Date : / /
Page No.

EXPERIMENT-1

AIM: Program for finding roots of $f(x)=0$ using Newton Raphson Method.

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <stdlib.h>
#define f(x) 3*x-cos(x)
#define g(x) 3+sin(x)

void main()
{ float x0, x1, f0, f1, g0, e;
  int step=1, N;
  clrscr();
  printf("\n Enter initial guess:\n");
  scanf("%f", &x0);
  printf("Enter tolerable error:\n");
  scanf("%f", &e);
  printf("Enter maximum iteration:\n");
  scanf("%d", &N);
  printf("\n Step \t x0 \t f(x0) \t x1 \t f(x1)");
  do { g0=g(x0);
       f0=f(x0);
       if(g0==0.0)
```

Date : / /
Page No.

```
{ printf("Mathematical Error.");
  exit(0);
}
x1 = x0 - f0/g0;
printf("%d \t %f \t %f \t %f \t %f \t \n", step, x0,
                                          x1, f1);

    x0 = x1;
    step = step + 1;
    if (step > N)
      { printf("Not Convergent.");
        exit(0);}
  f1 = f(x1);
} while (fabs(f1) > e);
printf("\n Root is: %f", x1);
getch();
```

OUTPUT:

Enter initial guess:

1

Enter tolerable error:

0.00001

Enter maximum iteration:

10

Step	x0	f(x0)	x1	f(x1)
1	1.000000	1.459698	0.620016	0.000000
2	0.620016	0.046179	0.607121	0.046179
3	0.607121	0.000068	0.607102	0.000068

Root is : 0.607102

Date : / /
Page No.

EXPERIMENT-2

AIM: Program for finding roots of f(x) = 0 by bisection method.

PROGRAM:

```
#include <stdio.h>
# include <conio.h>
# include <math.h>
# define f(x) cos(x) - x * exp(x)

Void main ()
 {
float x0, x1, x2, f0, f1, f2, e;
int step = 1;
Clrscr ();
up:
printf ("\n Enter two initial guesses: \n");
scanf ("%f %f", &x0, &x1);
printf ("Enter tolerable error : \n");
scanf ("%f", &e);
f0 = f(x0);
f1 = f(x1);
if (f0 * f1 > 0.0)
   { printf ("Incorrect Initial Guesses. \n");
     goto up: }
printf ("\n Step \t x0 \t x1 \t x2 \t f(x2) \n");
 do {
```

Date : / /
Page No.

```
x2 = (x0 + x1)/2;
f2 = f(x2);
printf("%d \t %f \t %f \t %f \t %f \n", step, x0, x2,
                                        x1, f2);
if (f0 * f2 < 0)
  { x1 = x2;
    f1 = f2; }
else
  { x0 = x2;
    f0 = f2; }
step = step + 1;
} while (fabs (f2) > e);
printf ("\n Root is : %f", x2);
getch();
}
```

OUTPUT:

Enter two initial guesses:
0
1
Enter tolerable error:
0.0001

Step	x0	x1	x2	f(x2)
1	0.000000	1.000000	0.500000	0.053222
2	0.500000	1.000000	0.750000	-0.856061
3	0.500000	0.750000	0.625000	-0.356691
4	0.500000	0.625000	0.562500	-0.141294
5	0.500000	0.562500	0.531250	-0.041512
6	0.500000	0.531250	0.515625	0.006475
7	0.515625	0.531250	0.523438	-0.017362
8	0.515625	0.523438	0.519531	-0.005404
9	0.515625	0.519531	0.517578	0.000545
10	0.517578	0.519531	0.518555	-0.002427
11	0.517578	0.518555	0.518066	-0.000940
12	0.517578	0.518066	0.517822	-0.000197
13	0.517578	0.517822	0.517700	0.000174
14	0.517700	0.517822	0.517761	-0.000012

Root is : 0.517761

Date : / /
Page No.

EXPERIMENT-3

AIM: Program for finding roots of f(x)=0 by secant method.

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <stdlib.h>
#define f(x) x*x*x - 2*x -5
void main()
{ float x0, x1, x2, f0, f1, f2, e;
  int step = 1, N;
  clrscr();
printf("\n Enter initial guesses: \n");
scanf("%f %f", &x0, &x1);
printf("Enter tolerable error: \n");
scanf("%f", &e);
printf("Enter maximum iteration: \n");
scanf("%d", &N);
printf("\n Step \t x0, \t x1 \t x2 \t f(x2) \n");
 do { f0 = f(x0);
      f1 = f(x1);
      if (f0 == f1)
       { printf("Mathematical Error.");
         exit(0);
       }
```

