**PROGRAM: 10**

**Name: Kiran Kanyal**

**Roll No: 29**

**Section: A1**

**Aim: Write a program in C language to implement Newton’s Backward Interpolation method.**

**ALGORITHM:**

**START**

1. Prompt the user to input the number of observations n.
2. Prompt the user to input the value of x[i] and f(x[i]) for n observations.
3. Construct a backward difference table where each is the backward difference calculated as: Δyi​=f(xi​)−f(xi−1​)
4. Prompt the user to input the value of x for which f(x) needs to be interpolated.
5. Find the value of h and u.
   1. h = x[n-1] – x[n-2]
   2. u = (x – x[n-1]) / h, where x[n−1] is the last value of x in the table.
6. Use Newton’s Backward Interpolation formula to compute the interpolated value of f(x).
7. Print the backward difference table & the interpolated value of f(x).

**STOP**

**PROGRAM:**

#include<stdio.h>

#include<math.h>

int main(){

int n ;

printf("Enter the no. of observations: ");

scanf("%d",&n);

float x[n];

float y[n][n];

printf("Enter x and y for %d observations: \n",n);

for (int i=0; i<n; i++){

scanf("%f",&x[i]);

scanf("%f",&y[i][0]);

}

for (int j=1; j<n; j++){

for (int i=n-1; i>=j; i--){

y[i][j] = y[i][j-1]-y[i-1][j-1];

} }

printf("Backward Difference Table : \n");

for (int i=0; i<n; i++){

for (int j=0; j<=i; j++){

printf("%.2f\t",y[i][j]);

}

printf("\n");

}

float xPredict;

printf("Enter the point at which you want to find the value: ");

scanf("%f", &xPredict);

float h = fabs(x[0]-x[1]);

float u = (xPredict-x[n-1])/h; // x = a+hu calculating u

float predictedValue = y[n-1][0];

float fact = 1;

float uValue = 1;

for (int i=1; i<n; i++){

fact = fact\*i;

uValue = uValue\*(u+i-1);

predictedValue += (uValue\*y[n-1][i])/fact;

}

printf("Predicted value for %f is %f",xPredict,predictedValue);

