Elgin Allen

EBA433

Lab 1

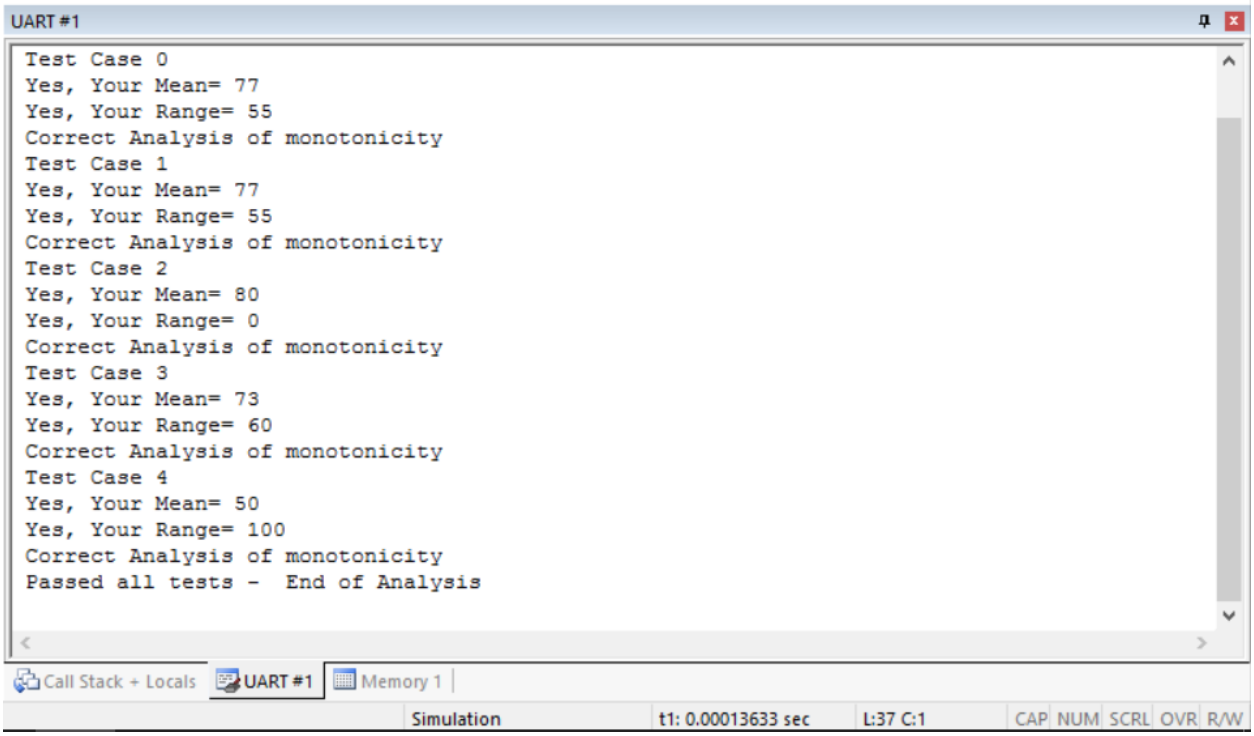
1.

```
// Return the computed Mean
uint8_t Find_Mean(){
        uint16_t sum = 0;   //holds sum, initialized to 0, needs to be 16bit
        uint8_t mean;               //holds mean
        uint8_t i = 0;              //Array counter
        while (i < N){
                sum = sum + Readings[i];      //adds current array value to sum
                i++;
        // increments counter to next array value
}
        mean = sum/N;               // usual way of calculating mean
  return(mean);
}
// Return the computed Range
uint8_t Find_Range(){
        uint8_t min = Readings[0];  //initialized as first value in array
        uint8_t max = Readings[0];  //initialized as first value in array
        uint8_t range;              //holds final range
        for (uint8_t i = 0; i < N; i++){

                if (Readings[i] < min) min = Readings[i];       //sees if current value in array is < value in min, if so it puts
                                                                array value into min
                if (Readings[i] > max) max = Readings[i];       //sees if current value in array is > value in max,
                                                                if so it puts array value into max
        }
        range = max-min;            //usual way of calculating range
  return(range);
}

// Return True of False based on whether the readings
// a non-increasing montonic series
uint8_t IsMonotonic(){
        uint8_t firstArray = 0;     //will hold the first former/first value in array (array [n])
        uint8_t secondArray = 0;    //will hold second/former value in array (array [n+1])
        uint8_t i = 0;              //array counter
        uint8_t returnValue = True; //intialize return value as true to make if else loop work
        while ( i < 21){
                if (firstArray >= secondArray){             //if the scores are monotonic, the else statement will never
be branched to, therefore the returnValue will stay True
                        firstArray = Readings[i];           // loads array [n]
                        i++;                                // array [n+1]
                        secondArray = Readings[i];  //loads array [n+1]
                }
                else
                {
                        i++;                                //increments counter inorder to exit while loop
                        returnValue = False;        //even if while loop loops back to the first if statement, the return
                                                    value is still set to False, therefore it will return non-monotonic
                                                    regardless
                }
}
  return(returnValue);
}
```

2.


UART #1
Test Case 0
Yes, Your Mean= 77
Yes, Your Range= 55
Correct Analysis of monotonicity
Test Case 1
Yes, Your Mean= 77
Yes, Your Range= 55
Correct Analysis of monotonicity
Test Case 2
Yes, Your Mean= 80
Yes, Your Range= 0
Correct Analysis of monotonicity
Test Case 3
Yes, Your Mean= 73
Yes, Your Range= 60
Correct Analysis of monotonicity
Test Case 4
Yes, Your Mean= 50
Yes, Your Range= 100
Correct Analysis of monotonicity
Passed all tests - End of Analysis
Call Stack + Locals
UART #1
Memory 1
Simulation
t1: 0.00013633 sec
L:37 C:1
CAP
NUM
SCRL
OVR
R/W