KARPAGAM ACADEMY OF HIGHER EDUCATION
(Deemed to be University Established Under Section 3 of UGC Act 1956)
Coimbatore – 641 021.

SYLLABUS
DEPARTMENT OF PHYSICS

STAFF NAME: Mrs.A. SAHANA FATHIMA **CLASS:** I B.Sc., (PHYSICS)
SUBJECT NAME: MATHEMATICAL PHYSICS PRACTICAL **SUB.CODE:** 18PHU213
SEMESTER: II

Objectives: The aim of this Lab is to use the computational methods to solve physical problems. Course will consist of lectures (both theory and PRACTICAL) in the Lab. Evaluation done not on the programming but on the basis of formulating the problem

Introduction to Numerical computation software Scilab

Introduction to Scilab, Advantages and disadvantages, Scilab environment, Command window, Figure window, Edit window, Variables and arrays, Initialising, variables in Scilab, Multidimensional arrays, Subarray, Special values, Displaying output data, data file, Scalar, and array operations, Hierarchy of operations, Built in Scilab functions, Introduction to plotting, 2D and 3D plotting, Branching Statements and program design, Relational & logical operators, the while loop, for loop, details of loop operations, break & continue statements, nested loops, logical arrays and vectorization User defined functions, Introduction to Scilab functions, Variable passing in Scilab, optional arguments, preserving data between calls to a function, Complex and Character data, string function, Multidimensional arrays an introduction to Scilab file processing, file opening and closing, Binary I/o functions, comparing binary and formatted functions, Numerical methods and developing the skills of writing a program.

Curve fitting, Least square fit:

Goodness Ohms law to calculate R, Hooke's law to calculate spring of fit, standard deviation constant

Solution of Linear system of equations

Solution of mesh equations of electric circuits (3 meshes), by Gauss elimination method and Gauss, Seidal method. Diagonalization of Solution of coupled spring mass systems (3 masses matrices, Inverse of a matrix, Eigen vectors, eigen values problems

Generation of Special functions using User defined functions in Scilab

Generating and plotting Legendre Polynomials Generating and plotting Bessel function

Solution of ODE First order differential equation

First order Differential equation Euler, modified Euler and Runge-Kutta second order methods: Newton's law of cooling, Classical equations of motion

Second order differential equation, Fixed difference method: Second order Differential Equation Harmonic oscillator (no friction) Damped Harmonic oscillator Over damped, Critical damped,

Oscillatory, Forced Harmonic oscillator, Transient and Steady state solution

TEXT BOOKS

1. Mathematical Methods for Physics and Engineers, K.F Riley, M.P. Hobson and S. J. Bence, 3rd ed., 2006, Cambridge University Press
2. Complex Variables, A.S. Fokas & M.J. Ablowitz, 8th Ed., 2011, Cambridge Univ. Press
3. First course in complex analysis with applications, D.G. Zill and P.D. Shanahan, 1940, Jones & Bartlett

REFERENCE BOOKS:

1. Computational Physics, D.Walker, 1st Edn., 2015, Scientific International Pvt. Ltd.
2. A Guide to MATLAB, B.R. Hunt, R.L. Lipsman, J.M. Rosenberg, 2014, 3rd Edn., Cambridge University Press
3. Simulation of ODE/PDE Models with MATLAB®, OCTAVE and SCILAB: Scientific and Engineering Applications: A.V. Wouwer, P. Saucez, C.V. Fernández. 2014 Springer Scilab by example: M. Affouf 2012, ISBN: 978-1
4. Zill D.G., Shanahan P.D. (2011), A First Course in Complex Analysis with Applications, New Delhi, Jones & Bartlett.

Program no: 1

DIFFERENTIAL EQUATION USING EULER METHOD

Aim:

To write a C++ program to find the solution of differential equation using Euler method.

