Bisection :-

f[x\_]:=x^3-5x+1;

Plot[f[x],{x,-3,3}]

f[x\_]:=x^3-5x+1

a[0]=0;

b[0]=1;

Do[p[n+1]=N[(a[n]+b[n])/2];

If[N[f[a[n]]\*f[p[n+1]]]<0,a[n+1]=a[n];

b[n+1]=p[n+1],a[n+1]=p[n+1];

b[n+1]=b[n]],{n,0,20}]

Print[“n”,”a[n]”,”b[n]”,”p[n+1]”]

TableForm[Table[{n,a[n],b[n],p[n+1],f[p[n+1]]},{n,0,20}]]

Regular Falsi :-

Do[p[n+1]=b[n]-((b[n]-a[n])/(f[b[n]]-f[a[n]]))\*f[b[n]];

If[N[f[a[n]]\*f[p[n+1]]]<0,a[n+1]=a[n];

b[n+1]=p[n+1],a[n+1]=p[n+1];

b[n+1]=b[n],{n,0,20}]

Print[“n”,”a[n]”,”b[n]”,”p[n+1]”]

TableForm[Table[{n,a[n],b[n],p[n+1],f[p[n+1]]},{n,0,20}]]

Secant Method :-

Do[a[n+2]=a[n+1]-(a[n+1]-a[n])/(f[a[n+1]]-f[a[n]]) f[a[n+1]],{n,0,9}]

TableForm[Table[{n,a[n],f[a[n]]},{n,0,9}]]

Newton Raphson Method :-

f[x\_]=x^3+3x+2

Plot[f[x],{x,-3,3}]

a[0]=-0.5;

Do[a[n+1]=a[n]-((f[a[n]]/f’[a[n]]),{n,0,9}]

TableForm[Table[{n,a[n],f[a[n]]},{n,0,9}]]

Gauss Jordan :-

MatrixForm[A={{ },{ },{ }}]

MatrixForm[B={ }]

lie1=A.{Subscript[x,1],Subscript[x,2],Subscript[x,3]==B

MatrixForm[aug1=Transpose[Join[Transpose[A],{B}]]]

MatrixForm[r=RowReduce[aug1]]

x=r[[All,4]]

lie1=A.Subscript[x,1],Subscript[x,2],Subscript[x,3]}==B

Solve[lie1,{Subscript[x+1],Subscript[x,2],Subscript[x,3]}]

Gauss-Seidel :-

A = {{4.0, 1.0, 2.0}, {-3.0, 5.0, 1.0}, {1.0, 1.0, 3.0}};

d = {{4.0, 0, 0), (0, 5.0, 0), (0, 0, 3.0));

u = {{0, 1.0, 2.0}, {0, 0, 1.0}, {0, 0, 1 = 0}};

l= {{0, 0, 0}, {-3.0, 0, 0}, {1.0, 1.0, 0}}; b = Transpose[{{4.0, 7.0, 3.0}}];

x[1] Transpose[{{0, 0, 0}}]; Do[x[n+ 1] = Linear Solve[(l + d), -u.x[n]+b];

Print(x^n, "", MatrixForm[x[n]]], {n, 1, 15}]

Gauss Jacobi :-

A=((3.0, 1.0, 2.0), (-3.0, 5.0, 1.0), (1.0, 1.0, 3.0)); d={{3.0, 6, 0), (6, 5.0, 0), (0, 0, 3.0));

u = ((0, 1.0, 2.0), (0, 0, 1.0), (0, 0, 0));

1= {{0, 0, 0), (-3.0, 0, 0), (1.0, 1., 0);

b Transpose([3.0, 7.0, 3.6}}];

x[1] = Transpose[{{0, 0, 0}}];

Do]x[n+ 1] = LinearSolveld, (lu). x[n] b];

Print xn, "a", MatrixForm[stol. (n, 1, 15}]

