(b) Write down the FORTRAN-95 or C or C++ expression for the algebraic expression $a = x + \frac{y}{z} - r^2 + c^3$.

```
a= x+y/z-r*r+c*c*c;
```

or

```
#include<math.h>
a=x+y/z-pow(r,2)+pow(r,3)
```

(c) How the following mathematical functions are invoked in FORTRAN-95 or C or C++?

(i) Absolute value of x

(ii) y to the power z

(i)
```
#include<stdlib.h>
int a;
a=abs(x);
```

(ii)
```
#include<math.h>
double a;
a=pow(z,y);
```

(d) What is nested loop?

A loop inside another loop is called a nested loop. The depth of the nested loop depends on the complexity of a problem. We can have any number of nested loops as required.

The syntax is

```
while(condition)
{
      while(condition)
      {
```

```
            .
            .
            .
            Statesments;
            .
            .
            .
        }
    }
```

(a) Briefly explain a statement use to implement looping in either FORTRAN-95 or C or C++.

`while(condition)body;`

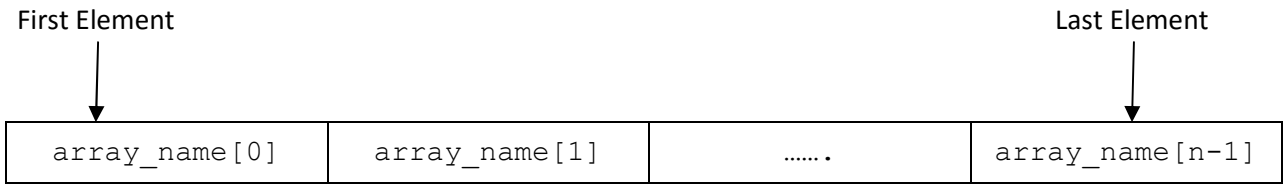
where the body can be either a single statement or a block of statements with in '{' and '}. '"while" loop is a condition-controlled loop, that means statements written in the body of the loop are being executed repeatedly until some condition is satisfied.

Example

```
int i=0;
while(i<=10)
{
      printf("%d ",i);
}
```

(b) What is an array? Write down the syntax for declaration of a one-dimensional array in either FORTRAN-95 or C or C++.

An array is a collection of data items, all of the same type, accessed using a common name. A one-dimensional array is like a list; A two dimensional array is like a table. The C language places no limits on the number of dimensions in an array. All arrays consist of contiguous memory locations. The lowest address corresponds to the first and the highest address to the last element.



Syntax for declaration

```
datatype array_name[size];
```

(c) Write down the FORTRAN-95 or C or C++ comparison operator corresponding to mathematical symbols (i) =, (ii) ≠, (iii) ≥ and (iv) ≤.

Mathematical Symbols	Symbols used in C Language
=	==
≠	!=
≥	>=
≤	<=

(b) How are the following mathematical functions written in FORTRAN-95 or C or C++ ?

(i) exponential (base e) of x.

(ii) natural logarithm (base e) of z.

(i)
```
#include<math.h>
double y;
y=exp(x);
```

(ii)
```
#include<math.h>
double p;
p=log(z);
```

(a) How will you represent the following ? Comment : "This program computes a solution to the equation", in FORTRAN-95 or C or C++.

```
/* This program computes a solution to the equation */
```

Or

```
// This program computes a solution to the equation
```

(b) Write one conditional and one logical operators each in FORTRAN-95 or C or C++.

Conditional operator: `? :`

`Syntax: expression1 ? expression2 : expression3`

expression2 is followed if condition is True and expression3 is followed if condition is False.

Logical operator: `&&` is called Logical AND operator. If both the operands are non-zero, then the condition becomes true.