Program:

```
#include<iostream.h>
#include<conio.h>
float func(float x,float y)
{
return(x+y+x*y);
}
void euler(float x0,float y,float h,float x)
{while(x0<x)
{y=y+h*func(x0,y);
x0=x0+h;
}
cout<<"approximate solution at x="<<x<<"is"<<y;}
int main()
{float x0=0;
float y0=1;
float h=0.025;
float x=0.1;
clrscr();
euler(x0,y0,h,x);
getch();
return 0;
}
```

Output:

approx sol at x = 1.11167

Result:

The above program for find the solution of differential equation using Euler method has been executed and output is verified.

Program no: 2

EQUATION OF CIRCLE

Aim:

To write a C++ program to solve equation of circle.

Program:

```
#include<iostream.h>
#include<conio.h>
void equation(double x1,double y1,double r)
{
double a=-2*x1;
doule b=-2*y1;
double c=(r*r)-(x1*x1)-(y1*y1);
cout<<"x^2+("<<a<<"x)+";
cout<<"y^2+("<<b<<"y)=";
cout<<c<<".";
}
int main()
{
double x1=2,y1=-3,r=8;
 equation(x1,y1,r);
getch();
```

```
return 0;
}
```

Output:

X^2+(-4x)+y^2+(6y)=51

Result:

The above program to solving equation of circle has been executed and output is verified.

Program no: 3

NUMBER OF INTEGRAL SOLUTIONS OF K

Aim:

To write a C++ program to find the number of integral solutions of $x_1 + x_2 + x_3 \ldots\ldots\ldots x_n = k$.

Program:

```
#include<iostream.h>
#include<conio.h>
int ncr(int n,int r)
{int fac[100]={1};
for(int i=1;i<n+1;i++)
{fac[i]=fac[i-1]*i;
}
int ans=fac[n]/(fac[n-r]*fac[r]);
return ans;
}
int main()
{
int n=3;
```

```
int k=3;
clrscr();
int ans=ncr(n+k-1,k)+ncr(k-1,n-1);
cout<<ans;
getch();
return 0;
}
```

Output:

11.0

Result:

The above program to find the number of integral solutions of $x_1 + x_2 + x_3 \ldots\ldots\ldots x_n = k$ has been executed and output is verified.

Program no: 4

TRAPEZOIDAL 1/3 RULE

Aim:

To write a C++ program for Trapezoidal 1/3 rule.

Program:

```
#include<iosteram.h>
#include<conio.h>
#include<math.h>
double f(double x)
{
double a=1/(1+x*x);
return a;
}
```

```
int main()
{
int n,i;
double a,b,h,sum=0,integral;
cout<<"enter the limts of intergration,\n intial limt,a=";
cin>>a;
cout<<"final limit,b=";
cin>>b;
cout<<"enter the number of subintervals,n=";
cin>>n;
double x[10],y[10];
h=(b-a)/n;
for(i=0;i<=n;i++)
{
x[i]=a+i*h;
y[i]=f(x[i]);
}
for (i=1;i<n;i++)
{
sum=sum+h*y[i];
}
integral=h/2.0*(y[0]+y[n])+sum;
cout<<"the definite integral is"<<integral;
getch();
return 0;
}
```

Output:

Enter the limits of integration,

Initial limit,

a=0

Final limit,

b=6

Enter the number of subintervals,

n=6

The definite integral is 1.4108

Result:

The above program for Trapezoidal 1/3 rule has been executed and output is verified.

Program no: 5

GAUSS JORDAN METHOD

Aim:

To write a C++ program to solve Gauss Jordan equation.

