Define Problem: Find the $y_n(x)$ polynomial of the nth degree such that y and $y_n(x)$ agree at the tabulated points .using Newton's forward or backward diffierence formula

Solution:

Newton's formula for interpolation : Given the set of (n+1) values ,viz., (x_0,y_0) ,(x_1,y_1), (x_2,y_2),…. (x_m,y_n) , of x and y, it is required to find $y_n(x)$ polynomial of the nth degree such that y and $y_n(x)$ agree at the tabulated points. Let the values of x be equidistant, i.e, Let,

$$X_i=x_0+ih \qquad i=0,1,2,3\ldots\ldots\ldots n \qquad \text{and } x=x_n+uh$$

For Newton's Forward difference formula:

$$y(x)=y_0+\frac{u}{1!}\Delta y_0+\frac{u(u-1)}{2!}\Delta^2 y_0+\ldots+\frac{u(u-1)(u-2)\ldots(u-n-1))}{n!}\Delta^n y_0 \; where\; u=\frac{x-x_0}{h}$$

For Newton's Backward difference formula:

$$y(x)=y_n+\frac{u}{1!}\Delta y_n+\frac{u(u+1)}{2!}\Delta^2 y_n+\ldots+\frac{u(u-1)(u-2)\ldots(u-\overline{n-1}))}{n!}\Delta^n y_n \; where\; u=\frac{x-x_n}{h}$$

Code:

```
#include<bits/stdc++.h>
using namespace std;

void forward(double x[],double y[][20],int n){

   int i,j;
   //Generating Forward difference Table
   for(i=1;i<n;i++){
      for(j=0;j<n-i;j++){
         y[j][i]=y[j+1][i-1]-y[j][i-1];
      }
   }

   //Display forward difference table:
            cout<<"\nForward Difference Table"<<endl;
   for(i=0;i<n;i++){
      cout<<x[i];
      for(j=0;j<n-i;j++){
         cout<<"\t"<<y[i][j];
      }
      cout<<endl;
   }

   double X,p;
      cout<<"\nEnter the value of X for forward:";
   cin>>X;
   double h=x[1]-x[0];
   double u=(X-x[0])/h;
   double sum=y[0][0];
   p=1.0;
   for(j=1;j<n;j++){
      p=p*(u-j+1)/j;
      sum+=p*y[0][j];
   }
    cout<<"The value of Y at X="<<X<<" is : "<<sum<<endl;
}

void backward(double x[],double y[][20],int n){
   int i,j;
   //Generating backward difference Table
   for(j=1;j<n;j++){
      for(i=j;i<n;i++){
         y[i][j]=y[i][j-1]-y[i-1][j-1];
      }
   }

   //Display backward difference table:
```

```
        cout<<"\nBackward Difference Table"<<endl;
    for(i=0;i<n;i++){
        cout<<x[i];
        for(j=0;j<=i;j++){
            cout<<"\t"<<y[i][j];
        }
        cout<<endl;
    }

    double X,p;
        cout<<"Enter the value of X for backward:";
    cin>>X;
    double h=x[1]-x[0];
    double u=(X-x[n-1])/h;
    double sum=y[n-1][0];
    p=1.0;
    for(j=1;j<n;j++){
        p*=(u+j-1)/j;
        sum+=p*y[n-1][j];
    }
     cout<<"\nThe value of Y at X="<<X<<" is : "<<sum<<endl;
}

int main(){
    double x[20],y[20][20];
    int i,j,n;
    //cout<<setprecision(7)<<fixed;
    //Input Section
    cout<<"Enter number of data:";
    cin>>n;
     cout<<"Enter Data: "<<endl;
    cout<<"x"<<"   "<<"y"<<endl;;
    for(i=0;i<n;i++){
        cin>>x[i]>>y[i][0];
    }
    int choice=-1;
    while(choice!=3){
        cout<<"Choose Newton's Interpolation Method:"<<endl;
            cout<<"1. Forward difference formula\n2. Backward difference Formula\n3. Exit\n";
        cout<<"Enter your choice: ";
        cin>>choice;

        switch(choice){
            case 1:
                forward(x,y,n);
                break;
            case 2:
                backward(x,y,n);
                break;
            case 3:
                cout<<"Exiting program."<<endl;
                break;
            default:
                    cout<<"Invail choice.Please try again."<<endl;
                break;
        }
        if(choice!=3){
            cout<<"\nPress Enter to continue....";
            cin.ignore();
            cin.get();
        }
    }

    return 0;
}
```

Output:

```
Enter number of data:4
Enter Data:
x  y
1 24
3 120
5 336
7 720
Choose Newton's Interpolation Method:
1. Forward difference formula
2. Backward difference Formula
3. Exit
Enter your choice: 1

Forward Difference Table
1       24      96      120     48
3       120     216     168
5       336     384
7       720

Enter the value of X for forward:8
The value of Y at X=8 is : 990

Press Enter to continue....
Choose Newton's Interpolation Method:
1. Forward difference formula
2. Backward difference Formula
3. Exit
Enter your choice: 2

Backward Difference Table
1       24
3       120     96
5       336     216     120
7       720     384     168     48
Enter the value of X for backward:8

The value of Y at X=8 is : 990

Press Enter to continue....
```

Result Analysis:

Both methods (forward and backward) return the same interpolated result, Y = 990, for X = 8. This outcome suggests that either method is appropriate for estimating the value of Y at this point, as both provide consistent results.

The forward difference method is typically more accurate if the interpolation point (here, X=8) is closer to the beginning of the data range. Conversely, the backward difference method is generally preferred if the interpolation point is closer to the end of the data range. In this case, X=8 is closest to X=7, making the backward difference method more applicable; however, both methods yield the same result here.

Conclusion:

The program demonstrates the application of Newton’s Forward and Backward Difference Interpolation methods effectively, producing accurate interpolated results.

Consistent values from both methods affirm the stability and accuracy of these approaches for this data set. Newton's Interpolation is particularly useful when the data points are evenly spaced, as seen in this example, and can yield reliable estimates for intermediate values.