(c) Write a brief statement to find square root of a natural number N in either FORTRAN-95 or C or C++.

```
#include<math.h>
double a;
a=sqrt(N);
```

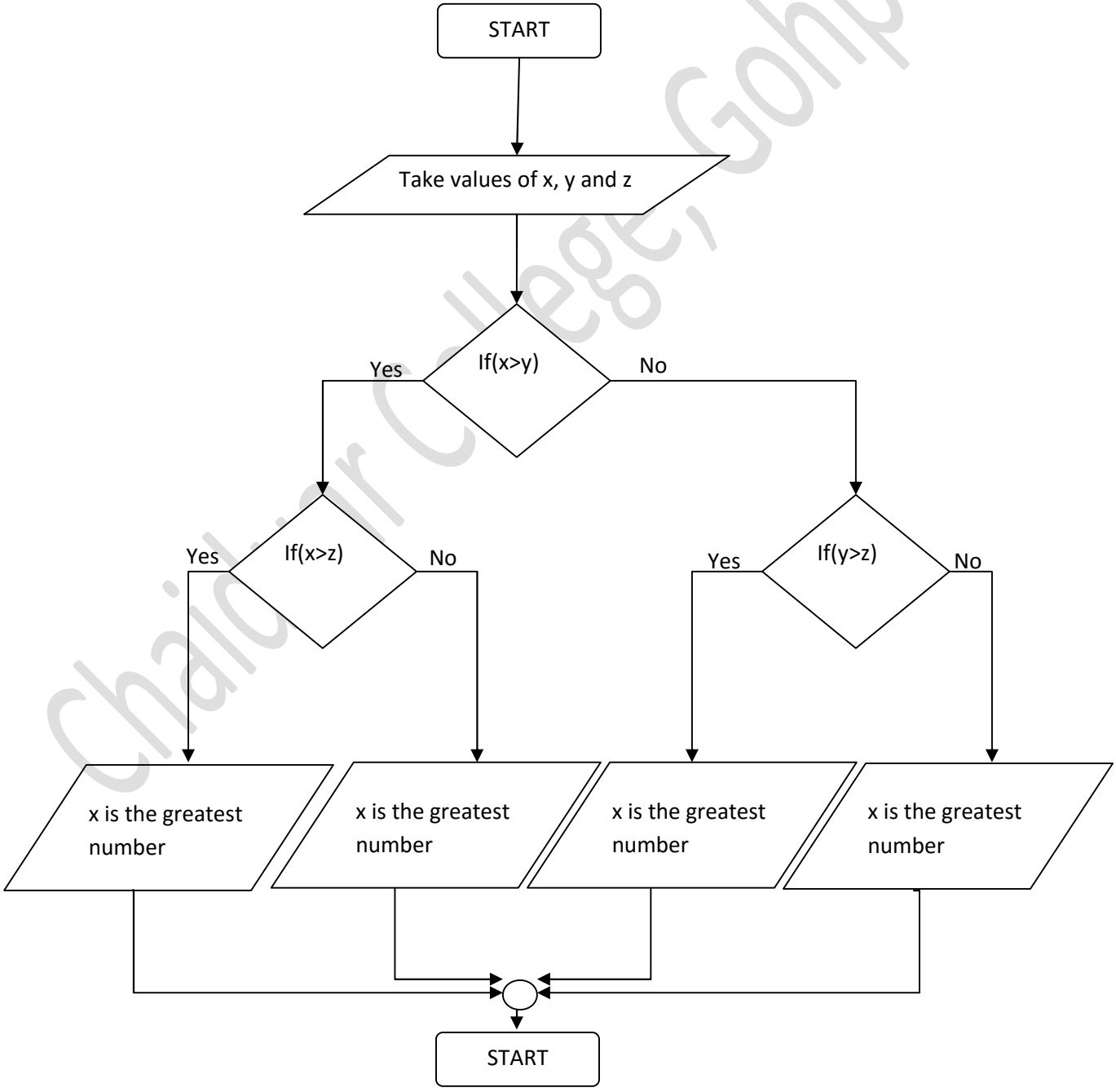
(a) Write down the flowchart and a program in either FORTRAN-95 or C or C++ to find the greatest of three given integers x, y and z.

```
#include<stdio.h>
void main()
{
      int x,y,z;
      printf("Enter the values of x, y and z");
      scanf("%d%d%d",&x,&y,&z);
      if(x > y)
      {
            if(x > z)
                  printf("%d is the greatest of %d,%d and %d",x,x,y,z);
            else
                  printf("%d is the greatest of %d,%d and %d",z,x,y,z);
```

```
        }
        else
        {
                if(y > z)
                        printf("%d is the greatest of %d,%d and %d",y,x,y,z);
                else
                        printf("%d is the greatest of %d,%d and %d",z,x,y,z);

        }
}
```

Flowchart:



(b) Write down the algorithm and a program in either FORTRAN-95 or C or C++ to find sum of N natural numbers.

