Program no: 1.a

Aim: To understand the error

Software used: scilab 5.5.2

Code:

```
clear;
n=3;
es=0.5*(10^(2-n));
x=0.5;
f(1)=1;
ft=1.648721;
et(1)=(ft-f(1))*100/ft;
ea(1)=100;
i=2;
while ea(i-1)>=es
    f(i)=f(i-1)+(x^(i-1))/(factorial(i-1));
    et(i)=(ft-f(i))*100/ft;
    ea(i)=(f(i)-f(i-1))*100/f(i);
    i=i+1;
end
for j=1:i-1
    disp(ea(j),"approximate estimate of error(%)=",et(j),"true % relative
error=",f(j),"result=",j,"term number=")
    disp("to understand the errors")
end
```

Output:

```
   100.
  "approximate estimate of error(%)="
   39.346924
  "true % relative error="
   1.
  "result="
   1.
  "term number="
  "to understand the errors"

   33.333333
  "approximate estimate of error(%)="
   9.0203861
  "true % relative error="
   1.5
  "result="
   2.
  "term number="
```

"to understand the errors"

 7.6923077
"approximate estimate of error(%)="
 1.4387516
"true % relative error="
 1.625
"result="
 3.
"term number="
"to understand the errors"

 1.2658228
"approximate estimate of error(%)="
 0.1751459
"true % relative error="
 1.6458333
"result="
 4.
"term number="
"to understand the errors"

 0.1579779
"approximate estimate of error(%)="
 0.0171951
"true % relative error="
 1.6484375
"result="
 5.
"term number="
"to understand the errors"

 0.0157953
"approximate estimate of error(%)="
 0.0014001
"true % relative error="
 1.6486979
"result="
 6.
"term number="
"to understand the errors"

Program no: 1.b

Aim: To understand the error

Software used: scilab 5.5.2

Code:

```
clear;
n=3;
es=0.8*(10^(2-n));
x=0.8;
f(1)=1;
ft=1.648721;
et(1)=(ft-f(1))*100/ft;
ea(1)=100;
i=2;
while ea(i-1)>=es
    f(i)=f(i-1)+(x^(i-1))/(factorial(i-1));
    et(i)=(ft-f(i))*100/ft;
    ea(i)=(f(i)-f(i-1))*100/f(i);
    i=i+1;
end
for j=1:i-1
    disp(ea(j),"approximate estimate of error(%)=",et(j),"true % relative
error=",f(j),"result=",j,"term number=")
    disp("to understand the errors")
end
```

Output:

```
   100.
  "approximate estimate of error(%)="
   39.346924
  "true % relative error="
   1.
  "result="
   1.
  "term number="
  "to understand the errors"

   44.444444
  "approximate estimate of error(%)="
  -9.1755367
  "true % relative error="
   1.8
  "result="
   2.
```

"term number="
"to understand the errors"

 15.094340
"approximate estimate of error(%)="
-28.584521
"true % relative error="
 2.12
"result="
 3.
"term number="
"to understand the errors"

 3.8694075
"approximate estimate of error(%)="
-33.760250
"true % relative error="
 2.2053333
"result="
 4.
"term number="
"to understand the errors"

 0.7679386
"approximate estimate of error(%)="
-34.795396
"true % relative error="
 2.2224
"result="
 5.
"term number="
"to understand the errors"

 0.1227194
"approximate estimate of error(%)="
-34.961019
"true % relative error="
 2.2251307
"result="
 6.
"term number="
"to understand the errors"

 0.0163599
"approximate estimate of error(%)="
-34.983102
"true % relative error="
 2.2254948
"result="
 7.
"term number="
"to understand the errors"

Program no: 2.a

Aim: Bisection method

Software used: scilab 5.5.2

Code:

```
deff('y=f(x)','y=x^3-x-1');
x1=1,x2=2;
d=0.0001;
c=1;
printf('Successive approximations \t x1\t \tx2\t \tm\t \tf(m)\n');
while abs(x1-x2)>d
   m=(x1+x2)/2;
printf('\t%f\t%f\t%f\t%f\n',x1,x2,m,f(m));
   if f(m)*f(x1)>0
      x1=m;
   else
      x2=m;
end
c=c+1;
end
printf('the solution of equation after %i iteration is %g',c,m)
```

