02/25/14 09:27:13 /Users/Clayton/Desktop/Repos/Comp-Org/Assignment 1/RieckSPIM1.a

```
 1  #-------------------------------------------------------------+
 2  # Author: Clayton Rieck                                       |
 3  #-------------------------------------------------------------+
 4  # DESCRIPTION:                                                |
 5  # This program finds the MIN, MAX, MEAN and VARIANCE          |
 6  # of given an array of up to 20 integers. It also displays    |
 7  # a bar graph of frequencies of numbers in the array.         |
 8  #-------------------------------------------------------------+
 9  # INPUT:                                                      |
10  # The user will input up to 20 integers and will end          |
11  # input once a negative value is entered. The integers        |
12  # may not be 0 and greater than 15 (Range = [1,15])           |
13  #-------------------------------------------------------------+
14  # OUTPUT:                                                     |
15  # The program will output the MIN, MAX, MEAN, and VARIANCE    |
16  # of the array of integers and will also output a bar graph   |
17  # of the frequencies of numbers in the array                  |
18  #-------------------------------------------------------------+
19      .data
20  myArray:    .word       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 # allocates 20 integer spaces in the array
21  prompt:     .asciiz     "Please enter up to 20 positive integers and a negative to end input"
22  var_div:    .asciiz     "VARIANCE: "
23  graph_div:  .asciiz     "BAR GRAPH"
24  endl:       .asciiz     "\n"
25  endl2:      .asciiz     "\n\n"
26  space:      .asciiz     ": "
27  space2:     .asciiz     ":  "
28  complete:   .asciiz     "Program complete"
29  array_err:  .asciiz     "Out of 1-15 range. Terminating program"
30  min:        .asciiz     "MIN: "
31  max:        .asciiz     "MAX: "
32  mean:       .asciiz     "MEAN: "
33      .text
34      .globl main
35
36  main:
37      # ask user to input numbers
38      la      $a1,    myArray             # load pointer into a1
39      la      $a0,    prompt              # load prompt into a0
40
41      li      $v0,    4                   # print out prompt
42      syscall
43
44      la      $a0,    endl
45      li      $v0,    4
46      syscall
47
48      li      $t0,    0                   # MIN
49      li      $t1,    0                   # MAX
50      li      $t2,    0                   # ARRAY OFFSET
51      li      $t3,    0                   # ARRAY LENGTH
```

```
52      li      $t5,    0                   # SUM
53
54  #-----------------------------------------------------
55  # Starts the input prompt and stores numbers in array \
56  # while also computing/finding the MIN, MAX, SUM and   \
57  # ARRAY LENGTH                                         \
58  #-----------------------------------------------------
59  gather_numbers:
60
61      bgt     $t3,    20,     three_ms        # if we've filled our list
62
63      li      $v0,    5                   # read integer into $v0
64      syscall
65
66      move    $t4,    $v0                 # $t4 holds the value that
    the user inputted
67
68      beq     $t4,    0,      array_error # if enter a number less than
    1
69      bgt     $t4,    15,     array_error # if enter a number above 15
70
71      bltz    $t4,    three_ms            # if enter negative then stop
    populating list
72
73      sw      $t4,    myArray($t2)        # store current element in
    $t4
74      add     $t3,    $t3,    1           # add 1 to array length
75
76      add     $t5,    $t5,    $t4         # add input number to sum
77
78      add     $t2,    $t2,    4           # go to next index in array
79
80      beqz    $t0,    set_min             # initialize min with first
    number entered
81
82      blt     $t4,    $t0,    set_min     # if input less than MIN, set
    new min
83
84  #-----------------------------------------------------
85  # Checks to see if input is bigger than current MAX   \
86  #-----------------------------------------------------
87  check_max:
88      bgt     $t4,    $t1,     set_max
89      j       gather_numbers
90
91  #-----------------------------------------------------
92  # Sets current MAX to inputted number                \
93  #-----------------------------------------------------
94  set_max:
95      move     $t1,    $t4
96      j        gather_numbers
97
98  #-----------------------------------------------------
99  # Checks to see if input is smaller than current MIN  \
100 #-----------------------------------------------------
101 set_min:
```

```
102     move     $t0,    $t4
103     j        check_max
104
105 #---------------------------------------------------
106 # Prints out the MAX, MIN, and MEAN of the inputted   \
107 # integers in the array                               \
108 #---------------------------------------------------
109 three_ms:
110
111     la      $a0,    endl
112     li      $v0,     4
113     syscall
114
115     # print out MIN
116     la      $a0,    min
117     li      $v0,     4
118     syscall
119
120     li      $v0,     1
121     addi    $a0,    $t0,    0
122     syscall
123
124     la      $a0,    endl
125     li      $v0,    4
126     syscall
127
128     # print out MAX
129     la      $a0,    max
130     li      $v0,    4
131     syscall
132
133     li      $v0,    1
134     addi    $a0,    $t1,     0
135     syscall
136
137     # print out MEAN
138     la      $a0,    endl
139     li      $v0,     4
140     syscall
141
142     la      $a0,    mean
143     li      $v0,     4
144     syscall
145
146     # calculates and then prints mean
147     li      $v0,    1
148     div     $t5,    $t3             # divide sum of numbers by length
    of array
149     mflo    $a0
150     syscall
151
152     la      $a0,    endl2
153     li      $v0,    4
154     syscall
```

```
155
156 # ---------------------------------------------------------
157 # Start Standard Deviation
158
159     li      $t2,    0                 # array offset
160     li      $t6,    1                 # current index in the array
161     li      $s0,    0                 # result of summation
162     li      $s3,    1                 # CONSTANT 1 (used for 1/(n-1))
163
164     div     $t5,    $t3               # calculate x bar (mean)
165     mflo    $t0                       # store mean in $t0
166
167 #---------------------------------------------------
168 # Loops through array and subtracts each element by   \
169 # the MEAN and then sqaures the result                 \
170 #---------------------------------------------------
171 sum:
172
173     bgt     $t6,    $t3, variance   # if current index is greater
    than length of list
174
175     lw      $t4,    myArray($t2)    # load current element
176
177     sub     $t1,    $t4,    $t0     # subtract list element by mean
178     mul     $t1,    $t1,    $t1     # square difference
179     add     $s0,    $s0,    $t1     # add terms of summation
180
181     add     $t6,    $t6,    1       # increment index by 1
182     add     $t2,    $t2,    4       # increment offset by 4
183     j       sum
184
185 #---------------------------------------------------
186 # Takes the result of the summation (from above) and  \
187 # multiplies that by 1/(n-1) giving us s^2 (VARIANCE)  \
188 #---------------------------------------------------
189 variance:
190     sub     $s1,    $t3,    1       # get n - 1
191     div     $s3,    $s1             # do 1/(n-1)
192     mflo    $s1                     # store answer in $s1
193
194     mul     $s1,    $s1,    $s0     # multiply 1/(n-1) with summation
195
196     # print out variance
197     la      $a0,    var_div
198     li      $v0,    4
199     syscall
200
201     move    $a0,    $s1             # move vairance to $a0 for
    printing
202     li      $v0,    1
203     syscall
204
205     la      $a0,    endl2
206     li      $v0,    4
207     syscall
208
```

```
209 # ---------------------------------------------------------
210 # Start Bar Graph
211
212 #---------------------------------------------------
213 # Initializes counters for the bar graph (range number \
214 # and offset constant for finding end index of the array\
215 #---------------------------------------------------
216 set_graph_counter:
217     la      $a0,    graph_div
218     li      $v0,    4
219     syscall
220
221     la      $a0,    endl
222     li      $v0,    4
223     syscall
224
225     li      $t6,    1               # keep counter for bar graph
    range
226     li      $s0,    4               # used as constant for finding
    end index of array
227
228 #---------------------------------------------------
229 # Does 1 iteration through the array for each number  \
230 # the 1-15 range our integers are within               \
231 #---------------------------------------------------
232 graph_loop:
233
234     bgt     $t6,    15,     finish  # if at end of our range
235
236     li      $t2,    0               # reset array index counter to 0
237     li      $t7,    0               # keep track of number of
    ocurrences
238
239     # print out number in range
240     li      $v0,    1
241     add     $a0,    $t6,    0
242     syscall
243
244     blt     $t6,    10,      double_space # if number in range > 9
245     # print a ': ' after number in range
246     la      $a0,    space
247     li      $v0,    4
248     syscall
249     j       set_graph
250
251 #---------------------------------------------------
252 # Formats bar graph output to do 2 spaces so that     \
253 # the bars line up for the single digits and double    \
254 # digits                                                \
255 #---------------------------------------------------
256 double_space:
257     la      $a0,    space2
258     li      $v0,    4
259     syscall
260
```

```
261 #-------------------------------------------------------
262 # Conducts checks on if a number was encountered in   \
263 # the array and if on the last index of the array.     \
264 # Also jumps to beginning of loop                       \
265 #-------------------------------------------------------
266 set_graph:
267
268     lw      $t4,    myArray($t2)    # load current element
269
270     beq     $t4,    $t6,    add_bar # if current item in array equals
    number
271                                     # we're checking go to add_bar
    label
272     add     $t2,    $t2,    4       # loop through list
273
274     div     $t2,    $s0             # divide the array offset by 4 to
    get current index number
275     mflo    $s1                        # move quotient to $s1
276     beq     $s1,    $t3,    increment_number # if the array length
    and quotient are
277                                             # equal then we are at
    the end of the list and
278                                             # need to start again
279     j       set_graph
280
281 #-------------------------------------------------------
282 # Outputs a 0 next to a number denoting that it was   \
283 # found                                                \
284 #-------------------------------------------------------
285 add_bar:
286
287     # print out 0 for bar
288     li      $v0,    1
289     li      $a0,    0
290     syscall
291
292
293     add     $t2,    $t2,    4       # loop through list
294
295     # check to see if at end of the list (same as above)
296     div     $t2,    $s0
297     mflo    $s1
298     beq     $s1,    $t3,    increment_number
299
300     j       set_graph
301
302 #-------------------------------------------------------
303 # Increments the range number by 1                     \
304 #-------------------------------------------------------
305 increment_number:
306
307     addi    $t6,    $t6,    1        # increment range number by 1
308
309     # new line for new bar
```

```
310     la      $a0,    endl
311     li      $v0,    4
312     syscall
313
314     j       graph_loop
315
316 #---------------------------------------------------
317 # Outputs a program completed message               \
318 #---------------------------------------------------
319 finish:
320
321     la      $a0,    endl2           # 2 new line characters
322     li      $v0,    4
323     syscall
324
325     la      $a0,    complete        # ending message
326     li      $v0,    4
327     syscall
328
329     li      $v0,    10
330     syscall
331
332 #---------------------------------------------------
333 # Displays an error when user enters number outside   \
334 # of 1-15 range                                        \
335 #-----------------------------------------------------
336 array_error:
337     la      $a0,    endl
338     li      $v0,    4
339     syscall

340
341     la      $a0,    array_err
342     li      $v0,    4
343     syscall
344
345     li      $v0,    10
346     syscall
```