Algorithm:

1. Read the value of N.
2. i=1, sum=0
3. sum=sum+i
4. i=i+1
5. repeat the step 3 and 4 till i=N
6. Display the value of sum
7. Stop

The Program has been given in the class

(a) Write down the input and output statements used in either FORTRAN-95 or C or C++.

Input statement:

```
scanf("format string",arg1,arg2,….);
```

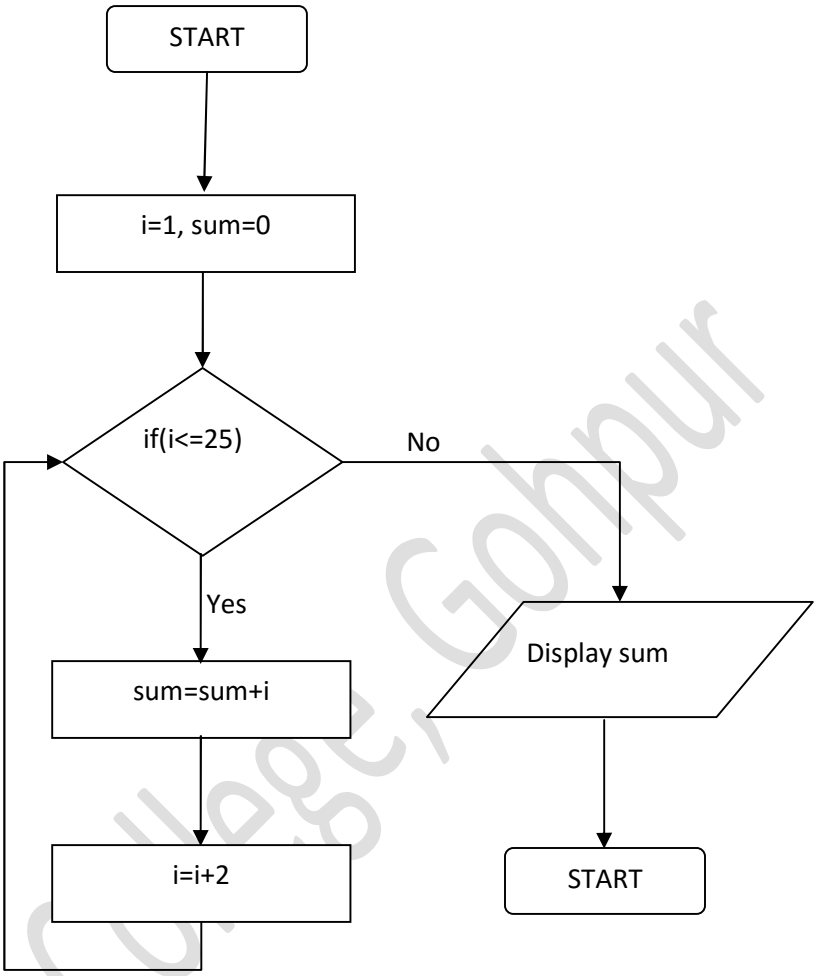
Output statement:

```
printf("format string",arg1,arg2,…..);
```

(a) Write down the flowchart and a program in either FORTRAN-95 or C or C++ to find the sum of the following series :

$$1+3+5+\ldots+25$$

The Program has been discussed in class.



(b) Write down the flow chart and a program in either FORTRAN-95 or C or C++ to find the sum of N odd numbers.

(For N odd numbers the above flowchart remain same instead of the condition part and to do so another variable will be required. Modify the flowchart accordingly.)

(a) Write down the flow chart and a program in either FORTRAN-95 or C or C++ to generate AP series with common difference 2 and number of elements 10 and also find its sum.

(The above flowchart can be use with little modification.)

