04/03/14 09:52:30 /Users/Clayton/Desktop/Repos/Comp-Org/Assignment 2/RieckSPIM2.a

```
 1  #-------------------------------------------------------------+
 2  # Author: Clayton Rieck                                       |
 3  #-------------------------------------------------------------+
 4  # DESCRIPTION:                                                |
 5  # This program finds the total cost of using a gas pump at    |
 6  # Cowberland Farms and rounds the result to the nearest       |
 7  # penny.                                                      |
 8  #-------------------------------------------------------------+
 9  # INPUT:                                                      |
10  # The user will input their payment type (S for SmahtPay,     |
11  # C for Credit or Q for Quitting). Then they'll enter a type |
12  # of gas they want to fill their car with (R for regular,     |
13  # P for Premium, or S for Super). Lastly, they'll enter the   |
14  # number of gallons they'd like to fill as a single           |
15  # precision floating point number.                            |
16  #-------------------------------------------------------------+
17  # OUTPUT:                                                     |
18  # The program will output a receipt including the payment     |
19  # type chosen, the type of gas chosen, the number of gallons |
20  # they wanted to fill as a single precision floating point    |
21  # number, the price per gallon as a single precision float,   |
22  # and the total cost rounded to the nearest penny and         |
23  # truncated to only 2 decimal places.                         |
24  #-------------------------------------------------------------+
25      .data
26  integer:        .word       0 # used to store converted total cost
27  done_prompt:    .asciiz     "\nThank you for choosing Cowberland Farms! Have
    a great day!"
28  endl:           .asciiz     "\n"
29      .text
30      .globl main
31
32  main:
33
34  #-----------------------------------------------------
35  # Begins while loop that continues until 'Q' or 'q'   \
36  # is typed                                             \
37  #-----------------------------------------------------
38  transaction_start:
39
40      jal     getPaymentAndType
41
42      # arguments peing passed to getGasAmount procedure
43      mov.s $f12,     $f0 # amount of gallons user wants
44      mov.s $f13,     $f1 # price per gallon
45
46      jal     getGasAmount
47
48      jal     printReceipt
49
50      j       transaction_start
51
```

```
52  done:
53
54      la    $a0,    done_prompt
55
56      li    $v0,    4
57      syscall
58
59      li    $v0,    10
60      syscall
61
62  # ____________________________________________________
63  #                                                    \
64  # NAME:      getPaymentAndType                        \
65  # ARGUMENTS: None                                      |
66  # RETURN:    1) Number of gallons user wants          |
67  #            2) Price per gallon                      |
68  # PURPOSE:   Takes in how many gallons a user         |
69  #            wants to fill and payment type           |
70  #            and determines the price based           |
71  #            on payment type. It also applies         |
72  #            a discount of 10 cents to                |
73  #            SmahtPay users                           |
74  # ___________________________________________________|
75      .data
76  pay_prompt:           .asciiz    "Please enter a payment type\n's':
    SmahtPay\n'c': Credit\n'q': Complete Transaction\n> "
77  gas_prompt:           .asciiz    "Please choose gas type:\nType
    Code    Price\nRegular      R      3.619\nPlus         P

    3.719\nSuper        S      3.839\n> "
78  gas_amount_prompt:    .asciiz    "\nPlease enter how many gallons of gas
    you'd like:\n> "
79      .text
80
81  getPaymentAndType:
82
83      # ascii values for 'Q' and 'q' respectively
84      li    $t0,    81
85      li    $t1,    113
86
87      # ascii values for 'S' and 's' respectively
88      li    $t2,    83
89      li    $t3,    115
90
91      # ascii values for 'C' and 'c' respectively
92      li    $t4,    67
93      li    $t5,    99
94
95      # display payment prompt
96      la    $a0,    pay_prompt
97
98      li    $v0,    4
99      syscall
100
```

```
101     # read in a character
102     li    $v0,    12
103     syscall
104
105     # move character read in into $s0
106     move  $s0,    $v0
107
108     # check if input is 'Q' or 'q'
109     beq    $s0,    $t0,    done
110     beq    $s0,    $t1,    done
111
112     la    $a0,    endl
113
114     li    $v0,    4
115     syscall
116
117     # display payment prompt
118     la    $a0,    gas_prompt
119
120     li    $v0,    4
121     syscall
122
123     # read in a character
124     li    $v0,    12
125     syscall
126
127     move  $s1,    $v0
128
129     # s0 = payment type
130     # s1 = gas type
131
132     # store price of gas type
133     # 'S' already stored in $t2
