## CISC2000: Computer Science II - InflationRate Part3

InflationRate3.cpp

InflationRate Part3 is going to calculate the median inflation rate, using arrays and sorting.

*Start with the code from InflationRate Part2.*

*Example: cp InflationRate2.cpp InflationRate3.cpp*

This lab is designed to provide practice of functions, call-by-reference parameters, step-wise refinement, partially filled arrays, and sorting.

Objective:

We are going to add code to calculate the Median Rate of inflation.

The median is the middle entry, where half are higher, half are lower.

If there is an odd number of elements, this is easy.

If there is an even number, we average the middle two.

However, the array must be sorted.

You might wish to print the array to be sure it is sorted. This is not part of the autograded output.

## *Directions*

*Do each part, one at a time.*

1. **Build getPCIValues**

If you have not already done it, make a function called getCPIValues to read in the two CPI values (both floats). If either value is invalid, give an error:

"Error: CPI values must be greater than 0."

…. and request new values.

1. **Create an array to accumulate the list of computed inflation rates**

a. The array should hold 20 inflation rates - also floating point.

b. Add code to main that inserts the computed inflation rate into the next position in the array.

c. Declare and implement a function called swap\_values that takes two float parameters.

1. **Add a function that sorts an array of floats. Call it sort\_array**

a. Declare and implement a function that uses either a selection sort or bubble sort to put the array values into ascending order (i.e. smallest to largest).

Using swap\_values to sort, using one of these two methods:

Bubble Sort:

Repeatedly swap mis-ordered values until none are out of order.

Selection Sort:

For each index i= 0..N, Find the minimum value and swap it with the value at A[i].

    + Function parameters: an array of floats and an int with the number of elements ACTUALLY in the array

    + Your sort function must use the swap\_values function defined above to exchange the values in the array during the sort.

 Please make a function to print each inflation rate on its own line. Call it to check that your sort works.

1. **Add a function called findMedianRate that calculates the median inflation rate using sort\_array**

a. Declare and implement a function called findMedianRate that takes two parameters and returns a float which will be the median rate.

    + parameters: an array of floats and an int with the number of elements in the array (e.g. numRates).

b. Sort the array using your sort\_array function defined above. Once the array is sorted, use the following logic to calculate the median value:

    + if the number of rates is odd, then it is the one in the middle.

    + if the number of rates is even, then the median rate is calculated as the average of the two rates in the middle.

**Expected Output:**

Enter the old and new consumer price indices: **272.35 276.50**Inflation rate is 1.52377Try again? (y or n): **y**Enter the old and new consumer price indices: **276.50 280.35**Inflation rate is 1.39241Try again? (y or n): **y**Enter the old and new consumer price indices: **280.35 285.50**Inflation rate is 1.83699Try again? (y or n): **y**Enter the old and new consumer price indices: **285.50 290.35**Inflation rate is 1.69878Try again? (y or n): **y**Enter the old and new consumer price indices: **290.35 294.55**Inflation rate is 1.44652

Try again? (y or n): **n**

Average rate is 1.57969

Median rate is 1.52377

Sorted inflation rates

1.39241

1.44652

1.52377

1.69878

1.83699

Enter the old and new consumer price indices: **272.35 276.50**Inflation rate is 1.52377Try again? (y or n): **y**Enter the old and new consumer price indices: **276.50 280.35**Inflation rate is 1.39241Try again? (y or n): **y**Enter the old and new consumer price indices: **280.35 285.50**Inflation rate is 1.83699Try again? (y or n): **y**Enter the old and new consumer price indices: **285.50 290.35**Inflation rate is 1.69878Try again? (y or n): **y**Enter the old and new consumer price indices: **290.35 294.55**Inflation rate is 1.44652Try again? (y or n): **y**Enter the old and new consumer price indices: **294.55 299.75**Inflation rate is 1.76541Try again? (y or n): **n**

Average rate is 1.61065Median rate is 1.61127Sorted inflation rates1.392411.446521.523771.698781.765411.83699

## Test Case input data (you can copy & paste)

* Odd number of Inflation Rates

**272.35 276.50y276.50 280.35y280.35 285.50y**

**285.50 290.35y290.35 294.55n**

* Even number of Inflation Rates

**272.35 276.50y276.50 280.35y280.35 285.50y285.50 290.35y290.35 294.55y294.55 299.75**

**n**

* Some Negative CPI Values

**237.844 238.323**

**y**

**-238.323 239.668**

**238.323 239.668**

**y**

**239.668 240.779**

**y**

**240.779 -241.645**

**240.779 241.645**

**n**

* Some zero CPI values

**0 239.111**

**239.111 0**

**239.111 240.222**

**n**