Output:

1.000000	2.000000	1.500000	0.875000
1.000000	1.500000	1.250000	-0.296875
1.250000	1.500000	1.375000	0.224609
1.250000	1.375000	1.312500	-0.051514
1.312500	1.375000	1.343750	0.082611
1.312500	1.343750	1.328125	0.014576
1.312500	1.328125	1.320313	-0.018711
1.320313	1.328125	1.324219	-0.002128
1.324219	1.328125	1.326172	0.006209
1.324219	1.326172	1.325195	0.002037
1.324219	1.325195	1.324707	-0.000047
1.324707	1.325195	1.324951	0.000995
1.324707	1.324951	1.324829	0.000474
1.324707	1.324829	1.324768	0.000214

```
--> printf('the solution of equation after % iteration is %g',c,m)
the solution of equation after  15teration is 1.32477
```

Program no: 2.b

Aim: Bisection method

Software used: scilab 5.5.2

Code:

```
deff('y=f(x)','y=x^3-2*x-5');
x1=2,x2=3;
d=0.0001;
c=1;
printf('Successive approximations \t x1\t \tx2\t \tm\t \tf(m)\n');
while abs(x1-x2)>d
   m=(x1+x2)/2;
printf('t%f\t%f\t%f\t%f\n',x1,x2,m,f(m));
   if f(m)*f(x1)>0
      x1=m;
   else
      x2=m;
end
c=c+1;
end
printf('the solution of equation after %i iteration is %g',c,m)
```

Output:

```
t2.000000    3.000000    2.500000    5.625000
t2.000000    2.500000    2.250000    1.890625
t2.000000    2.250000    2.125000    0.345703
t2.000000    2.125000    2.062500    -0.351318
t2.062500    2.125000    2.093750    -0.008942
t2.093750    2.125000    2.109375    0.166836
t2.093750    2.109375    2.101563    0.078562
t2.093750    2.101563    2.097656    0.034714
t2.093750    2.097656    2.095703    0.012862
t2.093750    2.095703    2.094727    0.001954
t2.093750    2.094727    2.094238    -0.003495
t2.094238    2.094727    2.094482    -0.000771
t2.094482    2.094727    2.094604    0.000592
t2.094482    2.094604    2.094543    -0.000090

--> printf('the solution of equation after %i iteration is %g',c,m)
the solution of equation after 15 iteration is 2.09454
```

Program no: 3

Aim: False position method

Software used: scilab 5.5.2

Code:

```
deff('y=f(x)','y=2*x^3-x-5');
a=2,b=3;
d=0.0001;
c=1;
printf('Successive approximations \ta\t b\t f(a)\t f(b)\t\ x1\n');
for i=1:25
   x1=b*f(a)/(f(a)-f(b))+a*f(b)/(f(b)-f(a));
   if (f(a)*f(x1))>0
      b=x1;
   else
        a=x1;
   end
      if abs(f(x1))<d
      break
   end
printf(' \t%f %f %f %f %f\n',a,b,f(a),f(b),x1);
end
printf('the root of the equation is %f',x1);
```

Output:

```
        2.000000 2.058824 -1.000000 -0.390800 2.058824
        2.096559 2.058824 0.022428 -0.390800 2.096559
        2.094511 2.058824 -0.000457 -0.390800 2.094511

--> printf('the root of the equation is %f',x1);
the root of the equation is 2.094552
```

Program no: 4

Aim: Newton-raphson method

Software used: scilab 5.5.2

Code:

```
deff('y=f(x)','y=sin(x)-x/2');
deff('y1=f1(x)','y1=cos(x)-1/2');
x0=%pi/2;
d=0.0001;
c=0;n=1;
printf('successive iterations\tx0\t f(x0)\t f1(x0)\n');
while n==1
   x2=x0;
   x1=x0-(f(x0)/f1(x0));
   x0=x1;
printf(' \t%f\t%f\t%f\n',x2,f(x1),(x1));
c=c+1;
if abs(f(x0))<d then
   break;
end
end
printf('the root of %i iteration is: %0.4g',c,x0);
```