Date : / /
Page No.

```
x2 = x1 - (x1-x0) * f1 / (f1-f0);
f2 = f(x2);
printf("%d \t %f \t %f \t %f \t %f \n", step, x0, x1,
                                        f2);
x0 = x1;
f0 = f1;
x1 = x2;
f1 = f2;
step = step + 1;
if (step > N)
  { printf("Not convergent.");
    exit(0); }
} while (fabs(f2) > e);
printf("\n Root is: %f", x2);
getch();
}
```

OUTPUT:

Enter initial guesses:
1
2
Enter tolerable error:
0.00001
Enter maximum iteration:
10

Step	x0	x1	x2	f(x2)
1	1.000000	2.000000	2.200000	1.248001
2	2.000000	2.200000	2.088968	-0.062124
3	2.200000	2.088968	2.094233	-0.003554
4	2.088968	2.094233	2.094553	0.000012
5	2.094233	2.094553	2.094552	0.000001

Root is : 2.094552

Date : / /
Page No.

EXPERIMENT-4

AIM: To implement Lagrange's Interpolation formula.

PROGRAM:

```
#include <stdio.h>
#include<conio.h>
void main()
{
float x[100],y[100], xp, yp=0, p;
int i,j,n;
clrscr();
printf("Enter number of data:");
scanf("%d",&n);
printf("Enter data:\n");
for( i=1; i<=n ; i++)
  { printf("x[%d] =", i);
    scanf("%f",&x[i]);
    printf("y[%d] =", i);
    scanf("%f",&y[i]);
  }
printf("Enter interpolation point:");
scanf("%f",&xp);
for(i=1; i<=n; i++)
  { p=1;
    for(j=1; j<=n; j++)
      { if(i!=j)
```

Date : / /
Page No.

```
    { p = p* (xp - x[j]) / (x[i] - x[j]) ; }
  }
  yp = yp + p * y[i];
}
printf ("Interpolated value at %.3f is %.3f.", xp, yp);
getch();
}
```

OUTPUT:

```
Enter number of data : 5
Enter data :
x[1] = 5
y[1] = 150
x[2] = 7
y[2] = 392
x[3] = 11
y[3] = 1452
x[4] = 13
y[4] = 2366
x[5] = 17
y[5] = 5202
Enter interpolation point : 9
Interpolated value at 9.000 is 810.000.
```

Date : / /
Page No.

EXPERIMENT-5

AIM: To implement Newton's Divided Difference Formula.

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
void main()
{ int x[10], y[10], p[10];
 int k, f, n, i, j=1, f1=1, f2=0;
printf("\n Enter the number of observations:\n");
scanf("%d", &n);
printf("\n Enter the different values of x :\n");
for(i=1; i<=n; i++)
   { scanf("%d", &x[i]); }
printf("\n the corresponding values of y are :\n");
for(i=1; i<=n; i++)
   { scanf("%d", &y[i]);
f=y[1];
printf("\n Enter the value of 'k' in f(k) you want to
        evaluate :\n");
scanf("%d", &k);
do
{ for(i=1; i<=n-1; i++)
    { p[i]=((y[i+1]-y[i])/(x[i+j]-x[i]));
      y[i]=p[i];
    }
```

Date : / /
Page No.