(b) Write down the algorithm and a program in either FORTRAN-95 or C or C++ to generate first fifteen numbers of the series

0, 1, 1, 2, 3, 5, 8, ...

Algorithm:

1. a=0, b=1
2. Display a and b
3. c=a+b
4. a=b
5. b=c
6. Display the value of c
7. Repeat step to 3 to 6 for 13 times
8. Stop

The Program has been given in the class

(b) Prepare a program in either FORTRAN-95 or C or C++ to compute the real as well as imaginary roots of the quadratic equation $4x^2 - 2x + 9 = 0$.

Instead of the following lines of given program write the statements

```
a=4;
b=-2;
c=9;
```

```
printf("Enter coefficients a, b and c: ");
scanf("%lf %lf %lf",&a, &b, &c);
```

(a) Write a program in either FORTRAN-95 or C or C++ to compute the real roots of the following quadratic equation :

$ax^2 + bx + c = 0$ for $a = 5$, $b = -8$ and $c = 1$

Same as the above solution.

(a) $Y = \frac{2x^2 + 3}{3x^2 + 4}$

(b) $e^{x^2} + \frac{3x^3}{1 + x^2}$

(c) $Z = \frac{x \sin^{-1} x + 1}{x^3 + \cos^{-1} x}$

```
(a)   Y=(2*x*x+3)/(3*x*x+4);
(b)   exp(x*x)+(3*pow(x,3)/(1+x*x));
(c)   z=(x*asin(x)+1)/(pow(x,3)+acos(x))
```

(b) How are the following mathematical functions expressed in FORTRAN-95 or C or C++?

(i) Absolute value of $x^2 + 3y^2$

(ii) Logarithm (base 10) of x^3

```
(i)     abs(x*x+3*y*y);
(ii)    double a=pow(x,3);
        log10(a);
```

(b) Write a program in either FORTRAN-95 or C or C++ to determine mean and standard deviation of given experimental data.

(a) Write a program in either FORTRAN-95 or C or C++ to compute the solution of the following simultaneous linear equations :

$a_1x + b_1y = c_1$

$a_2x + b_2y = c_2$

(a) Prepare a program in either FORTRAN-95 or C or C++ to compute the real as well as imaginary roots of the quadratic equation $ax^2 + bx + c = 0$.

Programs have been given in the class.

Develop an algorithm and write a program in either FORTRAN-95 or C or C++ to compute the numerical solution of the equation

$$\frac{dy}{dx} = \frac{1}{2}(1+x)y^2$$

in the interval [0, 1] having initial value $y = 1$ at $x = 0$ and step size $h = 0 \cdot 1$ using Runge-Kutta fourth order method.