Lagrange :-

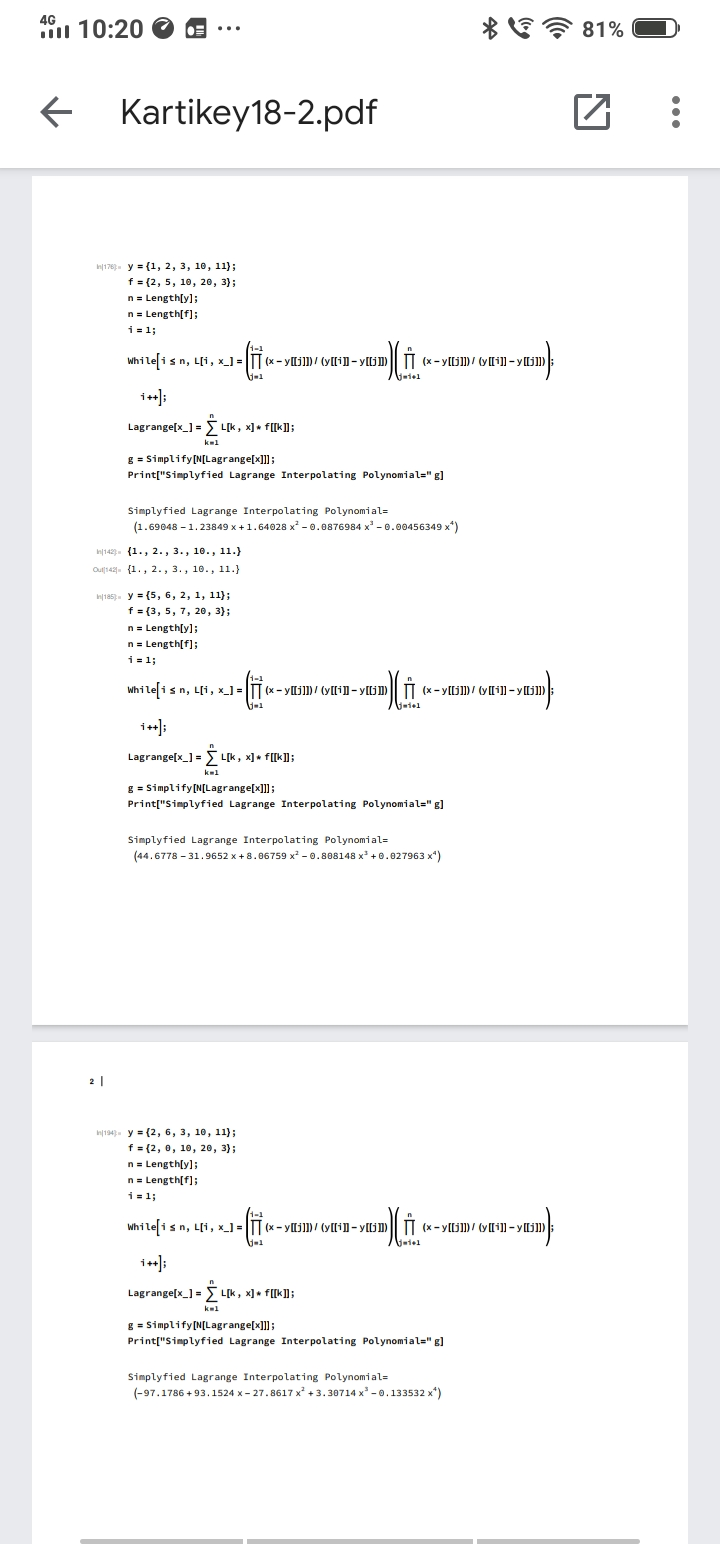
( newton aur Lagrange ke copy ni ho paye :)

y=(1, 2, 3, 10, 11);

