Algorithm: Uncertainty Principle

Include predefined class libraries / header files in the program using preprocessor directive #include <iostream> , <cmath>

using namespace std;

Class declaration: declare a class “unpr”

Private member declaration:

Data variables:
declare integer type data variables “i,n”
declare float type data variables “a,b,x[100],p[100],I[100],X,P,H,h,so1,se1,so2,se2,pi”

Public member declaration:
Declare member function (1) getdata (2) calculate of type void

Member function definitions:
Define member functions getdata using scope resolution operator

“void unpr::getdata()”

Get input from user:

cout<<"enter the lower and upper limit of integration"<<endl;
The input is written in allocated memory space using
cin>>a>>b;

cout<<"enter the even no of intercvals"<<endl;
The input is written in allocated memory space using
cin>>n;

Member function definition:
Define member functions calculate using scope resolution operator
“void unpr::calculate()”

Perform the calculations

```
h=(b-a)/n;                                                    //calculate step size
pi=4.0*atan(1);
   for(i=0;i<=n;i++)                                  //for loop to calculate (<x^2>,<p^2>)  at various x values
   {
      I[i]=a+(i*h);
      x[i]=(1/sqrt(pi))*pow(I[i],2)*exp(-pow(I[i],2));
      p[i]=(1/sqrt(pi))*exp(-pow(I[i],2))*(1-pow(I[i],2));
   }

   for(i=1;i<=n-1;i++)                          //for loop to calculate even ad odd terms (Simpson’s 1/3rd rule)
   {

      if(i%2==0)
```

```
        {
           se1=se1+x[i];
           se2=se2+p[i];
        }

        else
        {
           so1=so1+x[i];
           so2=so2+p[i];
        }

    }

    X=(h/3)*(x[0]+(4*so1)+(2*se1)+x[n]);          //Simpson's 1/3rd rule for numerical integration
    P=(h/3)*(p[0]+(4*se2)+(2*se2)+p[n]);

    cout<<"uncertainity in position "<<X<<endl;
    cout<<"uncertainity in momentum "<<P<<endl;
    H=X*P;
    cout<<"value of unbcertainty product"<<H<<endl;          //Uncertainty Product
}
```

Inside the main function of type integer , create an object of class unpr and named UP. Use this object to call public member functions and terminate the program using return 0 statement

```
int main()
{
    unpr UP;                                  //create object
    UP.getdata();                             //member function calling
    UP.calculate();
    return 0;                                 //program termination
}
```