Program:

```
#include<iostream.h>
#include<conio.h>
#include<math.h>
void main()
{
int i,j,k,n;
float x[10],a[10][10];
clrscr();
cout<<"\n enter the value of n";
cin>>n;
cout<<"\n enter the right hand side constants:";
for(i=0;i<n;i++)
```

```
{
cin>>a[i][n];
}
cout<<"\n enter the coefficients row wise";
for(i=0;i<n;i++)
{
 for(j=0;j<n;j++)
{
cin>>a[i][j];
}
}
for(k=0;k<n;k++)
{
for(i=0;i<n;i++)
{
if(i!=k)
{
for(j=k+1;j<n+1;j++)
a[i][j]=a[i][j]-(a[i][k]/a[k][k])*a[k][j];
}
}
}
cout<<"\n the solution is :";
for(i=0;i<n;i++)
{
x[i]=(a[i][n]/a[i][i]);
cout<<"x["<<i<<"]="x[i];
}
getch();
}
```

Output:

Enter the value of n

3

Enter the right hand side constants

4

9

2

Enter the coefficient row wise :

1 1 2

2 -1 3

3 -1 -1

The solution is

x[0]=1

x[1]=-1

x[2]=2

Result:

The above program to solve Gauss Jordan equation has been executed and output is verified.

Program no: 6

GAUSS SEIDAL METHOD

Aim:

To write a C++ program to solve Gauss Sedial equation.

Program:

#include<iostream.h>

```
#include<conio.h>
#include<math.h>
void main()
{
int i,j,m,n,l;
float x[10],a[10][10],b[10],c;
clrscr();
cout<<"\n enter the value of n";
cin>>n;
cout<<"enter the number of itration";
cin>>l;
cout<<"\n enter the right hand side constants:";
for(i=0;i<n;i++)
cin>>b[j];
cout<<"enter the coeffecients row wise";
for(i=0;i<n;i++)
{
x[i]=0;
for(j=0;j<n;j++)
cin>>a[i][j];
}
m=1;
line:
for(i=0;i<n;i++)
{
c=b[i];
for(j=0;j<n;j++)
{
if(i!=j)
c=c-a[i][j]*x[j];
```

```
}
x[i]=c/a[i][i];
}
m=m+1;
if(m<=1)
goto line;
else
{
cout<<"\n the solution is :";
for(i=0;i<n;i++)
cout<<"\n the solution is:";
for(i=0;i<n;i++)
cout<<"x("<<i<<")="<<x[i];
}
getch();
}
```

Output:

```
Enter the value of  n: 3
Enter the number of iterations: 100
Enter the right hand side constants
-6      -7      -8
Enter the coefficient row wise :
10      -2      -2
-1      10      -1
-1      -1      10
The solution is :
X(0)=-0.976744
X(1)=-0.896406
X(2)=-0.987315
```

Result:

The above program to solve Gauss Sedial equation has been executed and output is verified

Program no: 7

NUMBER OF SOLUTIONS OF n

Aim:

To write a C++ program to find number of solutions of $n=x+n^x$.

Program:

```
#include<iostream.h>
#include<conio.h>
int solutions(int n)
int c=0;
for(int x=0;x<=n;++x)
if(n==x+n^x)
++c;
return c;
}
int main()
{
int n=3;
cout<<solutions(n);
getch();
```

return 0;

}

Output:

4.0

Result:

The above program to find number of solutions of $n=x+n^x$ has been executed and output is verified.

Program no: 8

TRIGNOMETRIC RATIO

Aim:

To write a C++ program to find the trigonometric values for given degree.

Program:

```
#include<iostream.h>
#include<conio.h>
#include<math.h>
const double pi=3.14159265;
int main()
{
double radians,degrees;
clrscr();
cout<<"enter angle in degrees";
```

```
cin>>degrees;
radians=degrees*pi/180.0;
cout<<"values of trigonometric radians";
cout<<"\ncos(radians)="<<cos(radians);
cout<<"\nsin(radians)="<<sin(radians);
cout<<"\ntan(radians)="<<tan(radians);
getch();
return 0;
}
```

Output:

Enter angle in dgrees 30

Value of trignometrc ratios

Cos(radians)=0.866025

Sin(radians)=0.5

Tan(radians)=0.57735

Result:

The trigonometric ratio is found using C++ program and the output is verified.