MA202 LAB4

Name: Dipean Dasgupta ID:202151188

1. Write a C-program to numerically solve the ODE: $y'(t) = -\lambda y(t)$, to determine $y(10)$, given $y(0) = 10$. This to be done by implementing two methods: Runge-Kutta 2nd Order and Runge-Kutta 4th order, while taking step sizes $h = 0.001$ and $h = 0.01$ respectively. Consider three cases when $\lambda = 0.01, 0.1, 1$. Plot the results that you obtain in all the three cases and common upon them.

Solution Code:

Case1: RK 2nd Order

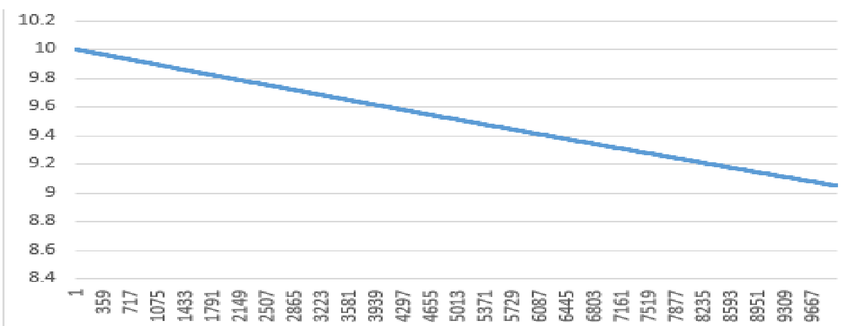
```
#include<stdio.h>
#include <math.h>
double function(float lmda,float y){
return (-1)*lmda*y;
}
double yS1(float h0,float x0,float y0,float lmda){
return h0*function(lmda,y0);
}
double yS2(float h0,float x0,float y0,float lmda,float h1){
return h0*function(lmda,y0 + yS1(h0,x0,y0,lmda));
}
int main(){
float x0,y0;
float h0,h1,lmda;
printf("Enter the value of h0: ");
scanf("%f",&h0);
printf("Enter the value of h1: ");
scanf("%f",&h1);
printf("Enter the value of x0: ");
scanf("%f",&x0);
printf("Enter the value of y0: ");
scanf("%f",&y0);
printf("Enter the value of lmda: ");
scanf("%f",&lmda);
float k1 = yS1(h0,x0,y0,lmda);
float k2 = yS2(h0,x0,y0,lmda,h1);
float k = (k1+k2)/2;
FILE *fp;
fp = fopen("output.txt","w");
```

```
int i=0;
while(x0<10 && i<10000){
x0 = x0 + h0;
y0 = y0 + k;
k1 = yS1(h0,x0,y0,lmda);
k2 = yS2(h0,x0,y0,lmda,h1);
k = (k1+k2)/2;
printf("%f\n",y0);
i++;
fprintf(fp,"%f\n",y0);
}
}
```

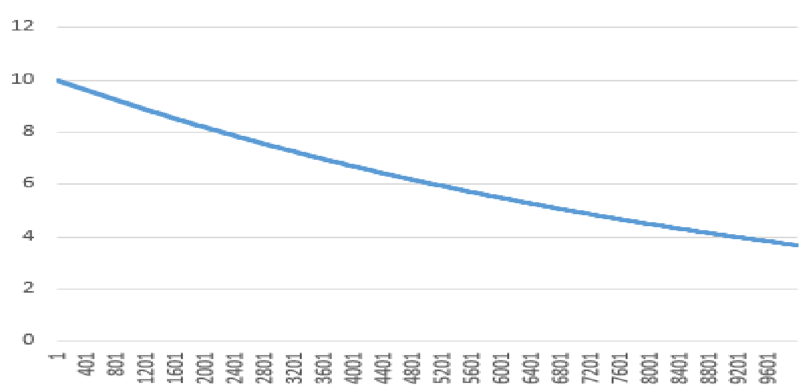
OUTPUT:

Step Size, h:0.001;