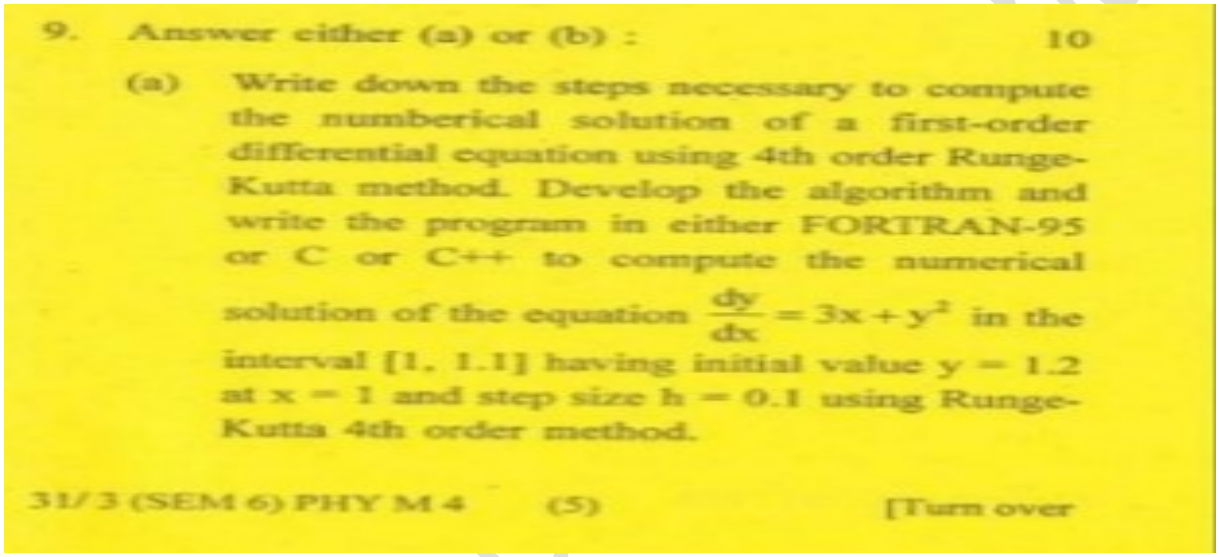
```
#include<stdio.h>
float dydx(float x, float y)
{
    return(1/2*(1+x)*y*y);
}
float rungeKutta(float x0, float y0, float x, float h)
{
    int n = (int)((x - x0) / h);
    float k1, k2, k3, k4, k5;
    float y = y0;
    for (int i=1; i<=n; i++)
    {
        k1 = h*dydx(x0, y);
        k2 = h*dydx(x0 + 0.5*h, y + 0.5*k1);
        k3 = h*dydx(x0 + 0.5*h, y + 0.5*k2);
        k4 = h*dydx(x0 + h, y + k3);
        y = y + (1.0/6.0)*(k1 + 2*k2 + 2*k3 + k4);
        x0 = x0 + h;
    }

    return y;
}
void main()
{
    float x0 = 0, y = 1, x, h = 0.1,r;
    printf("Enter the value of X between 0 and 1");
    scanf("%f",&x);
    r=rungeKutta(x0, y, x, h);
    printf("\nThe value of y at x is : %f");
}
```

Algorithm:

1. Take the initial value of x (x0=0), initial value of y (y0=1) and step size h (h=0.1)

2. Calculate the number of iterations using step size h
   n = (x - x0) / h where x is the value at which y to be calculated
3. Find the value of f(x, y)=1/2*(1+x)*y*y for each x0 and y
4. Calculate k1 = h*f(x0, y)
5. Calculate k2 = h*f(x0 + 0.5*h, y + 0.5*k1)
6. Calculate k3 = h*f(x0 + 0.5*h, y + 0.5*k2)
7. Calculate k4 = h*f(x0 + h, y + k3)
8. Calculate next value of y using Runge Kutta Formulas y= y + (1.0/6.0)*(k1 + 2*k2 + 2*k3 + k4)
9. x0=x0+h
10. Repeat the steps 4 to 9 for n times
11. Display the value of y.



9. Answer either (a) or (b) : 10

(a) Write down the steps necessary to compute the numberical solution of a first-order differential equation using 4th order Runge-Kutta method. Develop the algorithm and write the program in either FORTRAN-95 or C or C++ to compute the numerical solution of the equation $\frac{dy}{dx} = 3x + y^2$ in the interval [1, 1.1] having initial value y = 1.2 at x = 1 and step size h = 0.1 using Runge-Kutta 4th order method.

31/3 (SEM 6) PHY M 4 (5) [Turn over

Following values will be different

```
x0 = 1, y = 1.2, x, h = 0.1
```

and under the function dydx the return value will be

```
return(3*x+y*y);
```

The steps necessary to compute the numerical solution of a first order …….

$$k_1 = hf(x_n, y_n)$$
$$k_2 = hf(x_n + \frac{h}{2}, y_n + \frac{k_1}{2})$$
$$k_3 = hf(x_n + \frac{h}{2}, y_n + \frac{k_2}{2})$$
$$k_4 = hf(x_n + h, y_n + k_3)$$
$$y_{n+1} = y_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(h^5)$$