Output:

```
        1.570796      -0.090703     2.000000
        2.000000      -0.004520     1.900996
        1.900996      -0.000014     1.895512

--> printf('the root of %i iteration is: %0.4g',c,x0);
the root of 3 iteration is: 1.896
```

Program no: 5

Aim: Secant method

Software used: scilab 5.5.2

Code:

```
function[x]=secant(a,b,f)
    N=100;
    PE=10^-4
    for n=1:1:N
        x=a-(a-b)*f(a)/(f(a)-f(b));
        if abs(f(x))<=PE then break;
        else a=b;
            b=x;
        end
    end
    disp(n,"no.of iterations=")
endfunction
    deff('[y]=f(x)','y=x^6-x-1')
    a=1
    b=2
    secant(a,b,f)
```

Output:

```
-->     a=1
 a  =
   1.
-->     b=2
 b  =
   2.
-->     secant(a,b,f)
   7.
  "no.of iterations="
 ans  =
   1.1347241
```

Program no: 6

Aim: Trapezoidal Rule

Software used: scilab 5.5.2

Code:

```
function[i] = trapezoidal(a,b,n,f)
h=(b-a)/n;
x=(a:h:b);
y=f(x);
m=length(y);
i=y(1)+y(m)
for j = 2:m-1
   i=i+2*y(j);
end;
i=(h/2)*i;
return(i);
endfunction
deff('[y]=f(x)','y=exp(x)')
trapezoidal(0,1,4,f)
```

Output:

```
 ans  =
    1.727221904557517
```

Program no: 7.a

Aim: simpsons 1/3

Software used: scilab 5.5.2

Code:

```
function[i] = simpsons13(a,b,n,f)
h=(b-a)/n;
x=(a:h:b);
y=f(x);
m=length(y);
i=y(1)+y(m);
for j = 2:m-1
    if(modulo(j,2)==0)
        i=i+4*y(j);
    else
        i=i+2*y(j);
    end;
end;
i=(h/3)*i;
return(i);
endfunction
deff('[y]=f(x)','y=exp(x)')
simpsons13(0,4,4,f)
```

Output:

```
 ans  =
    53.86384574586413
```

Program no: 7.b

Aim: simpsons 3/8

Software used: scilab 5.5.2

Code:

```
function[i] = simpsons38(a,b,n,f)
h=(b-a)/n;
x=(a:h:b);
y=f(x);
m=length(y);
i=y(1)+y(m);
for j = 2:m-1
    if(modulo(j-1,3)==0)
        i=i+2*y(j);
    else
        i=i+3*y(j);
    end;
end;
i=(3*h/8)*i;
return(i);
endfunction
deff('[y]=f(x)','y=4+2*sin(x)')
simpsons38(0,%pi,6,f)
```

Output:

```
 ans  =
    16.57039030761629
```

Program no: 8

Aim: LInear Regression

Software used: scilab 5.5.2

Code:

```
x=[1,2,3,4,5,6,7];
y=[0.5,2.5,2,4,3.5,6,5.5];
n=7;
s=0;
xsq=0;
xsum=0;
ysum=0;
for i=1:7
    s=s+(det(x(1,i)))*(det(y(1,i)));
    xsq=xsq+(det(x(1,i))^2);
```

```
    xsum=xsum+det(x(1,i));
    ysum=ysum+(det(y(1,i));
end
disp(s,"sum of product of x and y = ")
disp(xsq,"sum of square of x = ")
disp(xsum,"sum of all the x = ")
disp(ysum,"sum of all the y = ")
a=xsum/n;
b=ysum/n;
a1=(n*s-xsum*ysum)/(n*xsq-xsum^2);
a0=b-a*a1;
disp(a1,"a1 = ")
disp(a0,"a0 = ")
disp("the equation of the line obtained is y = a0+a1*x")
```