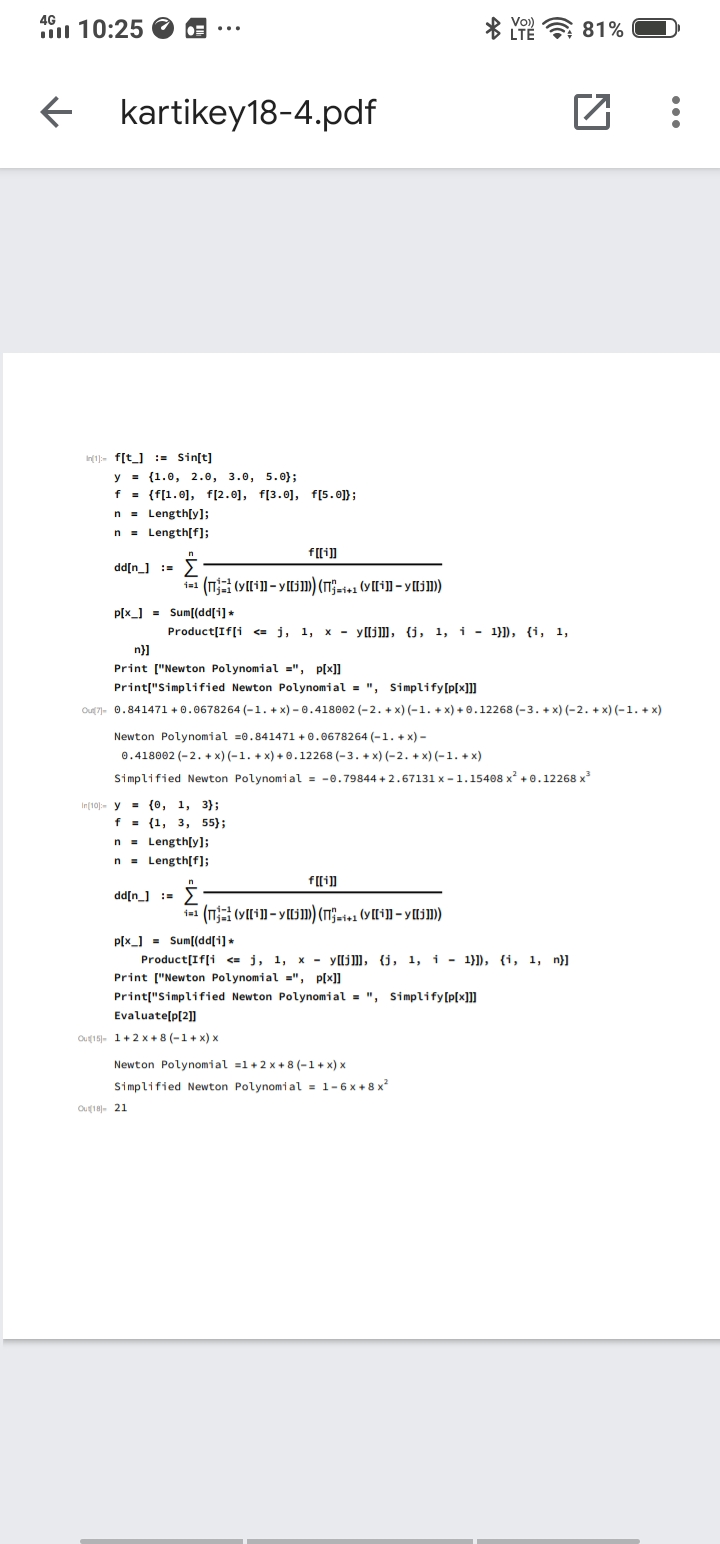
f={2, 5, 10, 20, 3);

n = Length[y];

