Project 1 Question 1 Version 2 Documentation

1. Previous Versions

1.1 The Aim of The Program:

The aim of the program is to predict the price of an item after or for certain years based on the inputs given by the user. Those inputs given by the user are: current price of the item, the current inflation rate, current year, number of years, choice number, and specific year.

```
This program allows users to calculate price for certain items for a specific or some number of years from now
Note: Please use only appropriate numeric values as an answer for every question asked below
please enter the current price of the item
10
Please enter the inflation rate(in %)
50
Please enter the current year
2023
Please enter the number of years from now on
5
```

Fig 1.1: Shows how the program takes input from a user

1.2 The Process:

After the program takes the necessary inputs, it calculates the difference between the price of the item in successive two years and displays it to the user It calculates the difference of the price between each successive years using the difference between each values stored in successive indexes in price array (The price array is created to store the price of each year using the formula: current price+((inflation rate/100)* current price)).

```
for(int i=0; i<2;i++){
    for(int j=0; j<num_year+1;j++){
        if (i==0){
            price[i][j+1]=current_year;
            current_year=current_year +1;

        }
        else{
            price[i][j]=current_price;
            current_price=current_price+((rate_inflation/100)*current_price);
        }


    }
}
```

Fig 1.2: Shows how the price array store the price of each year for certain period of years in the code

```
//for diff array

for(int i=0;i<num_year;i++){
            diff[i]=price[1][i+1]-price[1][i];
        }

//printing diff

cout<<"The difference of successive years price for the item is"<<endl;
    for(int j=0;j<num_year;j++){
        cout<<price[0][j]<<" and "<<price[0][j+1]<<": "<<diff[j]<<endl;
    }
```

Fig 1.3: Shows the code that stores the differences of the price of successive years and display it to the user

```
The difference of successive years price for the item is
2023 and 2024: 5
2024 and 2025: 7.5
2025 and 2026: 11.25
2026 and 2027: 16.875
2027 and 2028: 25.3125
```

Fig 1.4: Shows how the differences of each successive years is displayed on the screen

After calculating and displaying the differences of each successive years to the user, the program then prompts the user choose from two choices: the price of item for specific year or for certain period of years given by the user.

```
Please enter 1 if you want to know the price of the item for specific year or enter 2 for 5 years
1. For specific year
2. For 5 years
```

Fig 1.5: Shows how the program prompts the user to choose

```
Please enter 1 if you want to know the price of the item for specific year or enter 2 for 5 y
1. For specific year
2. For 5 years
2
2023: 10
2024: 15
2025: 22.5
2026: 33.75
2027: 50.625
2028: 75.9375
Thank you for using this program!
Process returned 0 (0x0)   execution time : 3384.411 s
Press any key to continue.
```

Fig 1.6: Shows what the program displays if the user enters choice number 2

```
Please enter 1 if you want to know the price of the item for specific year or enter 2 for 5 y
1. For specific year
2. For 5 years
1
Please enter the number of years from now on you want to know the price of the item
3
Please enter the number of item
3
The price for 3 number of the item after 3 year/years from now on(in 2026) is 101.25
Thank you for using this program!
Process returned 0 (0x0)   execution time : 26.135 s
Press any key to continue.
```

Fig 1.7: Shows what the program displays if the user enters choice number 1

2. Project 1 Question 1 Version 2

2.1 New Features Added:

This updated version of Project 1 Question 1 has added and utilized the concepts of structure, functions, pointer, and header file in the previous versions which enabled the program to be more functional. The following points below describe the task of the newly added features in the code.