(b) Give the mathematical relations used to compute numerical value of an integral using Simpson's one-third rule. Write the flowchart and a program in either FORTRAN-95 or C or C++ to compute the numerical value of the integral

$$\int_0^1 \frac{dx}{1+x^2}$$

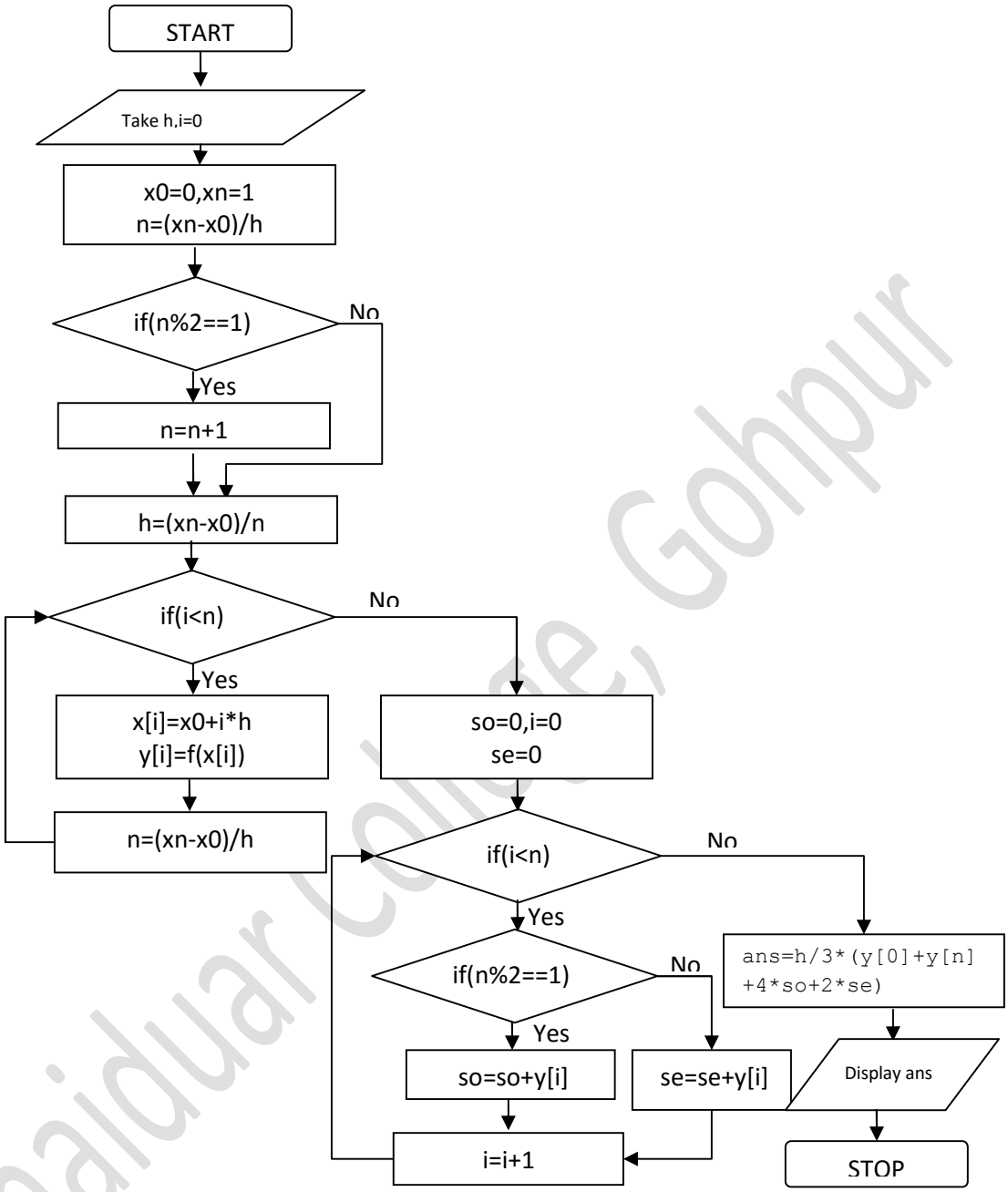
using Simpson's one-third rule.

```
#include<stdio.h>
float f(float x)
{
    return(1/(1+x*x));
}
void main()
{
    int i,n;
    float x0,xn,h,y[20],so,se,ans,x[20];
    x0=0;
    xn=1;
    printf("\n Enter values of h: ");
    scanf("%f",&h);
    n=(xn-x0)/h;
    if(n%2==1)
    {
        n=n+1;
    }
    h=(xn-x0)/n;
    printf("\n Refined value of n and h are:%d %f\n",n,h);
    printf("\n Y values: \n");
    for(i=0; i<=n; i++)
    {
```

```
        x[i]=x0+i*h;
        y[i]=f(x[i]);
        printf("\n %f\n",y[i]);
    }
    so=0;
    se=0;
    for(i=1; i<n; i++)
    {
        if(i%2==1)
        {
            so=so+y[i];
        }
        else
        {
            se=se+y[i];
        }

    }
    ans=h/3*(y[0]+y[n]+4*so+2*se);
    printf("\n Final integration is %f",ans);
}
```

