1,:) = a(1,:) * 2   or   a(1,:) = a(1,:) + 2

a(27) or a(9)



a(9,3)

(1,1)	(1,2)	(1,3)
(2,1)	(2,2)	(2,3)
(3,1)	(3,2)	(3,3)
(4,1)	(4,2)	(4,3)
(5,1)	(5,2)	(5,3)
(6,1)	(6,2)	(6,3)
(7,1)	(7,2)	(7,3)
(8,1)	(8,2)	(8,3)
(9,1)	(9,2)	(9,3)

Topic: Fortran Programming

DO Loops

DO LOOP is used to repeat a block of statements.

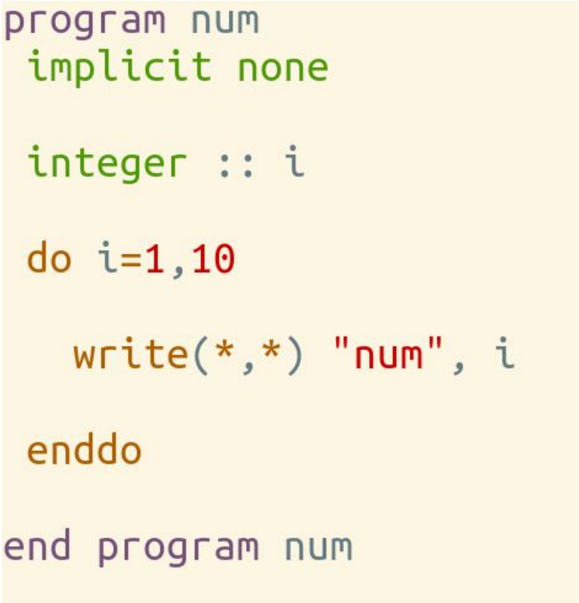
Syntax

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

- Index_variable must be 'integer' type
- 'step' is optional

Topic: Fortran Programming

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

Syntax

```
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

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 i = 1, 5
    if (mod(i, 2) == 0) then
      cycle  ! Skip even numbers
    end if
    write (*,*) "Current value:", i
end do
```

Topic: Fortran Programming

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 i = 1, 10
    if (i > 5) then
      exit  ! Exit the loop when i exceeds 5
    end if
    write (*,*) "Current value:", i
  end do
```

Topic: Fortran Programming

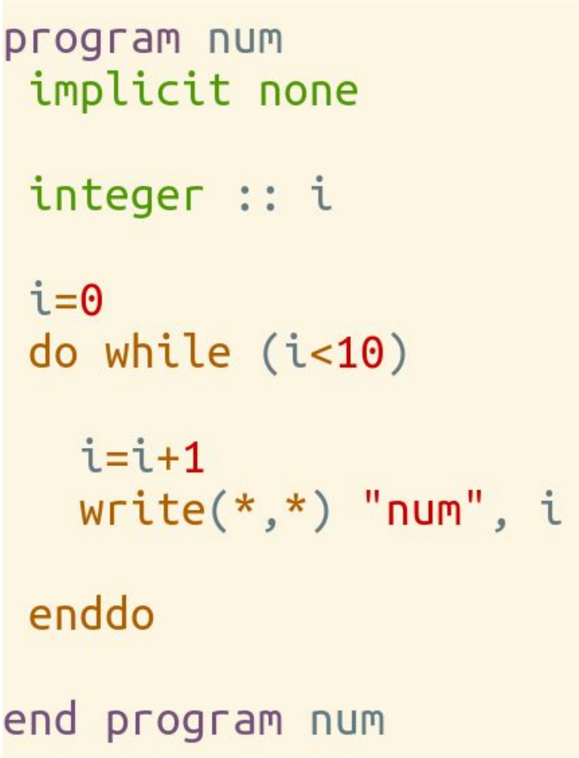
DO WHILE Loops

Syntax

```
DO WHILE (logical argument)
    ! block statements
 END DO
```

Topic: Fortran Programming

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
```

