Computational Methods Lab

Amneesh Singh I6

December 26, 2022

Contents

1 Program for finding roots of f(x) = 0 using Newton Raphson Method **2**

2 Program for nding roots of f(x) = 0 using bisection method **2**

3 Program for finding roots of f(x) = 0 using secant method **3**

4 Program to implement Langrange Interpolation. **4**

5 Program to implement Newton's Divided Difference formula. **5**

6 Program for solving numerical integration by trapezoidal method. **6**

Programs are followed by their respective inputs and outputs i.e, both stdin and stdout are shown together

1 Program for finding roots of f(x) = 0 using Newton Raphson Method

```c
#include <math.h>
#include <stdio.h>
#define EPSILON 0.0000001
#define f(x) ((352 * x * x * x) - (64 * x * x) + (198 * x) - 36)
#define f1(x) ((1056 * x * x) - (128 * x) + 198)
double newtonRaphson(double x) {
  double h = f(x) / f1(x);
  if (f(x) == 0 || fabs(h) < EPSILON)
    return x;
  return newtonRaphson(x - h);
}
int main() {
  printf("Root for f(x) = 352x^3 - 64x^2 + 198x - 36 is %lf",
         newtonRaphson(-4));
}
```

```
Root for f(x) = 352x^3 - 64x^2 + 198x - 36 is 0.181818
```

2 Program for nding roots of f(x) = 0 using bisection method

```c
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#define EPSILON 0.0000001
#define f(x) ((352 * x * x * x) - (64 * x * x) + (198 * x) - 36)
double bisection(double a, double b) {
  double x;
  if (f(a) * f(b) > 0) {
    printf("The values of function at the respective initial guesses must have "
           "opposite signs");
    exit(1);
  }
  x = (a + b) / 2;
  if (f(x) == 0 || fabs(b - a) < EPSILON)
    return x;
  if (f(x) > 0)
    return bisection(x, b);
  return bisection(a, x);
}
int main() {
  printf("Root for f(x) = 352x^3 - 64x^2 + 198x - 36 is %lf",
         bisection(1.6, -4));
}
```

```
Root for f(x) = 352x^3 - 64x^2 + 198x - 36 is 0.181818
```

3 Program for finding roots of f(x) = 0 using secant method

```c
#include <math.h>
#include <stdio.h>
#define EPSILON 0.0000001
#define f(x) ((352 * x * x * x) - (64 * x * x) + (198 * x) - 36)
double secant(double a, double b) {
  double x1;
  x1 = (a * f(b) - b * f(a)) / (f(b) - f(a));
  if (f(x1) == 0 || fabs(a - b) < EPSILON)
    return x1;
  return secant(b, x1);
}
int main() {
  printf("Root for f(x) = 352x^3 - 64x^2 + 198x - 36 is %lf", secant(1.6, -4));
}
```

```
Root for f(x) = 352x^3 - 64x^2 + 198x - 36 is 0.181818
```

4 Program to implement Langrange Interpolation.

```c
#include <stdio.h>

int main() {
  double xp, yp = 0, p;
  int i, j, n;

  printf("Number of inputs: ");
  scanf("%d", &n);

  double x[n], y[n];
  printf("Input sample space:\n");

  for (i = 0; i < n; i++) {
    printf("x%d: ", i);
    scanf("%lf", x + i);
    printf("y%d: ", i);
    scanf("%lf", y + i);
  }

  printf("Enter interpolation point x: ");
  scanf("%lf", &xp);

  for (i = 0; i < n; i++) {
    p = 1;
    for (j = 0; j < n; j++) {
      if (i != j) {
        p *= (xp - x[j]) / (x[i] - x[j]);
      }
    }
    yp += p * y[i];
  }
  printf("Interpolated value for %lf is %lf.", xp, yp);
}
```

```
Number of inputs: 4
Input sample space:
x0: 0
y0: 2
x1: 1
y1: 3
x2: 2
y2: 12
x3: 5
y3: 147
Enter interpolation point x: 3
Interpolated value for 3.000000 is 35.000000.
```

5 Program to implement Newton's Divided Difference formula.

```c
#include <stdio.h>

int main() {
  int n, i, j = 1;
  double xp, yp, f1, f2 = 0;

  printf("Enter the number of inputs: ");
  scanf("%d", &n);

  double x[n], y[n];

  printf("Enter input values:\n");
  for (i = 0; i < n; i++) {
    printf("x%d=", i);
    scanf("%lf", x + i);
    printf("y%d=", i);
    scanf("%lf", y + i);
  }

  yp = y[0];

  printf("Enter interpolation point x: ");
  scanf("%lf", &xp);

  do {
    for (i = 0; i < n - 1; i++)
      y[i] = ((y[i + 1] - y[i]) / (x[i + j] - x[i]));

    f1 = 1;

    for (i = 0; i < j; i++) {
      f1 *= (xp - x[i]);
    }

    f2 += (y[0] * f1);
    j++;
  } while ((--n) > 1);

  yp += f2;

  printf("Interpolated value for %lf is %lf.", xp, yp);
}
```

```
Enter the number of inputs: 4
Enter input values:
x0=3
y0=9
x1=5
y1=12
x2=9
y2=666
x3=15
y3=10245
Enter interpolation point x: 999
Interpolated value for 999.000000 is 9525764925.000002.
```

6 Program for solving numerical integration by trapezoidal method.

```c
#include <math.h>
#include <stdio.h>

#define f(x) ((352 * x * x * x) - (64 * x * x) + (198 * x) - 36)

double trapezoidal_integral(double a, double b, double n) {
  double h = (b - a) / n;

  double s = (f(a) + f(b)) / 2;

  // Add the other heights

  for (int i = 1; i < n; i++)
    s += f(a + i * h);

  return s * h;
}

int main() {
  printf("The area under the curve f(x) = 352x^3 - 64x^2 + 198x - 36 from x=3 "
         "to x=4 is %lf",
         trapezoidal_integral(3, 4, 10000));
}
```

```
The area under the curve f(x) = 352x^3 - 64x^2 + 198x - 36 from x=3 to x=4 is 1425.444900
```