ASSIGNMENT 02 SOLUTIONS (ALL IN ONE)

Creating a c file `integral.c` where I've written the functions of integrations methods (Trapezoidal,Simpson's 1/3), root finding methods (Bisection and Secant) and a factorial function.

In [ ]:

```c
// it contains functions to evaluate the integration and roots
// (trapezidal,simpson,bisection, secant)
// use #include"integral.c" in the program you wished to use this

//Function to perform factorial
double factorial(int n)
{
  int i;
  double fact=1;
  for(i=n;i>=1;i--)
  {
    fact=fact*i;
  }
  return fact;
}
/************************************************/
/**************TRAPEZOIDAL METHOD***************/
/************ONE VARIABLE EXPRESSION************/
/************************************************/
double trap(double f(double x),double a,double b)
{
    int i,n=2;  // starting with two interval
    double integral,answer,x,h,sum,accuracy=0.00001;
    do{
        integral=answer;
        h=fabs(b-a)/n;
        sum=0;
        for(i=1;i<n;++i)
        {
            x=a+i*h;
            sum=sum+f(x);
        }
        answer=(h/2)*(f(a)+f(b)+2*sum);
        n++;
    }while(fabs(answer-integral)>=accuracy);
  return answer;
}
/************************************************/
/**************TRAPEZOIDAL METHOD***************/
/************TWO VARIABLE EXPRESSION************/
/************************************************/
double trapezoidal(double f(double x,double var),double var,double a,double b)
{
    int i,n=2;  // starting with two interval
    double integral,answer,x,h,sum,accuracy=0.00001;
    do{
        integral=answer;
        h=fabs(b-a)/n;
        sum=0;
        for(i=1;i<n;++i)
        {
            x=a+i*h;
            sum=sum+f(x,var);
```

```
        }
        answer=(h/2)*(f(a,var)+f(x,var)+2*sum);
        n++;
    }while(fabs(answer-integral)>=accuracy);
  return answer;
}
/***********************************************/
/**************SIMPSON'S 1/3 METHOD**************/
/*************ONE VARIABLE EXPRESSION*************/
/***********************************************/
double simp13(double f(double x),double a,double b)
{
    int i,n=2;  // starting with two interval
    double integral,answer,x,h,sum,accuracy=0.00001;
    do{
        integral=answer;
        h=fabs(b-a)/n;
        sum=0;
        for(i=1;i<n;++i)
        {
            x=a+i*h;
            if(i%2==0){
                sum=sum+2*f(x);
            }
            else{
                sum=sum+4*f(x);
            }
        }
        answer=(h/3)*(f(a)+f(b)+sum);
        n=n+2;
    }while(fabs(answer-integral)>=accuracy);
  return answer;
}
/***********************************************/
/**************SIMPSON'S 1/3 METHOD**************/
/**************TWO VARIABLE EXPRESSION************/
/***********************************************/
double simpson13(double f(double x,double var),double var,double a,double b)
{
    int i,n=2;  // starting with two interval
    double integral,answer,x,h,sum,accuracy=0.00001;
    do{
        integral=answer;
        h=fabs(b-a)/n;
        sum=0;
        for(i=1;i<n;++i)
        {
            x=a+i*h;
            if(i%2==0){
                sum=sum+2*f(x,var);
            }
            else{
                sum=sum+4*f(x,var);
            }
        }
        answer=(h/3)*(f(a,var)+f(b,var)+sum);
        n=n+2;
    }while(fabs(answer-integral)>=accuracy);
  return answer;
}
/***********************************************/
```

```c
/***************ROOT FINDING***************/
/*************BISECTION METHOD*************/
/******************************************/
double bisection(double f(double x),float a,float b)
{
  double x,xm,xl,xr,accuracy=0.00001,xinc=0.5,z;
  for (x=a;x<=b;x+=xinc)
    {
      if (f(x)*f(x+xinc)<=0)
      {
        xl=x;
        xr=x+xinc;
        do
        {
          xm=(xl+xr)/2.0;
          if (f(xm)*f(xl)>=0)
          {
            xl=xm;
          }
          if (f(xm)*f(xl)<=0)
          {
            xr=xm;
          }
          z=fabs((xl-xr)/(xl+xr));
          //printf("xm=%f\tf(xm)=%f\tz=%f\taccuracy=%f\n",xm,J0(xm),z,accuracy);
        }
        while(z>accuracy);
        printf("\nroot=%f\tf(xm)=%f\tz=%f\taccuracy=%f\n",xm,f(xm),z,accuracy);
      }
    }
}
/******************************************/
/***************ROOT FINDING***************/
/***************SECANT METHOD**************/
/******************************************/
double secant(double f(double x), double a, double b)
{

    double x1,x2,x3; // x1 starting point a, x2 tending from a to b
    for(x1=a;x1<=b;x1=x1+0.01)
    {
        x2=x1+0.01;
        if(f(x1)*f(x2)<=0)
        {
            do{
                x3=(x1*f(x2)-x2*f(x1))/(f(x2)-f(x1));
                x1=x2;
                x2=x3;
            }while(fabs(f(x3))>0.00001);
            printf("\nIn the interval: %.3lf and %.3lf\n",x1,x2);
            printf("The root is: %.4lf\n",x3);
        }
    }
}
```