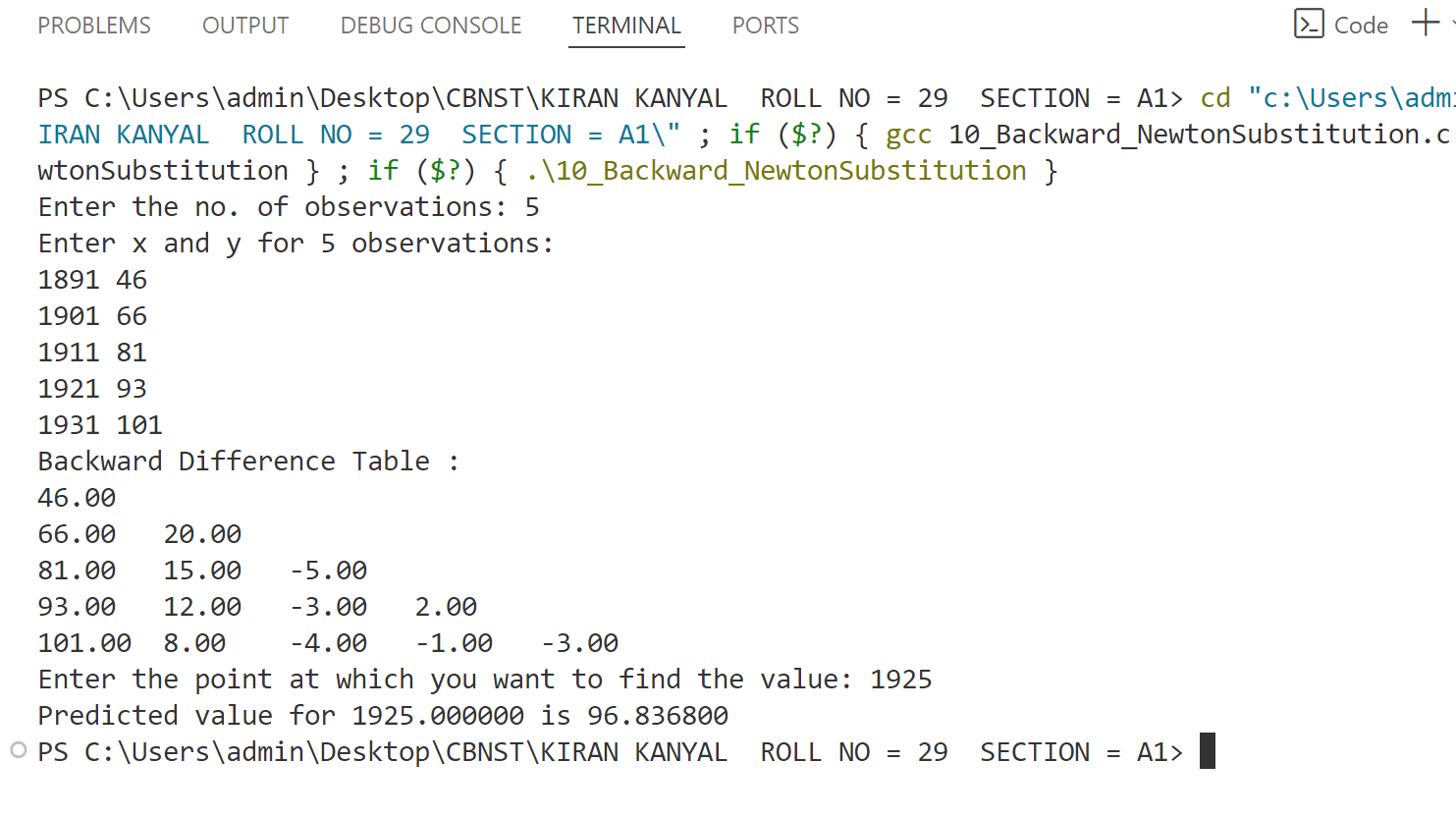
return 0; }

**OUTPUT:**

1. Find f(1925) for given table :-

x: 1891 1901 1911 1921 1931

y: 46 66 81 93 101



**PROGRAM: 11**

**Name: Kiran Kanyal**

**Roll No: 29**

**Section: A1**

**Aim: Write a program in C language to implement the Gauss Forward Interpolation method.**

**ALGORITHM:**

**START**

1. Prompt the user to input the number of observations n.
2. Prompt the user to input the value of x[i] and f(x[i]) for n observations.
3. Construct a forward difference table where each entry is the forward difference calculated as: Δyi​=f(xi​+1)−f(xi)
4. Prompt the user to input the value of x for which f(x) needs to be interpolated.
5. Find the value of h and u.
   1. h = x[1] – x[0]
   2. u = (x – x[0]) / h, where x[0] is the first value of x in the table.
6. Use Gauss’s Forward Interpolation formula to compute the interpolated value of f(x).
7. Print the forward difference table & the interpolated value of f(x).

**STOP**

**PROGRAM:**

#include <stdio.h>

int main() {

int n;

float x, y;

printf("Enter the number of data points (n): ");

scanf("%d", &n);

float arr[n][n + 1];

printf("Enter the values of x:\n");

for (int i = 0; i < n; i++) {

scanf("%f", &arr[i][0]);

}

printf("Enter the values of y:\n");

for (int i = 0; i < n; i++) {

scanf("%f", &arr[i][1]);

}

printf("Enter the value of x for which you want to find y: ");

scanf("%f", &x);

for (int j = 2; j < n + 1; j++) {

for (int i = 0; i < n - j + 1; i++) {

arr[i][j] = arr[i + 1][j - 1] - arr[i][j - 1];

}

}

printf("Difference table:\n");

for (int i = 0; i < n; i++) {

for (int j = 0; j <= n - i; j++) {

printf("%f ", arr[i][j]);

}

printf("\n");

}

float h = (arr[1][0] - arr[0][0]);

int i = 0;

while (arr[i][0] < x) {

i++;

}

i--;

float u = (x - arr[i][0]) / h;

float m = u;

if (u < 0 || u > 0.5) {

printf("Wrong data\n");

} else {

int k = 0, l = 3;

y = m \* arr[i - k][2];

for (int j = 2; j < n + 1; j++) {

if (j % 2 == 1) {

m = m \* ((u + k) / j);

} else {

k = k + 1;

m = m \* ((u - k) / j);

}

if ((i - k) < 0 || (i - k) > (n - j + 1)) {

break;

}

y = y + m \* arr[i - k][l];

l++;

}

}

y = y + arr[i][1];

printf("Value of y at x = %f is %f\n", x, y);

return 0;