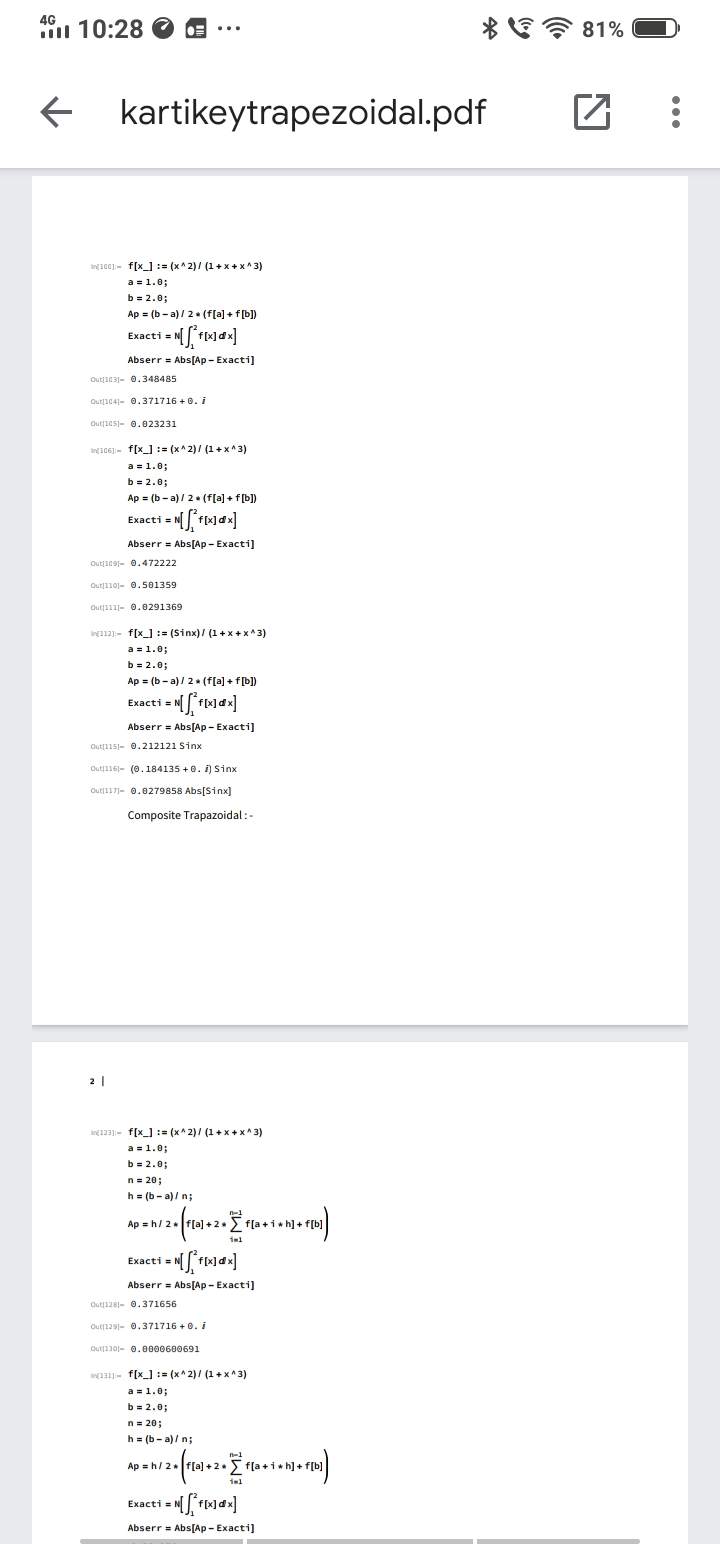
n = Length[f]; i=1;