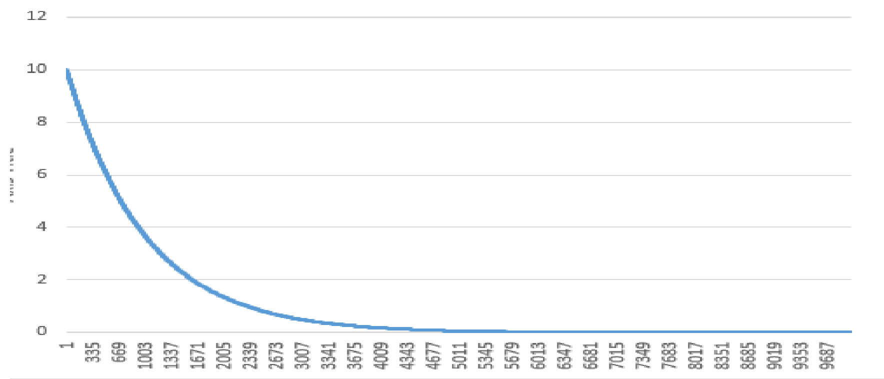
I) Lambda:0.01



II) lambda = 0.1



III) **For lambda = 1**



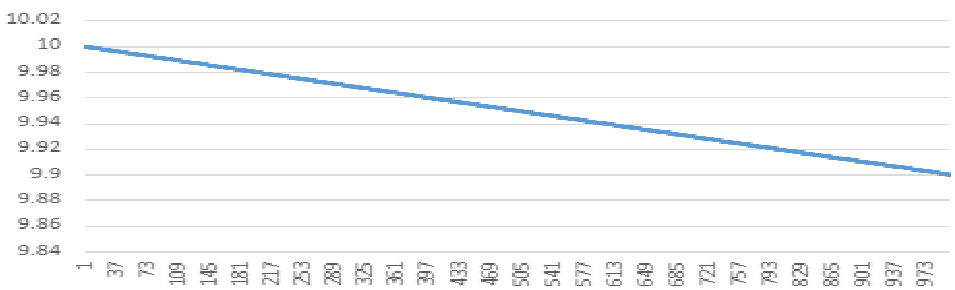
Case2: RK 4th Order

```
#include<stdio.h>
double Fnc(float lmda,float y){
return (-1)*lmda*y;
}
double yS1(float h0,float x0,float y0,float lmda){
return h0*Fnc(lmda,y0);
}
double yS2(float h0,float x0,float y0,float lmda){
return h0*Fnc(lmda,y0 + yS1(h0,x0,y0,lmda)/2);
}
double yS3(float h0,float x0,float y0,float lmda){
return h0*Fnc(lmda,y0 + (yS2(h0,x0,y0,lmda)/2));
}
double yS4(float h0,float x0,float y0,float lmda){
return h0*Fnc(lmda,y0 + yS3(h0,x0,y0,lmda));
}
int main(){
float x0,y0;
float h0,lmda;
printf("Enter the value of h0: ");
scanf("%f",&h0);
printf("Enter the value of x0: ");
scanf("%f",&x0);
printf("Enter the value of y0: ");
scanf("%f",&y0);
printf("Enter the value of lmda: ");
scanf("%f",&lmda);
float k1 = yS1(h0,x0,y0,lmda);
float k2 = yS2(h0,x0,y0,lmda);
float k3 = yS3(h0,x0,y0,lmda);
float k4 = yS4(h0,x0,y0,lmda);
```

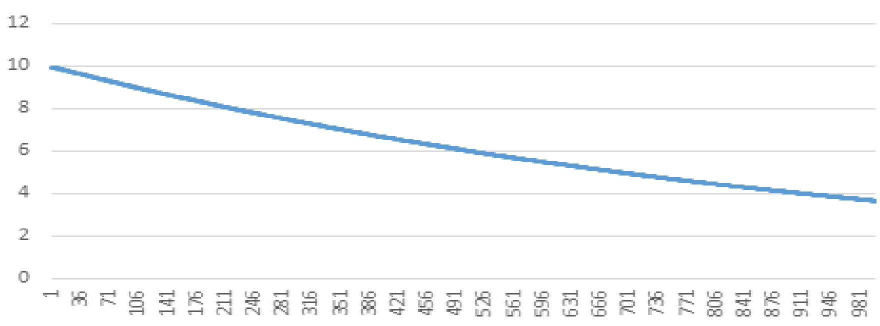
```
float k = (k1+2*k2+2*k3+k4)/6;
FILE *fp;
fp = fopen("output1.txt","w");
int i=0;
while(x0<10 && i<=1000){
x0 = x0 + h0;
y0 = y0 + k;
k1 = yS1(h0,x0,y0,lmda);
k2 = yS2(h0,x0,y0,lmda);
k3 = yS3(h0,x0,y0,lmda);
k4 = yS4(h0,x0,y0,lmda);
k = (k1+2*k2+2*k3+k4)/6;
printf("%f\n",y0);
i++;
fprintf(fp,"%f\n",y0);
}
}
```