2.1.1 Structure

The structure in the code is used as a container to declare variables needed for a specific task as well as needed for storing the inputs listed and used in the previous versions of the project with one more added input, number of **types** of items.

```
struct items{
     string name;
     double current_price,rate_inflation,current_year, *price_ptr,*diff_ptr;
     int choice_num,num_year, spec_year,num_item,size_price;
};
```

Fig 2.1: Shows how the structure stores and declares variables

2.1.2 Functions

This newer version of the program used 4 functions that perform specific task. The first one is the “accept_input” function. It has two parameters and has no return type. It is used to accept the inputs given by the user. Refer Fig 1.1 to see how the console takes the inputs from the user. However, in addition to the previous versions input, the “accept_input” functions display the new input to the user, which is not found in the previous versions.

```
void accept_input(items item1[], int arr_size){
    for (int i=0;i<arr_size;i++){
    cout<<"Please enter the name of the item: ";
    cin>>item1[i].name;
    cout<<"please enter the current price of the item"<<endl;
    cin>>item1[i].current_price;
    cout<<"Please enter the inflation rate"<<endl;
    cin>>item1[i].rate_inflation;
    cout<<"Please enter the current year"<<endl;
    cin>>item1[i].current_year;
    cout<<"Please enter the number of years from now on  "<<endl;
    cin>>item1[i].num_year;
    item1[i].size_price=2*(item1[i].num_year+1);
    item1[i].price_ptr= new double[item1[i].size_price];
    item1[i].diff_ptr=new double[item1[i].num_year];
    cin.ignore();
    }

    order_price(item1,arr_size);

   display_diff(item1,arr_size);
   do{
    display_price(item1, arr_size);
    }while(last_choice=='y');
     cout<<"Thank you for using this program!";
    for(int i=0;i<arr_size;i++){
        delete[] item1[i].price_ptr;
      delete[] item1[i].diff_ptr;
    }

}
```

Fig 2.2: Shows how the "accept_input" takes inputs from the user

Note: The "arr_size" parameter used to accept the number of types of the items.

The second function is called "order_price". It has two parameters and no return type. Its task is to store the inputs in arrays called price_ptr and diff_ptr. These arrays store the price for each successive years and differences of them respectively using the same logic as the previous versions.

```
void order_price(items item1[],int arr_size){
    for(int i=0;i<arr_size;i++){
            int c=0;
           // int size_index=2*(item1[i].num_year+1);
        for(int j=0;j<item1[i].size_price;j++){
            if(j<item1[i].num_year+1){
                item1[i].price_ptr[j]=item1[i].current_year;
                item1[i].current_year=item1[i].current_year+1;
            }
            else{
                item1[i].price_ptr[j]=item1[i].current_price;
                item1[i].current_price=item1[i].current_price+((item1[i].rate_inflation/100)*item1[i].current_price);
            }
        }
    }

    //for difference array
    for(int i=0;i<arr_size;i++){
        for(int j=0;j<item1[i].num_year;j++){
            item1[i].diff_ptr[j]=item1[i].price_ptr[j+2+item1[i].num_year]-item1[i].price_ptr[item1[i].num_year+j+1];
        }
    }
}
```

Fig 2.3: The "order_price" function

The third and the fourth functions task is to display information to the user. The former displays the difference of the prices of each successive years, whereas the later displays the price/s of the item for/after certain number of years entered by the user. Refer figures (Fig 1.4-1.7) to see how the outputs are displayed on the console.

2.1.3 Pointers

In this version of the project, two pointers, namely "*price_ptr" and "*diff_ptr ", are used. They function as "diff" and "price" arrays in the previous versions. However, these are different in that they utilize the concept of dynamic memory management to store the differences of the prices in each successive years and prices of each year. Both the memory is allocated and deallocated in the "accept_input" function.

```
item1[i].size_price=2*(item1[i].num_year+1);
item1[i].price_ptr= new double[item1[i].size_price];
item1[i].diff_ptr=new double[item1[i].num_year];
cin.ignore();
}
```

Fig 2.4: Shows how the memory is allocated dynamically in the program

```
        for(int i=0;i<arr_size;i++){
            delete[] item1[i].price_ptr;
          delete[] item1[i].diff_ptr;
        }

    }
```

Fig 2.5: Shows how the memory is deallocated in the program

2.2 Advantages of this Version:

- The use of structure enabled the program to easily process the prices and differences for different number of types of items that was not included in the previous versions.

- The use of functions enables to debug as well as add new features easily
- The use of functions and header file increased the readability of the program
- The use of dynamic memory management system
- It loops over the results as long as the user wants to do so