Now since we have written all the required functions in the `integral.c`, we'll use `#include"integral.c"` as a library.

How to use the functions?

Trapezoidal Rule

- Single variable function f(x): `trap(f,a,b)`
- Two variable function g(x,var): `trapezoidal(g,var,a,b)`

Simpsons 1/3 Rule

- Single variable function f(x): `simp13(f,a,b)`
- Two variable function g(x,var): `simpson13(g,var,a,b)`

Where **f(x)** and **g(x,var)** are the functions to be integrated while a is the lower limit and b is the upper limit of integration.

- Bisection Method: `bisection(f,a,b)`
- Secant Method: `secant(f,a,b)`

Where **f** is the function which roots we are looking while **[a,b]** is the range of course roots.

PROBLEM 1:

In [ ]:
```c
// problem 1
// make sure "integral.c" file is in the same directory
#include<stdio.h>
#include<math.h>
#include"integral.c"

// defining the function to evaluate
double fx(double x)
{
    return atan(x)/(x*x);
}
int main()
{
    float a=5,b=10,tra,sim;
    tra=trap(fx,a,b);   // using trapezoidal function
    printf("The integral using trapezoidal Rule is: %lf\n",tra);
    sim=simp13(fx,a,b);   // using simpsons 1/3 function
    printf("The integral using Simpson's Rule is: %lf\n",sim);
}
```

OUTPUT:

The integral using trapezoidal Rule is: 0.142294

The integral using Simpson's Rule is: 0.142205

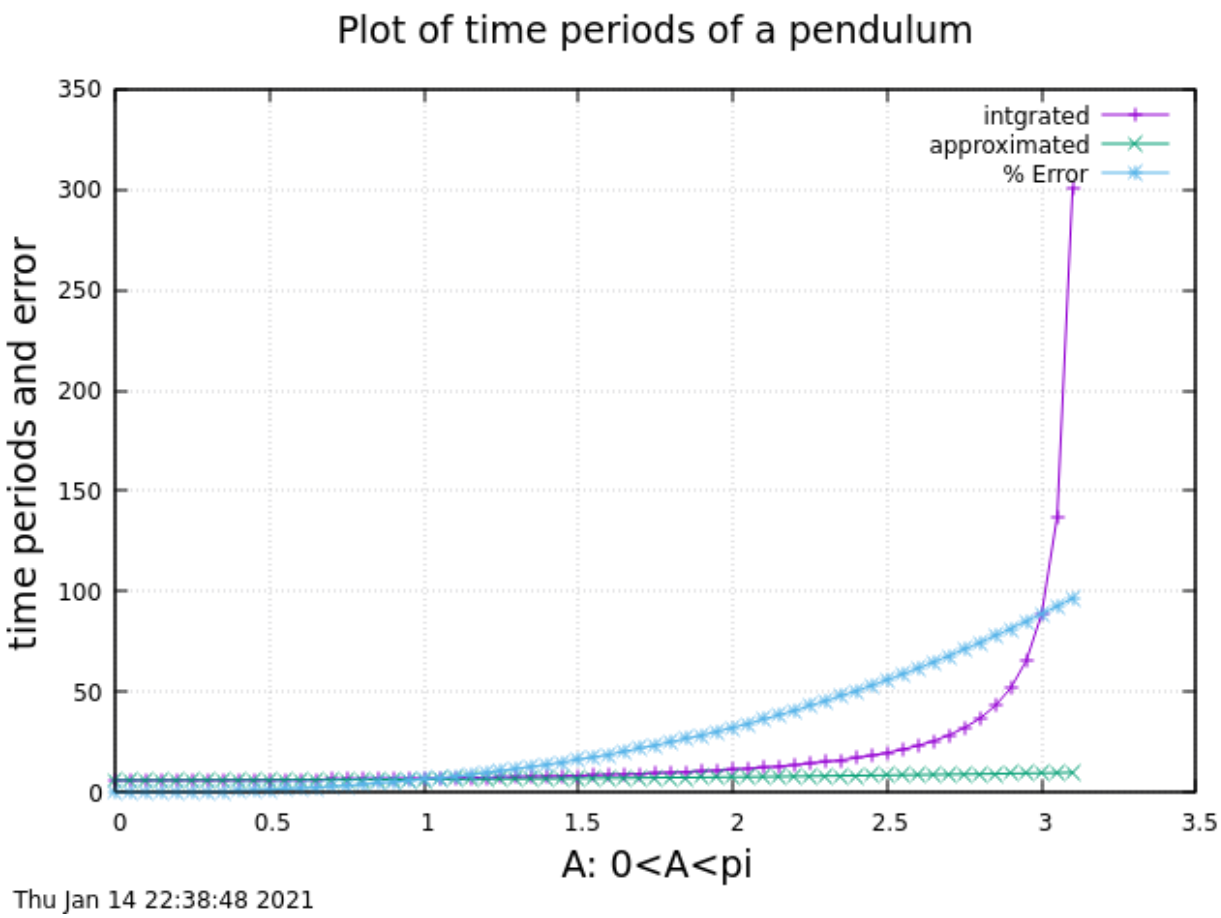
PROBLEM 2:

In [ ]:
```c
// problem 2
// make sure "integral.c" file is in the same directory
#include<stdio.h>
#include<math.h>
#include"integral.c"
#define pi 3.1415927

// function to be intgrated fxa(x,A)
double fxa(double x,double A){
```

```
    return 1/(1-sin(A/2)*sin(A/2)*sin(x)*sin(x));
}
// function for whole time-period T(f,A)
double T(double f(double x,double A),double A){
    return 4*simpson13(fxa,A,0,pi/2);
}
// function for approx time-period T1(A)
double T1(double A){
    return 2*pi*(1+pow(A/4,2));
}
// main function to do our job
int main()
{
    FILE*fp=NULL;
    fp=fopen("prob2.txt","w");

    double A,t,t1,error;
    // getting values for A range
    for(A=0;A<=pi;A+=0.05)
    {
        t=T(fxa,A);   // time-period
        t1=T1(A);   // approx time-period
        error=(t-t1)/t*100; // % error in both
        fprintf(fp,"%lf\t%lf\t%lf\t%lf\n",A,t,t1,error);
    }
}
```

OUTPUT

Program generated a text file "prob2.txt" and the plot of this data file is below:



PROBLEM 3:

In [ ]:

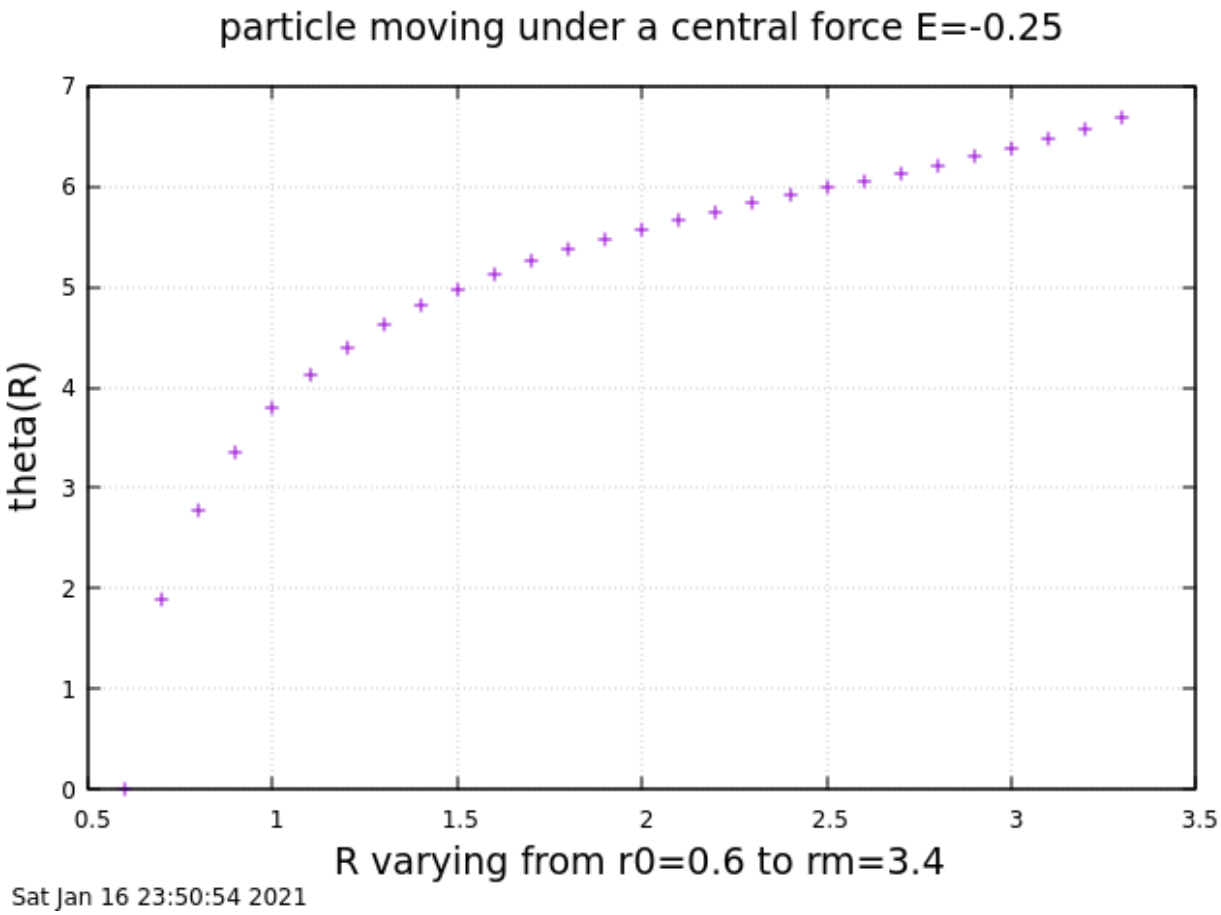
```c
// problem 3
#include<stdio.h>
#include<math.h>
// function to be intgrated fre(r,E)
double fre(double r,double E)
{
    return (1/(r*r*sqrt(2*E+2/r-1/(r*r))));
}
// gauss quadrature function to evaluate integration
double gauss(double f(double r,double E),double r,double E,double a, double b)
{
    double x1,x2;
    x1=((b-a)/2.0)*(1/1.73)+((b+a)/2);
    x2=((b-a)/2.0)*(-1/1.73)+((b+a)/2);
    return (b-a)/a*(f(x1,E)+f(x2,E));
}
int main()
{
    FILE*fp=NULL;
    fp=fopen("prob3.txt","w");

    double E,r,r0,rm;
    printf("Enter the value of E:");
    scanf("%lf",&E);
    printf("Enter the lower and upper limit r0 & rm:");
    scanf("%lf%lf",&r0,&rm);

    // varaying R from r0 to rm
    for (r=r0;r<=rm;r=r+0.1)
    {
        fprintf(fp,"%lf\t%lf\n",r,gauss(fre,r,E,r0,r));
    }
}
```