Step Size, h:0.01;

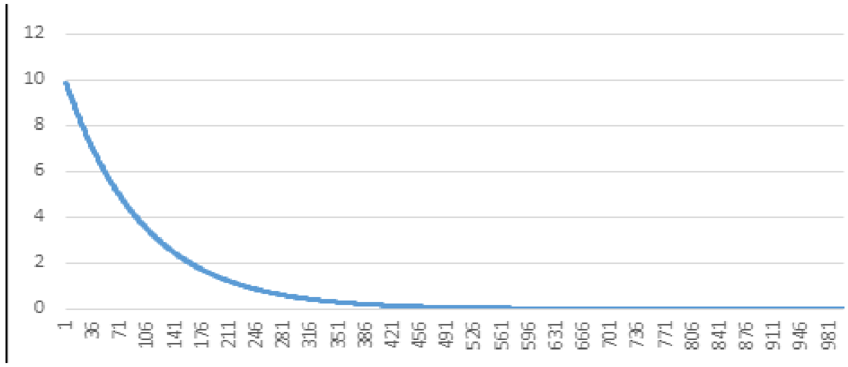
I) Lambda:0.01



II) lambda = 0.1



III) **lambda = 1**



2. Modify the above program to solve the nonlinear ODE $y'(t) = -\lambda y(t) + \epsilon y^2(t)$, using Runge-Kutta 4th order with the initial condition $y(0) = 1$. Consider $\lambda = 1$ and $\epsilon = 1.001$. Plot the results and comment upon upon it. What happens when $\epsilon = 0.1$?

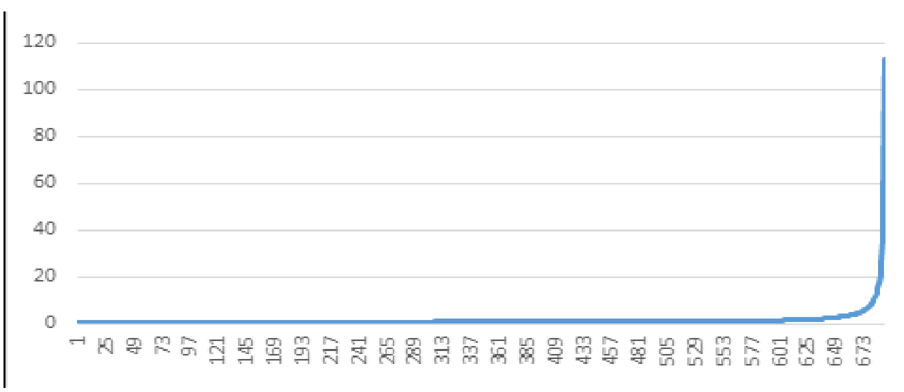
Solution Code:

```
#include<stdio.h>
double Fnc(double lmda,double y,double eps){
return ((-1)*lmda*y) + eps*y*y;
}
double yS1(double h0,double x0,double y0,double lmda,double eps){
return h0*Fnc(lmda,y0,eps);
}
double yS2(double h0,double x0,double y0,double lmda,double eps){
return h0*Fnc(lmda,y0 + yS1(h0,x0,y0,lmda,eps)/2,eps);
}
double yS3(double h0,double x0,double y0,double lmda,double eps){
return h0*Fnc(lmda,y0 + (yS2(h0,x0,y0,lmda,eps)/2),eps);
}
double yS4(double h0,double x0,double y0,double lmda,double eps){
return h0*Fnc(lmda,y0 + yS3(h0,x0,y0,lmda,eps),eps);
}
int main(){
double x0,y0;
double h0,lmda,eps;
printf("Enter the value of h0: ");
scanf("%lf",&h0);
printf("Enter the value of x0: ");
scanf("%lf",&x0);
```