134     li    $t6,    82    # ascii value for 'R'
135     li    $t7,    80    # ascii value for 'P'
136
137     beq   $s1,    $t6,    regular    # get price for regular gas
138     beq   $s1,    $t7,    plus       # get price for plus gas
139     beq   $s1,    $t2,    super      # get price for super gas
140
141     # set the price depending on the gas the user chose
142
143 #-------------------------------------------------------
144 # If Regular, set price to 3.619                       \
145 #-------------------------------------------------------
146 regular:
147     li.s  $f21,   3.619
148     j     amount
149
150 #-------------------------------------------------------
151 # If Plus, set price to 3.719                          \
152 #-------------------------------------------------------
153 plus:
154     li.s  $f21,   3.719
```

```
155     j     amount
156
157 #-----------------------------------------------------

158 # If Super, set price to 3.839                       \
159 #-------------------------------------------------------
160 super:

161     li.s  $f21,   3.839
162     j     amount
163
164 #-------------------------------------------------------
165 # Checks if the user is using SmahtPay and if they are \
166 # then the program reduces the price per gallon by 10   \
167 # cents                                                  \
168 #---------------------------------------------------------
169 amount:
170
171     # check if not using SmahtPay
172     beq   $s0,    $t4,    no_discount
173     beq   $s0,    $t5,    no_discount
174
175     li.s  $f24,    0.1
176
177     # give 10 cent discount per gallon if using SmahtPay
178     sub.s $f21,   $f21,    $f24
179
180 #-------------------------------------------------------
181 # Takes in the amount of gallons they want to fill    \
182 #-------------------------------------------------------
183 no_discount:
184     # ask for the amount of gas they want
185     la    $a0,    gas_amount_prompt
186
187     li    $v0,    4
188     syscall
189
190     # read in float for gallons
191     li    $v0,    6
192     syscall
193
194     mov.s   $f1,    $f21
195
196     # f0 = gallons of gas user wants
197     # f1 = price per gallon

198     # $f0 and $f1 get passed back as a return values
199
200     j    $ra
201
202 # _____________________________________________

203 #                                              \
204 # NAME:      getGasAmount                       \
205 # ARGUMENTS: 1) Number of gallons user wants   |
206 #            2) Price per gallon               |
```

```
206 #            2) Price per gallon                    |
207 # RETURN:    None                                   |
208 # PURPOSE:   Takes in how many gallons a user       |
209 #            wants to fill and the price from       |
210 #            the getPaymentAndType procedure.       |
211 #            The total price is calculated          |
212 #            and then rounded to the nearest        |
213 #            while also truncating extra            |
214 #            decimal spots.                         |
215 # _____________________________________________|
216     .data
217 dot:         .asciiz        "."
218     .text
219
220 getGasAmount:
221
222     mov.s    $f20,    $f0
223     mov.s    $f21,    $f1
224
225     #f20 = gallons
226     #f21 = price
227
228     la    $a0,    endl
229     li    $v0,    4
230     syscall
231
232     # ascii values for 'S' and 's' respectively
233     li    $t2,    83
234     li    $t3,    115
235
236     # multiply gallons and price to get total
237     mul.s $f22,   $f21,    $f20
238
239     # f22 = price (NOT ROUNDED)
240
241     # --------- start truncating decimal points ---------
242
243     # multiply by 100.0 to set up for integer
244     # representation of price
245     li.s  $f5,    100.0
246     mul.s $f22,   $f22,    $f5
247
248     # add 0.5 for rounding to nearest penny
249     li.s  $f5,    0.5
250     add.s $f22,   $f22,    $f5
251
252     cvt.w.s $f22,  $f22 # convert single precision to int
253
254     # store float in a word
255     s.s   $f22,    integer
256
257     # load that new integer back into an int register
258     lw    $t0,     integer
259
260     # divide the integer by 100 and get
```

```
260     # divide the integer by 100 and get
261     # left hand side of total price in LO
262     # and right hand side in HI
263     li    $t1,    100
264     div   $t0,    $t1
265
266     mflo  $s4 # move LO to s4
267     mfhi  $s5 # move HI to s5
268
269     # $s4 = left hand side of decimal
270     # $s5 = right hand side of decimal
271
272     j     $ra
273
274 # ____________________________________________
275 #                                            \
276 # NAME:      printReceipt                     \
277 # ARGUMENTS: None                              |
278 # RETURN:    None                              |
279 # PURPOSE:   Outputs the resulting receipt     |
280 #            onto the screen using the info    |
281 #            inputted and calculated in the    |
