Date -
17/10/2022

PRACTICAL 6

- **Objective:** To write a C++ program for solving numerical integration by Simpson 1/3 rule.

- **Algorithm:**

```
1. Start
2. Define function f(x)
3. Read lower limit of integration, upper limit of integration and number of sub interval
4. Calculate: step size = (upper limit - lower limit)/number of sub interval
5. Set: integration value = f(lower limit) + f(upper limit)
6. Set: i = 1
7. If i > number of sub interval then goto
8. Calculate: k = lower limit + i * h
9. If i mod 2 =0 then Integration value = Integration Value + 2* f(k)
   Otherwise Integration Value = Integration Value + 4 * f(k)
   End If
10. Increment i by 1 i.e. i = i+1 and go to step 7
11. Calculate: Integration value = Integration value * step size/3
12. Display Integration value as required answer
13. Stop
```

- **Theory:**

$$\int_a^b f(x)dx \approx S_n = \frac{\Delta x}{3}[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \ldots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)]$$

- **Practical Code:**

```
#include<iostream>
#include<math.h>
#define f(x) 1/(1+pow(x,2))
using namespace std;

int main() {
 float lower, upper, integration=0.0, stepSize, k;
 int i, subInterval;
 cout<<"Enter lower limit of integration: ";
 cin>>lower;
 cout<<"Enter upper limit of integration: ";
 cin>>upper;
 cout<<"Enter number of sub intervals: ";
 cin>>subInterval;
 stepSize = (upper - lower)/subInterval;
 integration = f(lower) + f(upper);

 for(i=1; i<= subInterval-1; i++) {
  k = lower + i*stepSize;
  if(i%2==0) { integration = integration + 2 * f(k) ; }
  else { integration = integration + 4 * f(k) ; }
 }
 integration = integration * stepSize/3;
 cout<< endl <<"Required value of integration is: "<< integration;
 return 0;
}
```

- **Output:**

```
Enter lower limit of integration: 0
Enter upper limit of integration: 1
Enter number of sub intervals: 6
Required value of integration is: 0.785398
```

- **Application:**

  a. Used for solving complex integration problems.