Numerical Integration (Simpson's Rule)

Example: The dimensionless temperature of a fluid under steady, fully developed laminar flow in a cylindrical pipe with walls heated electrically, is given by

$$\boldsymbol{\theta = -4Z - r^2 + \frac{r^4}{4} + \frac{7}{24}}$$

where r is the dimensionless radius. The cup-mixing (bulk) dimensionless temperature is given by

$$\boldsymbol{\theta_b = 4\int_0^1 \theta\left(1 - r^2\right) r\, dr}$$

Generate values of $\boldsymbol{\theta}$ at Z = 0.5 and r = 0, 0.25, 0.5, 0.75, 1 and use the composite Simpson's rule to estimate $\boldsymbol{\theta_b}$ numerically.

Solution Procedure

Step 1: Write a subroutine to calculate the value of θ (**Output**), given r, Z (**Inputs**).

Step 2: Write a subroutine to calculate $\mathbf{y(r) = 4\left(\theta\left(1 - r^2\right)r\right)}$ **(Output), given θ and r (Inputs).**

Step 3: Write a main program

(i) that calculates θ at r = 0, 0.25. 0.5, 0.75, 1 **(call step 1 subroutine)**

(ii) using the above values of θ and r, calculate y(r) $(y_0, y_1, y_2, y_3, y_4)$ **(call step 2 subroutine)**

(iii) calculate the numerical integral of $\boldsymbol{y(r)}$ ($\boldsymbol{that\ is\ \theta_b(r)}$) using the composite Simpson's rule

Composite Simpson's Rule

$$\int_{r_0}^{r_4} y(r)\, dr = \frac{\Delta r}{3}\left[y_0 + 4y_1 + 2y_2 + 4y_3 + y_4\right]$$

Step 1: Write a subroutine to calculate the value of θ (**Output**), given r, Z (**Inputs**).

Note: For integer division, Please use "7.0/24.0" instead of "7/24" and apply the same syntax in case of any other places of integer division.

Actual Output: 7/24 = 0.2916

Syntax in Fortran

```
real :: a
a = 7/24
print *, a
```

✗ a = 0.0

```
real :: a
a = 7.0/24.0
print *, a
```

✓ a = 0.2916

Step 2: Write a subroutine to calculate $\mathbf{y(r) = 4\left(\theta\left(1 - r^2\right)r\right)}$ **(Output), given θ and r (Inputs).**

```
subroutine function_name(z,r,theta)
    implicit none
    real,intent(in) :: z,r
    real,intent(out) :: theta
    theta = - 4 * z - r ** 2 + (r ** 4)/4.0 + 7.0/24.0
end subroutine function_name
```

```
subroutine y_values(theta,r,y)
    implicit none
    real, intent(in) :: theta,r
    real, intent(out) :: y
    y = 4*theta*(1-r**2)*r

end subroutine y_values
```

Step 3: Write a main program for calculating the numerical integral.

Console Output

The numerical solution is -1.9976

(The analytical solution is -2).

```
program values
   implicit none
   real :: z, delr, theta_b
   real, dimension(5) :: r, theta, y
   integer :: i
   z = 0.5
   r(1) = 0
   do i = 2,5
      r(i) = r(i-1) + 0.25
   end do
   do i = 1,5
     call function_name(z,r(i),theta(i))
   end do

   delr = r(2) - r(1)
   do i = 1,5
     call y_values(theta(i),r(i),y(i))
   end do

   theta_b = (delr/3.0) * (y(1) + 4*y(2) + 2*y(3) + 4*y(4) + y(5))
   print *, "The numerical solution is",theta_b
end program values
```

Points to remember

- Given a problem for coding,
  - ✓ Understand the question clearly.
  - ✓ Make sure that the algorithm to be implemented is clear.
  - ✓ Divide the solution procedure into small steps (or subroutines).
  - ✓ Implement each step → Save the project → Build the project → Run as Fortran application.
- Except for subprograms, create a new Fortran project for every new program.
- Add the extension .f95 for every new Fortran source file.

Points to remember

- Do not forget to declare the variables being used in the code, especially the loop variables.

- We can call any number of subroutines in a program or subprogram.

- While calling subprogram(s) in the main program, the order of variables declared in both the programs should be equal.

subroutine	Main program
subroutine statistics(num1,num2,sum,Prod)	**call statistics(a,b,sum,Prod)** ✓
implicit none	
end subroutine statistics	**call statistics(sum,Prod,a,b)** ✗

- In case of integer division, write the integers as floating point numbers.

  Ex: Write 3.0/2.0 instead of 3/2