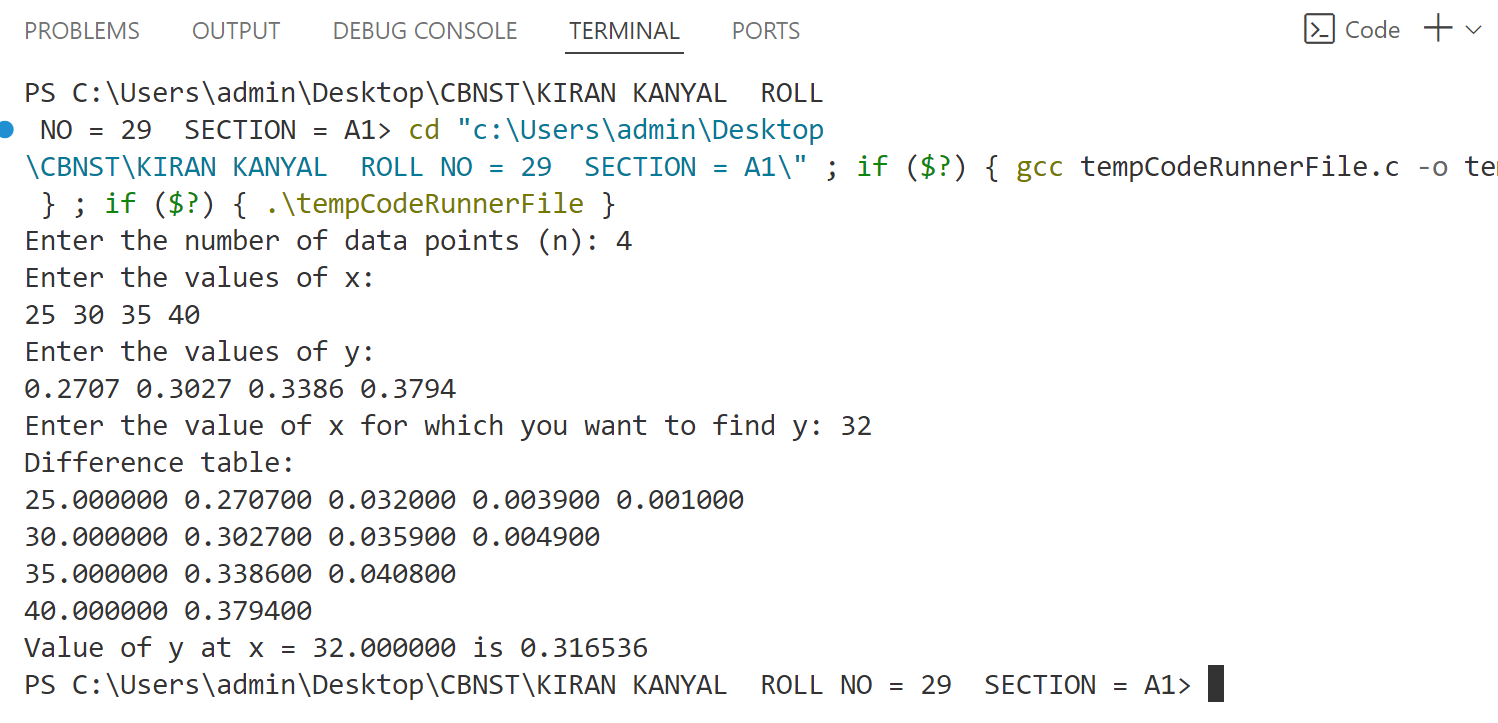
}

**OUTPUT:**

1. Find f(32) for given table :-

x: 25 30 35 40

y: 0.2707 0.3027 0.3386 0.3794



**PROGRAM: 12**

**Name: Kiran Kanyal**

**Roll No: 29**

**Section: A1**

**Aim: Write a C program to interpolate numerically using Lagrange’s method.**

**ALGORITHM:**

**START**

1. Prompt the user to input the number of observations n.
2. Prompt the user to input the value of x[i] and f(x[i]) for n observations.
3. Construct the Lagrange polynomial:
4. For each value of x[i], calculate the Lagrange basis polynomial (𝑥). Then, multiply each basis polynomial Li(x) by the corresponding f(xi), and sum them up to form the Lagrange interpolation polynomial.
5. Prompt the user to input the value of x for which f(x) needs to be interpolated.
6. Use the Lagrange interpolation formula to compute the interpolated value of f(x).
7. Print the Lagrange basis polynomials and the interpolated value of f(x)

**STOP**

**PROGRAM:**

#include<stdio.h>

int main(){

int n;

printf("Enter the number of observations: ");

scanf("%d",&n);

float x[n];

float y[n];

printf("Enter x & y for %d observations: ",n);

for (int i=0; i<n; i++){

scanf("%f%f", &x[i], &y[i]);

}

float xPredict;

printf("Enter the point at which you want to find the value: ");

scanf("%f",&xPredict);

float res = 0;

for (int i=0; i<n; i++){

float numerator = 1;

float denominator = 1;

for (int j=0; j<n; j++){

if (i==j) continue;

numerator \*= (xPredict-x[j]);

denominator \*= (x[i]-x[j]);

}

res += (numerator/denominator)\*y[i];

}

printf("Predicted Value: %f",res);

}

**OUTPUT:**

1. Find f(10) for given table :-

x: 5 6 9 11

y: 12 13 14 16

