```
1:  #Sawyer Fenwick 6005011
2:  #COSC 2P12 Assign_3
3:  #PART D
4:  #This function reads data off a text file, then finds and prints the floating point nu
mbers
5:  #that are on the file.
6:
7:      .data
8: fileName:    .asciiz "file.txt"
9: error1:      .asciiz "Error: File Failed To Open"
10: FILE:       .word
11: buffer:     .space 1000
12: zero:       .asciiz "0"
13: nine:       .asciiz "9"
14:         .text
15: main:
16:     la $a0 fileName     #Call the file open procedure
17:     jal openfile
18:     sb $v0, FILE
19:
20:     #read the entire contents of the file to buffer
21:     li $v0, 14
22:     lb $a0, FILE
23:     la $a1, buffer
24:     li $a2, 999
25:     syscall
26:     #close file
27:     li $v0, 16
28:     lb $a0, FILE
29:     syscall
30:     #print file
31:     la $a0, buffer
32:     li $v0, 4
33:     syscall
34:     #print newline
35:     li $a0, 10
36:     li $v0, 11
37:     syscall
38:
39:     li $t8, 0   #set t8 to 0 (counter)
40:     li $t5, 0   #set t5 to 0 (n)
41:
42:     la $t2, buffer  #load t2 with the first char of the buffer
43:     jal getint  #get the first integer
44: nextint:
45:     beqz $v1, end   #test v1 to see if we could read a digit
46:
47:     jal convert #convert the integer back to a float
48:     li $t5, 0   #reset n
49:     li $t8, 0   #reset i
50:     jal getint  #get the next integer
```

```
51:     b nextint   #loop
52: end:
53:     li $v0, 10  #syscall exit
54:     syscall
55: getint:
56:     lb $t7, nine        #Loads t7 with the character nine
57:     lb $t3, zero        #Loads t3 with the character zero
58:
59:     move $t9, $0        # $t9 is used as a flag if we can at least read 1 digit
60:     move $t6, $0        #clear out register
61:
62:     lb $t1, ($t2)       #get first char of string
63: skipLead:
64:     beqz $t1, exit      #detects a char 0 and exits.
65:     bgt $t1, $t7, nextChar  #if char is not a digit
66:     blt $t1, $t3, nextChar  #skip and read next char
67:     b endLead               #we found a digit
68: nextChar:
69:         add $t2, $t2, 1     #inc pointer into i/p string

70:     lb $t1, ($t2)       #read the char
71:     b skipLead      #restart loop
72: endLead:
73:



74: loop:
75:     li $t9, 1           #Here is the flag we set if we read at least 1 digit
76:     sub $t4, $t1, $t3       #convert that char to a number by subtracting char zero fr
om it.
77:     add $t2, $t2, 1         #increment pointer into i/p string
78:
79:     mul $t6, $t6, 10    #convert to integer  10xvalue + last_digit
80:     add $t6, $t6, $t4
81:
82:     lb $t1, ($t2)       #get next char
83:
84:     beq $t1, 32, skip   #space was read before a decimal, number is true integer and n
ot a float
85:     beq $t1, 46, next   #if t1 is a decimal weve found a float, skip the decimal and a
dd the rest of the number to our integer (convert later)
86:     bgt $t1, $t7, exit  #only stay in the loop if next char
87:     blt $t1, $t3, exit  #is a digit
88:
89:     b loop          #do it again
90: skip:
91:     j getint        #get the next integer
92: next:
93:     add $t2, $t2, 1     #increment pointer
94:     lb $t1, ($t2)       #get next char
95:     bgt $t1, $t7, exit  #continue if next char is a digit
```

```
96:     blt $t1, $t3, exit
97: count:
98:     li $t9, 1       #same as above "loop", only added the incrementer for n and remove
d the beq checks. Only care about when the
99:     sub $t4, $t1, $t3   #number ends since we know it is a float
100:    add $t2, $t2, 1
101:    addi $t5, $t5, 1    #increment n by 1
102:
103:    mul $t6, $t6, 10    #convert to integer  10xvalue + last_digit
104:    add $t6, $t6, $t4
105:
106:    lb $t1, ($t2)       #get next char
107:    bgt $t1, $t7, exit  #only stay in the loop if next char
108:    blt $t1, $t3, exit  #is a digit
109:
110:    b count         #do it again
111: exit:
112:    move $t1, $t6       #move integer to return register t1
113:    move $v1, $t9       #move Flag to return register v1
114:    jr $ra
115: convert:
116:    mtc1 $t1, $f1       #move t1 to a floating point register
117:    cvt.s.w $f1, $f1    #convert it into a single precision
118:
119:    li $t1, 10      #move 10 into t1
120:    mtc1 $t1, $f2       #move t1 to a floating point register
121:    cvt.s.w $f2, $f2    #convert it into a single precision 10.0
122: convertloop:
123:    beq $t5, $t8, print #if n = i, we are done dividing, print the float
124:    addi $t8, $t8, 1    #increment counter i++
125:
126:    div.s $f1, $f1, $f2 #divide our integer by 10.0
127:
128:    b convertloop       #loop
129: print:
130:    mov.s $f12, $f1     #move the float into f12 for printing
131:    li $v0, 2       #syscall print_float
132:    syscall
133:
134:    li $a0, 32      #set a0 to space char
135:    li $v0, 11      #syscall print_char
136:    syscall
137:
138:    jr $ra          #return
139: openfile:
140:    #Open a file for read only
141:    la $a0, fileName        #name of file to open
142:    li $a1, 0               #read only
143:    li $a2, 0           #mode is ignored
144:    li $v0, 13
145:    syscall
```

```
146:    move $s1, $v0
147:
148:    #Test if the file was open
149:    bgez $v0, skiperror
150:    la $a0, error1
151:    li $v0, 4
152:    syscall
153: skiperror:
154:    move $v0, $s1       #set s1 to v0
155:    jr $ra          #return
```