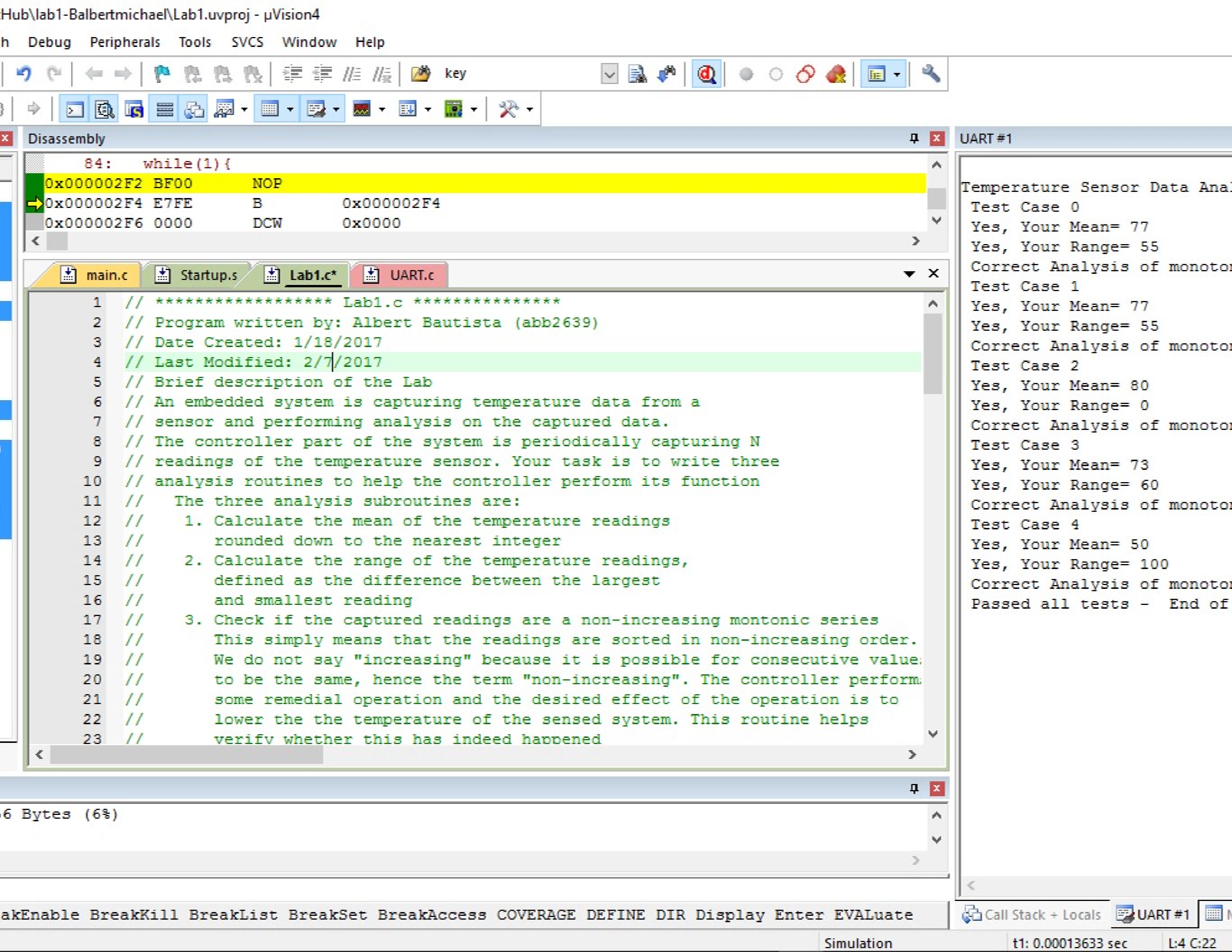

```
    84:     while(1){
0x000002F2 BF00      NOP
0x000002F4 E7FE      B          0x000002F4
0x000002F6 0000      DCW        0x0000
```

```
// ****************** Lab1.c ***************
// Program written by: Albert Bautista (abb2639)
// Date Created: 1/18/2017
// Last Modified: 2/7/2017
// Brief description of the Lab
// An embedded system is capturing temperature data from a
// sensor and performing analysis on the captured data.
// The controller part of the system is periodically capturing N
// readings of the temperature sensor. Your task is to write three
// analysis routines to help the controller perform its function
//   The three analysis subroutines are:
//    1. Calculate the mean of the temperature readings
//       rounded down to the nearest integer
//    2. Calculate the range of the temperature readings,
//       defined as the difference between the largest
//       and smallest reading
//    3. Check if the captured readings are a non-increasing montonic series
//       This simply means that the readings are sorted in non-increasing order.
//       We do not say "increasing" because it is possible for consecutive value
//       to be the same, hence the term "non-increasing". The controller perform
//       some remedial operation and the desired effect of the operation is to
//       lower the the temperature of the sensed system. This routine helps
//       verify whether this has indeed happened
```

```
Temperature Sensor Data Anal
 Test Case 0
 Yes, Your Mean= 77
 Yes, Your Range= 55
 Correct Analysis of monoto
 Test Case 1
 Yes, Your Mean= 77
 Yes, Your Range= 55
 Correct Analysis of monoto
 Test Case 2
 Yes, Your Mean= 80
 Yes, Your Range= 0
 Correct Analysis of monoto
 Test Case 3
 Yes, Your Mean= 73
 Yes, Your Range= 60
 Correct Analysis of monoto
 Test Case 4
 Yes, Your Mean= 50
 Yes, Your Range= 100
 Correct Analysis of monoto
 Passed all tests -  End of
```

```
6 Bytes (6%)
```

```
akEnable BreakKill BreakList BreakSet BreakAccess COVERAGE DEFINE DIR Display Enter EVALuate
```

```
// ***************** Lab1.c **************
// Program written by: Albert Bautista (abb2639)
// Date Created: 1/18/2017
// Last Modified: 2/7/2017
// Brief description of the Lab
// An embedded system is capturing temperature data from a
// sensor and performing analysis on the captured data.
// The controller part of the system is periodically capturing N
// readings of the temperature sensor. Your task is to write three
// analysis routines to help the controller perform its function
//   The three analysis subroutines are:
//    1. Calculate the mean of the temperature readings
//       rounded down to the nearest integer
//    2. Calculate the range of the temperature readings,
//       defined as the difference between the largest
//       and smallest reading
//    3. Check if the captured readings are a non-increasing montonic series
//       This simply means that the readings are sorted in non-increasing
order.
//       We do not say "increasing" because it is possible for consecutive
values
//       to be the same, hence the term "non-increasing". The controller
performs
//       some remedial operation and the desired effect of the operation is
to
//       lower the the temperature of the sensed system. This routine helps
//       verify whether this has indeed happened
#include <stdint.h>
#define True 1
#define False 0
#define N 21      // Number of temperature readings
uint8_t Readings[N]; // Array of temperature readings to perform analysis on

// Return the computed Mean
uint8_t Find_Mean(){
        int16_t Sum,Find_M,FM; //Initialize Variables
        Sum=0; //Make sure Sum is intialized to 0
        for( FM=0 ; FM!=N ;++FM){
                Sum = Sum + Readings[FM];      //Sums up all the Readings
values
        }
        Find_M= (Sum-(Sum%N))/N;                        //Calculates the Mean

  return(Find_M);
}

// Return the computed Range
uint8_t Find_Range(){
        uint8_t Range,Least,Greatest,FR; //Init Variables
        Least=100; //Initialize the Least to be the highest # to ensure change
        Greatest=0; //Same with Least except lowest # to ensure change
        for(FR=0 ; FR!=N ; ++FR){
                if(Least>Readings[FR]){
                        Least=Readings[FR]; //Changes Least to new lowest value
for array values [0-FR]
                }
                if(Greatest<Readings[FR]){
                        Greatest=Readings[FR]; //Changes Greatest to new
greatest value for array values [0-FR]
                }
        }
        Range = Greatest - Least;
  return(Range);
}

// Return True of False based on whether the readings
// a non-increasing montonic series
uint8_t IsMonotonic(){
// Replace ths following line with your solution
        uint8_t Monotonic,CurrVal,NextVal,IM,Debug;
        Monotonic=0;
        Debug=0;
        for(IM=0 ; IM!=(N-1) ; ++IM){
                CurrVal=Readings[IM];
                NextVal=Readings[(IM+1)]; //Looks ahead of IM for NextVal
                //Debug=++Debug; //Just to make sure I look through 20
comparisons as it looks ahead eliminating 1 for iteration
                if(NextVal>CurrVal){
                        Monotonic=0;
                        IM=(N-2); //Escapes the loop as the for loop adds 1
before the check
                }
                else{
                        Monotonic=1; //Monotonic:True for array elements[0-IM]
                }
        }
  return(Monotonic);
}


//Testcase 0:
// Scores[N] =
{80,75,73,72,90,95,65,54,89,45,60,75,72,78,90,94,85,100,54,98,75};
// Range=55 Mean=77 IsMonotonic=False
//Testcase 1:
// Scores[N] =
{100,98,95,94,90,90,89,85,80,78,75,75,75,73,72,72,65,60,54,54,45};
// Range=55 Mean=77 IsMonotonic=True
//Testcase 2:
// Scores[N] =
{80,80,80,80,80,80,80,80,80,80,80,80,80,80,80,80,80,80,80,80,80};
// Mean=80 Range=0 IsMonotonic=True
//Testcase 3:
// Scores[N] =
{100,80,40,100,80,40,100,80,40,100,80,40,100,80,40,100,80,40,100,80,40};
// Mean=73 Range=60 IsMonotonic=False
//Testcase 4:
// Scores[N] =
{100,95,90,85,80,75,70,65,60,55,50,45,40,35,30,25,20,15,10,5,0};
// Range=100 Mean=50 IsMonotonic=True
```