Output:

```
 sum of product of x and y =
    0.5
 -->disp(xsq,"sum of square of x = ")
 sum of square of x =
    1.
-->disp(xsum,"sum of all the x = ")
 sum of all the x =
    1.
-->disp(ysum,"sum of all the y = ")
 sum of all the y =
    0.
-->a=xsum/n;
-->b=ysum/n;
-->a1=(n*s-xsum*ysum)/(n*xsq-xsum^2);
-->a0=b-a*a1;
-->disp(a1,"a1 = ")
 a1 =
    0.583333333333333
-->disp(a0,"a0 = ")
 a0 =
  - 0.083333333333333
-->disp("the equation of the line obtained is y = a0+a1*x")
 the equation of the line obtained is y = a0+a1*x
```

Program no: 9

Aim: Runge-Kutta

Software used: scilab 5.5.2

Code:

```
deff('y=f(x,y)','y=y-x')
y=2;x=0;h=0.1;
k1=h*f(x,y);
k2=h*f(x+h,y+k1);
y1=y+(k1+k2)/2
disp('y(0.1)by second order runge kutta method: %0.4f',y1)
y=y1;x=0.1;h=0.1;
k1=h*f(x,y);
k2=h*f(x+h,y+k1);
y1=y+(k1+k2)/2
disp('y(0.2)by second order runge kutta method: %0.4f',y1)
y=2,x=0,h=0.1;
k1=h*f(x,y);
k2=h*f(x+h/2,y+k1/2);
k3=h*f(x+h/2,y+k2/2);
k4=h*f(x+h,y+k3);
y1=y+(k1+2*k2+2*k3+k4)/6;
disp('y(0.1)by fourth order runge kutta method: %0.4f',y1)
y=y1,x=0.1,h=0.1;
k1=h*f(x,y);
k2=h*f(x+h/2,y+k1/2);
k3=h*f(x+h/2,y+k2/2);
k4=h*f(x+h,y+k3);
y1=y+(k1+2*k2+2*k3+k4)/6;
disp('y(0.1)by fourth order runge kutta method: %0.4f',y1);y=2,x=0,h=0.1;
```

Output:

```
--> deff('y=f(x,y)','y=y-x')
--> y=2;x=0;h=0.1;
--> k1=h*f(x,y);
--> k2=h*f(x+h,y+k1);
--> y1=y+(k1+k2)/2
 y1  =
   2.205
--> disp('y(0.1)by second order runge kutta method: %0.4f',y1)
  "y(0.1)by second order runge kutta method: %0.4f"
   2.205
--> y=y1;x=0.1;h=0.1;
--> k1=h*f(x,y);
```

```
--> k2=h*f(x+h,y+k1);
--> y1=y+(k1+k2)/2
 y1  =
   2.421025
--> disp('y(0.2)by second order runge kutta method: %0.4f',y1)
  "y(0.2)by second order runge kutta method: %0.4f"
   2.421025
--> y=2,x=0,h=0.1;
 y  =
   2.
 x  =
   0.
--> k1=h*f(x,y);
--> k2=h*f(x+h/2,y+k1/2);
--> k3=h*f(x+h/2,y+k2/2);
--> k4=h*f(x+h,y+k3);
--> y1=y+(k1+2*k2+2*k3+k4)/6;
--> disp('y(0.1)by fourth order runge kutta method: %0.4f',y1)
  "y(0.1)by fourth order runge kutta method: %0.4f"
   2.2051708
--> y=y1,x=0.1,h=0.1;
 y  =
   2.2051708
 x  =
   0.1
--> k1=h*f(x,y);
--> k2=h*f(x+h/2,y+k1/2);
--> k3=h*f(x+h/2,y+k2/2);
--> k4=h*f(x+h,y+k3);
--> y1=y+(k1+2*k2+2*k3+k4)/6;
--> disp('y(0.1)by fourth order runge kutta method: %0.4f',y1);y=2,x=0,h=0.1;
  "y(0.1)by fourth order runge kutta method: %0.4f"
   2.4214026
 y  =
   2.
 x  =
   0.
```