```
fl = 1;
for (i = 1; i <= j; i++)
    { fl *= (k - x[i]); }
f2 += (y[j] * fl);
  n--;
  j++;
} while (n != 1);
  f += f2;
printf ("\n f(%d) = %d", k, f);
 getch();
}
```

OUTPUT

```
Enter the number of observations:
5
Enter the different values of x:
5 7 11 13 17
The corresponding values of y are:
150 392 1452 2366 5202
Enter the value of 'k' in f(k) you want to evaluate:
9
f(9) = 810
```

Date : / /
Page No.

EXPERIMENT-6

AIM: Program for solving numerical integration by Trapezoidal rule.

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define f(x) 1/(1+pow(x,2))
int main()
{ float lower, upper, integration=0.0, stepSize, k;
  int i, subInterval;
  clrscr();
  printf("Enter lower limit of integration:");
  scanf("%f", &lower);
  printf("Enter upper limit of integration:");
  scanf("%f", &upper);
  printf("Enter number of sub intervals:");
  scanf("%d", &subInterval);

stepSize = (upper - lower)/subInterval;
integration = f(lower) + f(upper);
for(i=1; i<=subInterval-1; i++)
   { k = lower + i*stepsize;
  integration = integration + 2* f(k);
  }
```

Date : / /
Page No.

```
integration = integration * stepSize / 2;
printf ("\n Required value of integration is : %.3f",
                                        integration);
getch();
return 0;
}
```

OUTPUT:

```
Enter lower limit of integration : 0
Enter upper limit of integration : 1
Enter number of sub intergration : 6
Required value of integration is : 0.784
```

Date : / /
Page No.

EXPERIMENT-7

AIM: Program for solving numerical integration by Simpson's 1/3 rule.

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define f(x) 1/(1+x*x)
int main()
{ float lower, upper, integration 0.0, stepsize, k;
  int i, subInterval;
  clrscr();
  printf("Enter lower limit of integration:");
  scanf("%f", &lower);
  printf("Enter upper limit of integration:");
  scanf("%f", &upper);
  printf("Enter number of subintervals:");
  scanf("%d", &subInterval);
  stepSize = (upper-lower)/subInterval;
  integration = f(lower)+f(upper);
  for(i=1; i<=subInterval-1; i++)
    { k=(lower + i*stepSize);
      if(i%2==0)
        {integration = integration + 2*f(k);}
      else
```

Date : / /
Page No.

```
    { integration = integration + 4*f(k); }
}
integration = integration * stepSize/3;
printf("\n Required value of integration is : %.3f",
                                        integration);
getch();
return 0;
}
```

OUTPUT:

Enter lower limit of integration : 0
Enter upper limit of integration : 1
Enter number of sub intervals : 6

Required value of integration is : 0.785

Date : / /
Page No.

EXPERIMENT-8

AIM: To implement Numerical Integration Simpson's 3/8 rule.

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define f(x) 1/(1+x*x)
int main()
{ float lower, upper, integration=0.0, stepSize, k;
  int i, subInterval;
  clrscr();
  printf("Enter lower limit of integration:");
  scanf("%f", &lower);
  printf("Enter upper limit of integration:");
  scanf("%f", &upper);
  printf("Enter number of subintervals:");
  scanf("%d", &subInterval);
  stepSize = (upper-lower)/subInterval;
  integration = f(lower) + f(upper);
  for( i=1; i<=subInterval-1; i++)
  { k = lower + i*stepsize;
    if (i%3==0)
      { integration = integration + 2*f(k); }
    else
      {integration = integration + 3*f(k);}
```

```
}
integration = integration * stepSize * 3/8;
printf("\n Required value of integration is : %.3f",
                                        integration);
getch();
return 0;
}
```

OUTPUT:

```
Enter lower limit of integration : 0
Enter upper limit of integration : 1
Enter number of sub intervals : 12

Required value of integration is : 0.785
```