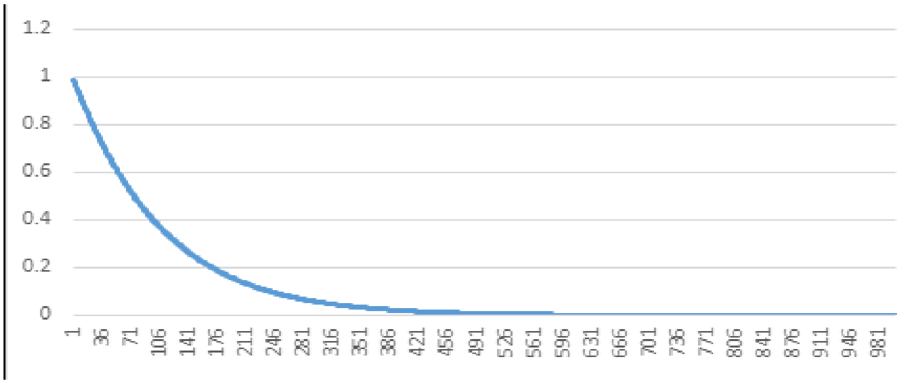
```
printf("Enter the value of y0: ");
scanf("%lf",&y0);
printf("Enter the value of lmda: ");
scanf("%lf",&lmda);
printf("Enter the value of eps: ");
scanf("%lf",&eps);
double k1 = yS1(h0,x0,y0,lmda,eps);
double k2 = yS2(h0,x0,y0,lmda,eps);
double k3 = yS3(h0,x0,y0,lmda,eps);
double k4 = yS4(h0,x0,y0,lmda,eps);
double k = (k1+2*k2+2*k3+k4)/6;
FILE *fp;
fp = fopen("output2.txt","w");
int i=0;
while(x0<10 && i<=1000){
x0 = x0 + h0;
y0 = y0 + k;
k1 = yS1(h0,x0,y0,lmda,eps);
k2 = yS2(h0,x0,y0,lmda,eps);
k3 = yS3(h0,x0,y0,lmda,eps);
k4 = yS4(h0,x0,y0,lmda,eps);
k = (k1+2*k2+2*k3+k4)/6;
i++;
printf("%lf\n",y0);
fprintf(fp,"%lf\n",y0);
}
}
```

OUTPUT PLOT:

Case1: epsilon = 1.001



Case2: epsilon = 0.1



3. Can you write a program to solve SHO equation $x''(t) + \omega^2 x(t) = 0$, for various values of initial conditions $x(0) = x_0$ and $x'(0) = v_0$, for $\omega = 1$. Are you able to reproduce the simple harmonic motion ? Provide the plots to validate your results.

Solution Code:

```
#include <stdio.h>
#include <math.h>
// Defining the SHO equation
double SHM(double x, double v) {
return -x;
}
// Implement the fourth-order Runge-Kutta method
void RK4(double *x, double *v, double dt) {
double kx1, kv1, kx2, kv2, kx3, kv3, kx4, kv4;
kx1 = dt * (*v);
kv1 = dt * SHM(*x, *v);
kx2 = dt * (*v + 0.5 * kv1);
kv2 = dt * SHM(*x + 0.5 * kx1, *v + 0.5 * kv1);
kx3 = dt * (*v + 0.5 * kv2);
kv3 = dt * SHM(*x + 0.5 * kx2, *v + 0.5 * kv2);
kx4 = dt * (*v + kv3);
kv4 = dt * SHM(*x + kx3, *v + kv3);
*x += (kx1 + 2.0 * kx2 + 2.0 * kx3 + kx4) / 6.0;
*v += (kv1 + 2.0 * kv2 + 2.0 * kv3 + kv4) / 6.0;
}
int main() {
// Setting conditions and time step
double xo = 1.0;
```

```c
double vo = 0.0;
double t = 0.0;
double dt = 0.1;
for (int i = 0; i < 101; i++) {                //Solving SHM EQN using RK4
printf("%lf\t%lf\t%lf\n", t, xo, vo);
RK4(&xo, &vo, dt);
t += dt;
}
return 0;
}
```

OUTPUT: