CS 219 – Assignment #4

Purpose: Become familiar with the MIPS instruction set and function calling conventions
Points: 100

Reading/References:

Chapter 2, 2.1-2.10
MIPS Assembly Language Programming using QtSpim, Chapters 1 - 8

Assignment:

Use the provided MIPS assembly language main program and the create the following functions:

- Write a void MIPS function, ***randomNumbers()***, to create a series of random numbers, which are then stored in an array. The address of the array and the count of numbers to generate are passed. To generate a pseudo random number, use the linear congruential generator[1] method. The next random number is generated from the previous one by:

$$R_{n+1} = \left[\left(A \times R_n + B\right) \bmod 2^{24}\right] \bmod \text{RAND\_LIMIT}$$

  The initial random number R_n (on which the rest are based on is referred to as the "seed"). The value for A must be a prime number. For our purposes, set A=127691 B=7 and **RAND_LIMIT** = 100000. The seed will be set uniquely for each set of data (in the provided main).

- Write a MIPS assembly language function, ***gnomeSort()***. The function should sort a list of numbers into ascending order (small to large). To sort the numbers, use the following Gnome sort[2] algorithm:

```
gnomeSort(a[0..size-1]) {
    i := 1
    j := 2
    while (i < size)
         if (a[i-1] <= a[i])
              i := j
              j := j + 1
         else
              swap a[i-1] and a[i]
              i := i - 1
              if (i = 0) i := 1
}
```

  The Gnome Sort is based on the technique used by Dutch gardeners to sort a line of flower pots. Basically, the gardener looks at the flower pot next to him and the previous one; if they are in the right order the gardener steps one pot forward, otherwise he swaps them and steps one pot backwards. Boundary conditions: if there is no previous pot, he steps forward; if there is no pot next to him, he is done.

  You ***must*** use the above Gnome sort algorithm (i.e., do **not** use a different sort). *Note*, the algorithm assumes array index's start at 0. As necessary, you can define additional variables. ***Submissions not based on this algorithm will not be scored***.

1 For more information, refer to: https://en.wikipedia.org/wiki/Linear_congruential_generator
2 For more information, refer to: http://en.wikipedia.org/wiki/Gnome_sort

- Write a MIPS void function, ***stats()***, that will find the minimum, median, maximum, sum, and average of the numbers array. The function is called after the list is sorted. The average should be calculated and returned as a floating point value.

Submission:

When complete, submit:

- A copy of the **source file** via the class web page by class time.
  Assignments received after the start time of class will not be accepted.

Example Output:

Below is an example of the output for the first two (of five) data sets is shown below.

```
CS 219 MIPS Assignment #4

----------------------------
Data Set #1
Length: 15

 Random Numbers:
  26136  38543  60524  72747  41056  62119  26996  38595  77736  93343
  52796   5371  18800  30487  64548

 Sorted Numbers:
   5371  18800  26136  26996  30487  38543  38595  41056  52796  60524
  62119  64548  72747  77736  93343

         Sum = 709797.00000000
     Average = 47319.80078125
     Minimum = 5371
      Median = 41056
     Maximum = 93343


----------------------------
Data Set #2
Length: 85

 Random Numbers:
  79956  20771   7880   5663  81916  11963  73936  31351  51204  61235
  16152  69615  70380  11275  66016  37863  73396  87459  19336  45567
  89500  36539  21296  95607  39044  45075  73016  28655  72332  32555
  30528  78951  24916  15395  65768  87039  79228  45627  19984   1111
  14820  38195  64728  53039  20620  46347  14752  50855  31252  20579
  61160  54783  70940  29595  98224  46935   3748  50899  56536  84559
  86348  14539  52608  15303  77748   7619  76456   7679  87004  90203
  95632  89111  28516  82099  56728  76399  74796  78763  66624  77063
  15572  24163   2856  93215  70044

 Sorted Numbers:
   1111   2856   3748   5663   7619   7679   7880  11275  11963  14539
  14752  14820  15303  15395  15572  16152  19336  19984  20579  20620
  20771  21296  24163  24916  28516  28655  29595  30528  31252  31351
  32555  36539  37863  38195  39044  45075  45567  45627  46347  46935
  50855  50899  51204  52608  53039  54783  56536  56728  61160  61235
  64728  65768  66016  66624  69615  70044  70380  70940  72332  73016
  73396  73936  74796  76399  76456  77063  77748  78763  78951  79228
  79956  81916  82099  84559  86348  87004  87039  87459  89111  89500
  90203  93215  95607  95632  98224

         Sum = 4264754.00000000
     Average = 50173.57812500
     Minimum = 1111
      Median = 51204
     Maximum = 98224
```