OUTPUT

Enter the value of E:-0.25

Enter the lower and upper limit r0 & rm:0.6 3.4

a text file `prob3.txt` *has been generated and the plot is:*

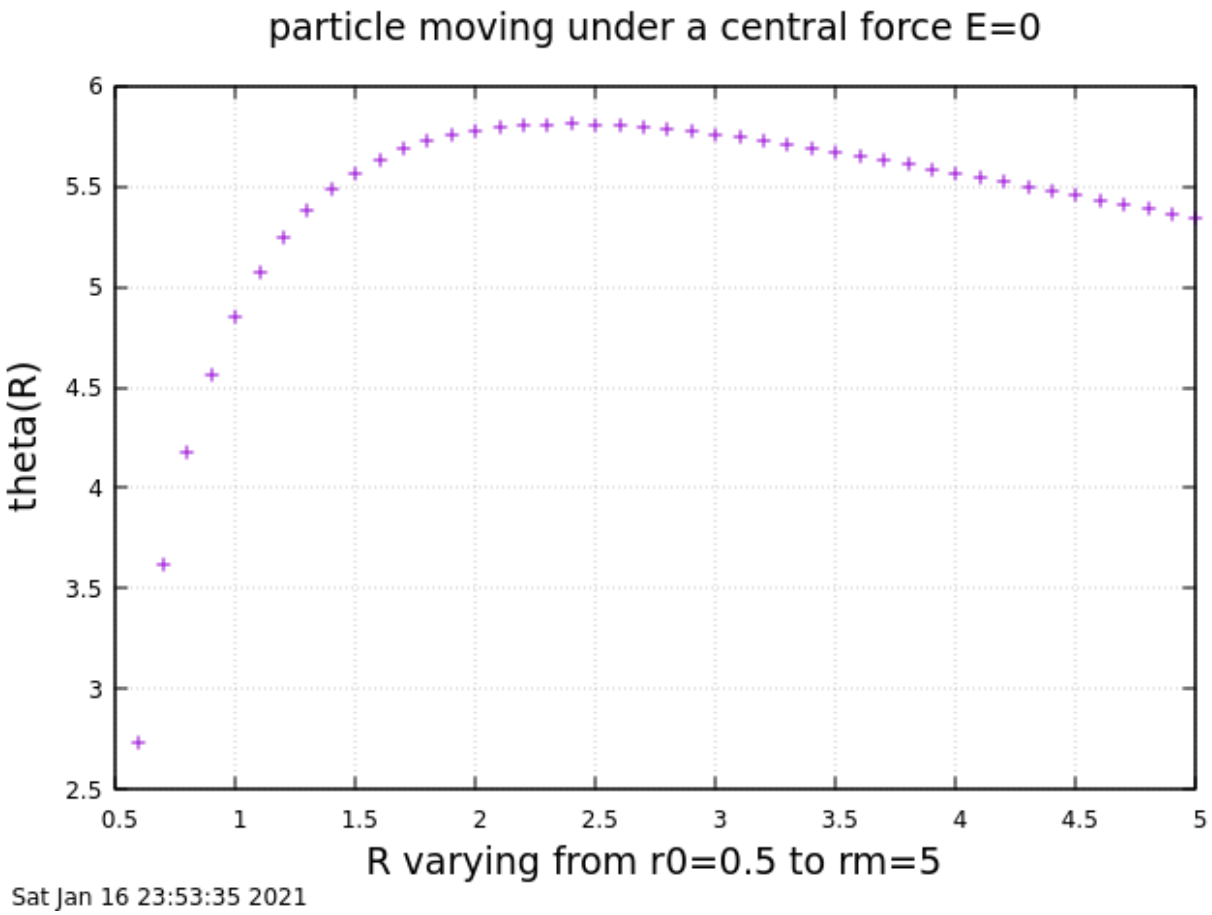


Again running the program for different E,r0,rm values

Enter the value of E:0

Enter the lower and upper limit r0 & rm:0.5 5

a text file `prob3.txt` *has been generated and the plot is:*



PROBLEM 4:

In [ ]:

```c
// problem 4
// make sure "integral.c" file is in the same directory
#include<stdio.h>
#include<math.h>
#include"integral.c"
#define pi 3.1415927

// function to be intgrated ftx(t,x)
double ftx(double t,double x){
    return cos(pow(x,1.5)*cos(t))*pow(sin(t),3);
}
// function which root is needed Fx(x)
double Fx(double x){
    return simpson13(ftx,x,0,pi);
}
// main program to do neccessary job
int main()
{
        float a=0,b=5; // range of roots
        printf("USING BISECTION METHOD\n");
        bisection(Fx,a,b);   // using bisection function
        printf("\nUSING SECANT METHOD\n");
        secant(Fx,a,b);   // using secant function
}
```

OUTPUT

USING BISECTION METHOD

root=2.723053 f(xm)=-0.000013 z=0.000006 accuracy=0.000010
root=3.907898 f(xm)=0.000004 z=0.000008 accuracy=0.000010
root=4.917297 f(xm)=-0.000000 z=0.000006 accuracy=0.000010

USING SECANT METHOD

In the interval: 2.723 and 2.723
The root is: 2.7230

In the interval: 3.913 and 3.908
The root is: 3.9079

In the interval: 4.923 and 4.917
The root is: 4.9173

HENCE THE SMALLEST ROOT IS:2.7230 (secant)

PROBLEM 5:

In [ ]:
```c
// problem 5
// make sure "integral.c" file is in the same directory
#include<stdio.h>
#include<math.h>
#include"integral.c"
#define pi 3.1415927

// function to be intgrated fxz(x,z)
double fxz(double x,double z){
    return cos(z*cos(x));
}
// function which root is needed J0(z)
double J0(double z){
    return 1/(2*pi)*simpson13(fxz,z,0,2*pi);
}
// main program to do neccessary job
int main()
{
  float a=0,b=12;       // range of roots
  secant(J0,a,b);       // using secant function
}
```

OUTPUT

In the interval: 2.410 and 2.405
The root is: 2.4048

In the interval: 5.530 and 5.520
The root is: 5.5201

In the interval: 8.660 and 8.654
The root is: 8.6537

In the interval: 11.800 and 11.792
The root is: 11.7915

PROBLEM 6:

In [ ]:

```c
// problem 6
// make sure "integral.c" file is in the same directory
#include<stdio.h>
#include<math.h>
#include"integral.c"
#define pi 3.1415927

double f(double theta,double z){
    return cos(z*cos(theta))*pow(sin(theta),5);
}
double J2(double z){
    return pow(z,2)/(pow(2,3)*factorial(2))*simpson13(f,z,0,pi);
}

int main()
{
  float a=0,b=10;       // range of roots
  printf("USING BISECTION METHOD\n");
  bisection(J2,a,b);   // using bisection function
  printf("\nUSING SECANT METHOD\n");
  secant(J2,a,b);   // using secant function
}
```

OUTPUT

USING BISECTION METHOD

root=0.499992 f(xm)=0.016371 z=0.000008 accuracy=0.000010
root=5.763489 f(xm)=-0.000000 z=0.000005 accuracy=0.000010
root=9.095093 f(xm)=0.000006 z=0.000007 accuracy=0.000010

USING SECANT METHOD

In the interval: 0.010 and 0.000
The root is: 0.0000

In the interval: 5.770 and 5.763
The root is: 5.7635

In the interval: 9.100 and 9.095
The root is: 9.0950