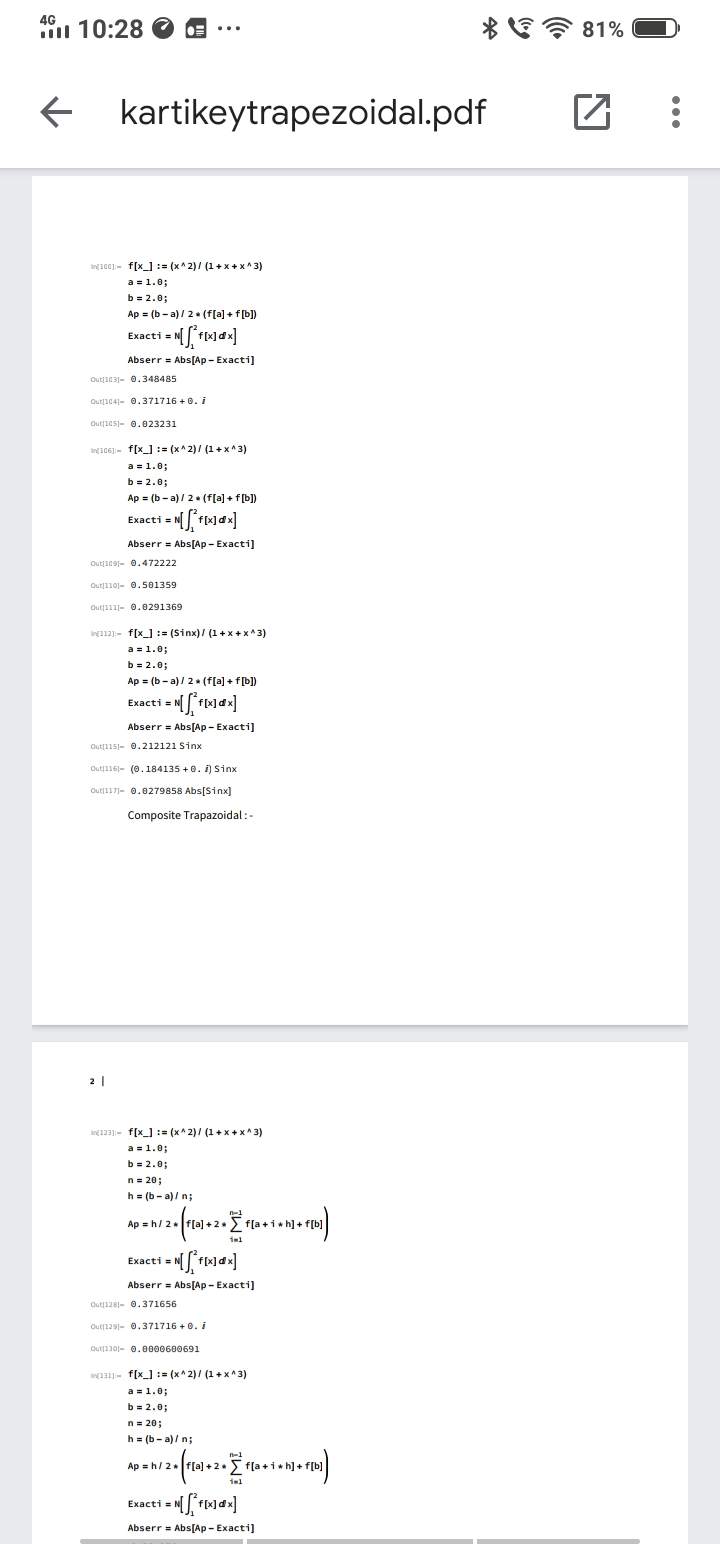
Newton Interpolating :-



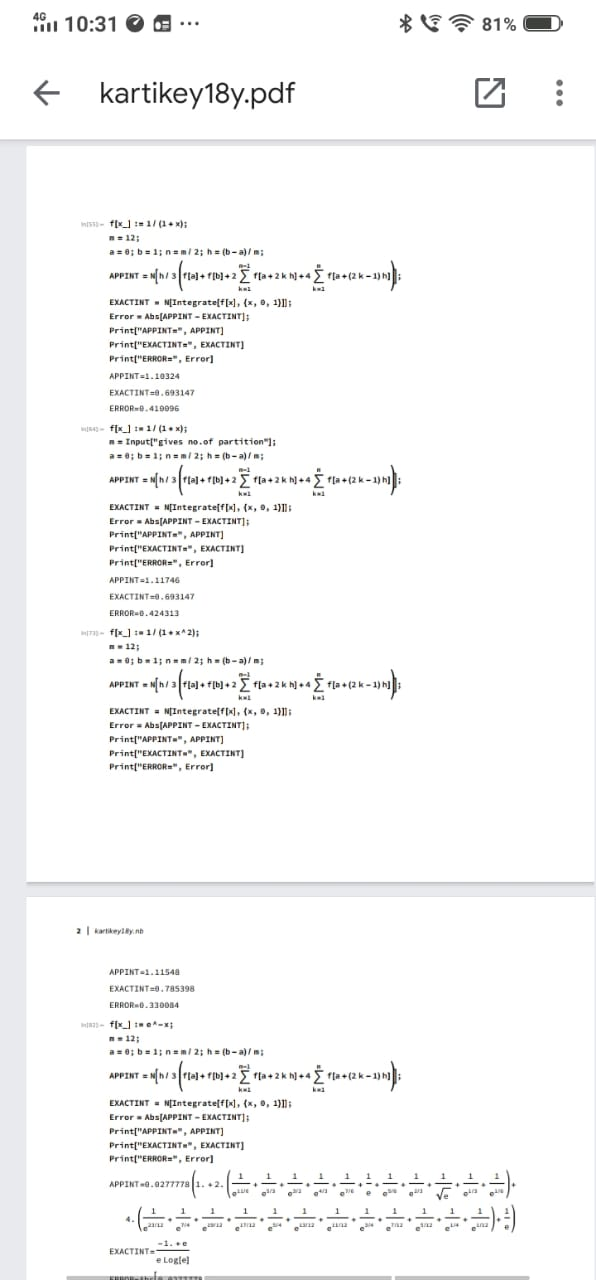
Trapezoidal :-



Composite Trapezoidal :-



Simpsons :-



Euler :-