282 #            previous 2 procedures.            |
283 # ____________________________________________|
284         .data
285 padding1:       .asciiz         " "
286 padding2:       .asciiz         "  "
287 padding4:       .asciiz         "    "
288 padding8:       .asciiz         "        "
289 title:          .asciiz         "Cowberland Farms"
290 gasAndPayType:  .asciiz         "Payment Type        Gas Type"
291 gallons:        .asciiz         "Gallon(s)"
292 costPerGallon:  .asciiz         "$/Gallons"
293 total:          .asciiz         "Total"
294         .text
295
296 printReceipt:
297
298     la    $a0,    padding8
299     li    $v0,    4
300     syscall
301
302     la    $a0,    title
303     li    $v0,    4
304     syscall
305
306     la    $a0,    endl
307     li    $v0,    4
308     syscall
309
310     la    $a0,    endl
311     li    $v0,    4
312     syscall
313
314     la    $a0,    padding2
```

```
314     la    $a0,    padding2
315     li    $v0,    4
316     syscall
317
318     la    $a0,    gasAndPayType
319     li    $v0,    4
320     syscall
321
322     la    $a0,    endl
323     li    $v0,    4
324     syscall
325
326     la    $a0,    padding4
327     li    $v0,    4
328     syscall
329
330     la    $a0,    padding2
331     li    $v0,    4
332     syscall
333
334     # print out payment type
335     move  $a0,    $s0
336     li    $v0,    11
337     syscall
338
339     la    $a0,    padding8
340     li    $v0,    4
341     syscall
342
343     la    $a0,    padding8
344     li    $v0,    4
345     syscall
346
347     la    $a0,    padding4
348     li    $v0,    4
349     syscall
350
351     # print out gas type
352     move  $a0,    $s1
353     li    $v0,    11
354     syscall
355
356     la    $a0,    endl
357     li    $v0,    4
358     syscall
359
360     la    $a0,    endl
361     li    $v0,    4
362     syscall
363
364     la    $a0,    padding2
365     li    $v0,    4
366     syscall
367
```

```
368     la    $a0,    gallons
369     li    $v0,    4
370     syscall
371
372     la    $a0,    padding8
373     li    $v0,    4
374     syscall
375
376     la    $a0,    padding2
377     li    $v0,    4
378     syscall
379
380     la    $a0,    padding1
381     li    $v0,    4
382     syscall
383
384     la    $a0,    costPerGallon
385     li    $v0,    4
386     syscall
387
388     la    $a0,    endl
389     li    $v0,    4
390     syscall
391
392     la    $a0,    padding1
393     li    $v0,    4
394     syscall
395
396     # just print a float because the number
397     # of gallons are already in $f12
398     li    $v0,    2
399     syscall
400
401     la    $a0,    padding8
402     li    $v0,    4

403     syscall
404
405     la    $a0,    padding2
406     li    $v0,    4
407     syscall
408
409     la    $a0,    padding1
410     li    $v0,    4
411     syscall

412
413     # moves price/gallon to $f12 to be
414     # printed out
415     mov.s $f12,    $f13
416     li    $v0,    2
417     syscall
418
419     la    $a0,    endl
420     li    $v0,    4
```

```
421     syscall
422
423     la    $a0,    endl
424     li    $v0,    4
425     syscall
426
427     la    $a0,    padding8
428     li    $v0,    4
429     syscall
430
431     la    $a0,    padding4
432     li    $v0,    4
433     syscall
434
435     la    $a0,    padding2
436     li    $v0,    4
437     syscall
438
439     la    $a0,    total
440     li    $v0,    4
441     syscall
442
443     la    $a0,    endl
444     li    $v0,    4
445     syscall
446
447     la    $a0,    padding8
448     li    $v0,    4
449     syscall
450
451     la    $a0,    padding4
452     li    $v0,    4
453     syscall
454
455     la    $a0,    padding2
456     li    $v0,    4
457     syscall
458
459     move  $a0,   $s4
460     li    $v0,    1
461     syscall
462
463     la    $a0,    dot
464     li    $v0,    4
465     syscall
466
467     # checks if price is 0.0
468     # if it is then add another 0 to right hand side
469     # else, print remainder
470     bne   $s5,    $zero,     not_zero
471
472     li    $a0,    0
473     li    $v0,    1
474     syscall
```

```
475
476 #-------------------------------------------------------
477 # Acts as a branch when the remainder is not 0        \
478 #-------------------------------------------------------
479 not_zero:
480
481     # check if remainder is less than or
482     # equal to 10
483     li    $t1,   10
484     bgt   $s5,   $t1,    no_zero_before
485
486     # puts an extra 0 before remainder
487     # if remainder less than 10
488     li    $a0,    0
489     li    $v0,    1
490     syscall
491
492 #-------------------------------------------------------
493 # Acts as a branch when the remainder is less than 10 \
494 #-------------------------------------------------------
495 no_zero_before:
496
497     move  $a0,   $s5
498     li    $v0,    1
499     syscall
500
501     la    $a0,    endl
502     li    $v0,    4
503     syscall
504
505     la    $a0,    endl
506     li    $v0,    4
507     syscall
508
509     j    $ra
```