Exercise 2: Design a program to implement Runge - Kutta Methods.

1. **Find y(1.1),y(1.2) if $dy/dx = x^3+(y/2)$ using R-K method of 4^{th} order where y(1) =2 through C-Programming.**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
/*#define f(x,y) ((pow(X,3))+(y/2))*/
float x,y;
#define f(x,y) (pow(x,3)+(y/2))
int main()
{
 float x0,y0,xn,h,yn,k1, k2, k3, k4,k;
  int i, n;
  printf("Enter Initial Condition\n");
 printf("x0 = ");
 scanf("%f", &x0);
 printf("y0 = ");
 scanf("%f", &y0);
 printf("Enter calculation point xn = ");
 scanf("%f", &xn);
 printf("Enter number of steps: ");
 scanf("%d", &n);
 printf("Enter step-size: ");
 scanf("%f", &h);
 /* Runge Kutta Method */
 for(i=0; i < n; i++)
 {
  k1 =h*f(x0,y0);
  k2 = h*f((x0+h/2),(y0+k1/2));
  k3 =h*f((x0+h/2),(y0+k2/2));
  k4 =h*f((x0+h),(y0+k3));
  k = ((k1+2*k2+2*k3+k4)/6);
  yn = y0 + k;
  printf("\nx%d\ty%d\n",i,i);
  printf("%0.4f\t%0.4f\n",x0,y0);
  x0 = x0+h;
  y0 = yn;
 }
 printf("\nx%d\ty%d\n",n,n);
  printf("%0.4f\t%0.4f\n",x0,y0);
    getch();
}
```

OUTPUT

Enter Initial Condition

```
x0 = 1
y0 = 2
Enter calculation point xn = 1.2
Enter number of steps: 2
Enter step-size: 0.1

x0     y0
1.0000  2.0000

x1     y1
1.1000  2.2214

x2     y2
1.2000  2.4913
```

2. Find y(0.7),y(0.8) if $dy/dx = y - x^2$ using R-K method of 4^{th} order where y(0.6) =1.7379 through C-Programming.

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
/*#define f(x,y) ((pow(X,3))+(y/2))*/
float x,y;
#define f(x,y) (y-x*x)
int main()
{
 float x0,y0,xn,h,yn,k1, k2, k3, k4,k;
  int i, n;
  printf("Enter Initial Condition\n");
 printf("x0 = ");
 scanf("%f", &x0);
 printf("y0 = ");
 scanf("%f", &y0);
 printf("Enter calculation point xn = ");
 scanf("%f", &xn);
 printf("Enter number of steps: ");
 scanf("%d", &n);
 printf("Enter step-size: ");
 scanf("%f", &h);
 /* Runge Kutta Method */
 for(i=0; i < n; i++)
 {
  k1 =h*f(x0,y0);
  k2 = h*f((x0+h/2),(y0+k1/2));
```

```
  k3 =h*f((x0+h/2),(y0+k2/2));
  k4 =h*f((x0+h),(y0+k3));
  k = ((k1+2*k2+2*k3+k4)/6);
  yn = y0 + k;
  printf("\nx%d\ty%d\n",i,i);
  printf("%0.4f\t%0.4f\n",x0,y0);
  x0 = x0+h;
  y0 = yn;
 }
 printf("\nx%d\ty%d\n",n,n);
  printf("%0.4f\t%0.4f\n",x0,y0);
    getch();
}
```

OUTPUT

```
Enter Initial Condition
x0 = 0.6
y0 = 1.7379
Enter calculation point xn = 0.8
Enter number of steps: 2
Enter step-size: 0.1

x0      y0
0.6000  1.7379

x1      y1
0.7000  1.8763

x2      y2
0.8000  2.0145
```