Topic: Fortran Programming

Infinite DO Loops

Syntax

```
DO
      ! block statements

       If (logical expression) then
           exit
        endif

      ! block statements

 END DO
```

```
do
    count = count + 1
    write (*,*) "Loop iteration:",
count
    if (count >= 5) then
      exit  ! Exit the loop
    end if
  end do
```

Topic: Fortran Programming

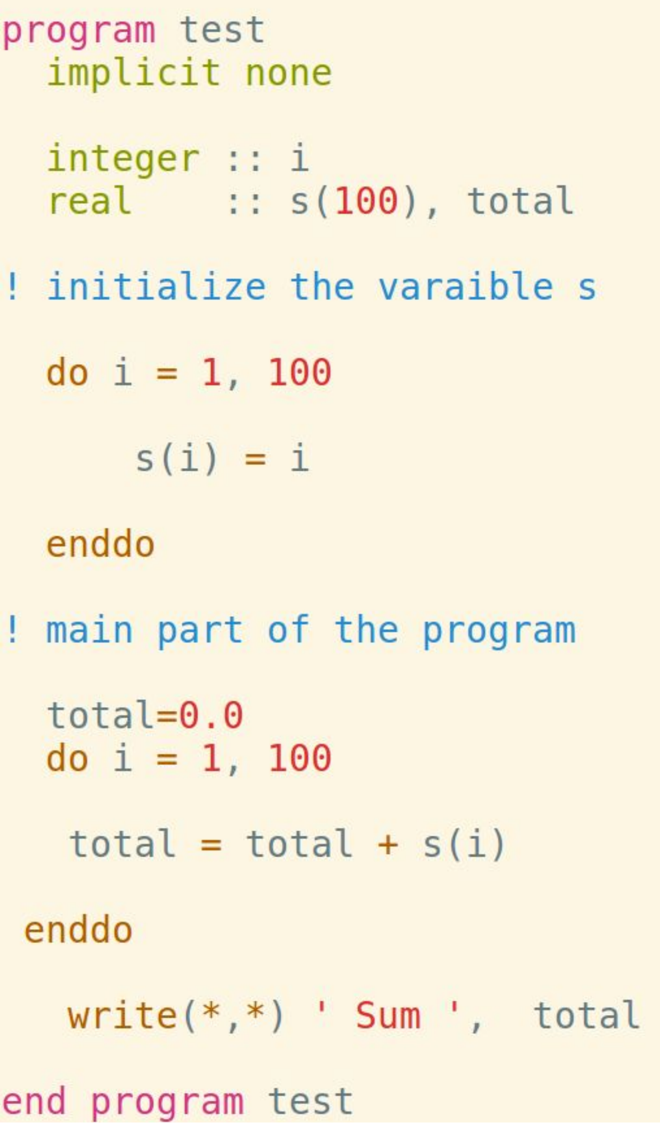
nested DO loops

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
```

```
 do i = 1, 2
    do j = 1, 3
      do k = 1, 4
        write (*,*) "i:", i, "j:", j, "k:", k
      end do
    end do
  end do
```

Topic: Fortran Programming



```fortran
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
```

Topic: Fortran Programming

Output

```
sum = 0
  do i = 1, 5
    sum = sum + i
  end do

  write (*,*) "The SUM is:", sum
```

Topic: Fortran Programming

Output

```
do i = 1, 3
    do j = 1, 2
      write (*,*) i, " * ", j, " = ", (i * j)
    end do
  end do
```

Topic: Fortran Programming

Output

```
do i = 3, 2
    write (*,*) i
  end do
```

Topic: Fortran Programming

Write a program that computes the distance a ball travels when thrown at a certain speed (v_0) and angle (θ). Given the initial velocity input by the user, calculate the range for angles ranging from 5 to 85 degrees in increments of 5 degrees.

$$\text{Range} = -(2*v_0^2/g)\cos\theta\sin\theta$$

Note that here θ is radians.

Topic: Fortran Programming

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
-

Topic: Fortran Programming

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
- Book: Computer Programming in Fortran 90 and 95, V. Rajaraman