Flowchart

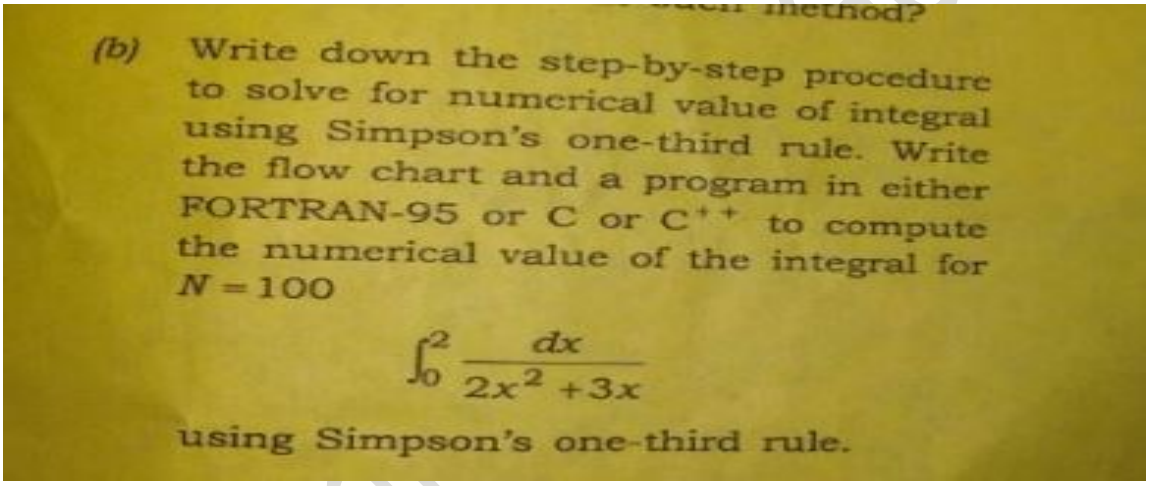


(b) Write the mathematical relations needed to compute numerical value of a finite size integral using Simpson's one-third rule : Write the flowchart and a program in either FORTRAN-95 or C or C++ to compute the numerical value of the integral $\int_0^1 \frac{x^2 dx}{1+x^3}$ using Simpson's one-third rule.

```
Change the function f as follows
float f(float x)
{
    return(x*x/(1+x*x));
}
```

Mathematical relation

- x_0 is the value of lower boundary value of x, x_n is the upper boundary value of x i.e. x_n and width of the strip, h.
- The value of number of strip as n=($x_n - x_0$)/h and checks whether it is even or odd. If the value of 'n' is odd, refine the value of 'h' so that the value of 'n' comes to be even.
- After that, calculate the value of f(x) i.e 'y' at different intermediate values of 'x.
- Integral = $\frac{h}{3} \times((y_0 + y_n) + 4(y_1 + y_3 + \ldots\ldots\ldots + y_{n-1}) + 2(y_2 + y_4 + \ldots\ldots\ldots + y_{n-2}))$


(b) Write down the step-by-step procedure to solve for numerical value of integral using Simpson's one-third rule. Write the flow chart and a program in either FORTRAN-95 or C or C++ to compute the numerical value of the integral for $N = 100$

$$\int_0^2 \frac{dx}{2x^2 + 3x}$$

using Simpson's one-third rule.

Change the f function and calculate h where N is given.