**IT Systems & Networks Assessed Exercise 1**

Sean Davies

**2243317D**

I managed to complete the exercise: the program performed as expected on a number of test cases. It will be subject to the limitation of overflow, so the size of the variable possum and/or negcount is limited by what a 16-bit two’s complement code can represent. No other limitations or bugs were found for the cases I tried during the testing: all input cases returned the expected outputs. The “acceptance test” data resulted in 4 for negcount and 109 for possum which is as expected. A sample of the test data and corresponding output (converted from hexadecimal) from the program follows

x={3, -6, 9, 14, -4} possum=26 negcount=2

x={4, 5, 17, 23, 10} possum=59 negcount=0

x={-1, -1, -1, -1, -1, -1, -1} possum=0 negcount=7

x={-1, -1, -1, -1, 34} possum=34 negcount=4

x={6, 8, 7, 12, -2, -20, -3} possum=33 negcount=3

It should be noted that n was altered to match the size of the array in each case. All these results are as expected for the data presented.

**Source Code**

;boolean loopactive=n>i  
;while(loopactive){  
; if(x[i]>0) {negcount++}  
; else {possum=possum+x[i]}  
; i++}  
;the following allocates register values and initial values

LEA R1,1[R0] ;R1=1  
 LEA R2,0[R0] ;R2=i  
 LEA R3,0[R0] ;R3=possum  
 LOAD R4,n[R0] ;R4=n   
 LEA R5,0[R0] ;R5=negcount  
  
;the following includes the while loop and if-else clause

while LOAD R7,x[R2] ;R7=x[i]  
 CMPGT R9,R0,R7 ;R9=(0>x[i])  
 JUMPT R9,else[R0] ;jump to the line that increments negcount if x[i] is negative  
 ADD R3,R3,R7 ;possum=possum+x[i]  
 JUMP done[R0] ;jump past the line that increments negcount  
else ADD R5,R5,R1 ;negcount=negcount+1  
done ADD R2,R2,R1 ;i=i+1  
 CMPGT R6,R4,R2 ;R6=(n>i)  
 JUMPT R6,while[R0] ; jump back to the beginning of the loop if the counter is less than n

;the following stores the final values of possum and negcount upon loop completion

STORE R3,possum[R0] ;possum=R3  
 STORE R5,negcount[R0] ;negcount=R5  
 TRAP R0,R0,R0 ;termination

;data statements

n DATA 11 ;n specifies the number of iterations the loop will go through  
x DATA 3  
 DATA -6  
 DATA 27  
 DATA 50  
 DATA 0  
 DATA -20  
 DATA -21  
 DATA 19  
 DATA 6  
 DATA 4  
 DATA -10  
possum DATA 0  
negcount DATA 0

**Assembly Listing**

Line Addr Code Source statement

1 0000 ;boolean loopactive=n>i  
2 0000 ;while(loopactive)  
3 0000 ;{  
4 0000 ;  
5 0000 ; if(x[i]>0)  
6 0000 ; {  
7 0000 ; negcount++  
8 0000 ; }  
9 0000 ; else  
10 0000 ; {  
11 0000 ; possum=possum+temp  
12 0000 ; }  
13 0000 ; i++  
14 0000 ;}  
15 0000 ;the following block allocates register values and initial values  
16 0000   
17 0000 f100 0001 LEA R1,1[R0] ;R1=1  
18 0002 f200 0000 LEA R2,0[R0] ;R2=i  
19 0004 f300 0000 LEA R3,0[R0] ;R3=possum  
20 0006 f401 001c LOAD R4,n[R0] ;R4=n   
21 0008 f500 0000 LEA R5,0[R0] ;R5=negcount  
22 000a   
23 000a ;the following block includes the while loop and if-else clause  
24 000a   
25 000a f721 001d while LOAD R7,x[R2] ;R7=x[i]  
26 000c 6907 CMPGT R9,R0,R7 ;R9=(0>R7)  
27 000d f905 0012 JUMPT R9,else[R0] ;jump to the piece of code that increments negcount if x[i] is negative  
28 000f 0337 ADD R3,R3,R7 ;possum=possum+x[i]  
29 0010 f003 0013 JUMP done[R0] ;jump past the piece of code that increments negcount  
30 0012 0551 else ADD R5,R5,R1 ;negcount=negcount+1  
31 0013 0212 done ADD R2,R1,R2 ;i=i+1  
32 0014 6642 CMPGT R6,R4,R2 ;R6=(n>i)  
33 0015 f605 000a JUMPT R6,while[R0] ;jump back to the beginning of the loop if the counter is less than n  
34 0017   
35 0017 f302 0028 STORE R3,possum[R0] ;possum=R3  
36 0019 f502 0029 STORE R5,negcount[R0] ;negcount=R5  
37 001b d000 TRAP R0,R0,R0 ;termination  
38 001c   
39 001c ;data statements  
40 001c   
41 001c 000b n DATA 11 ;n specifies the number of iterations the loop will go through  
42 001d 0003 x DATA 3  
43 001e fffa DATA -6  
44 001f 001b DATA 27  
45 0020 0032 DATA 50  
46 0021 0000 DATA 0  
47 0022 ffec DATA -20  
48 0023 ffeb DATA -21  
49 0024 0013 DATA 19  
50 0025 0006 DATA 6  
51 0026 0004 DATA 4  
52 0027 fff6 DATA -10  
53 0028 0000 possum DATA 0  
54 0029 0000 negcount DATA 0

Addr Symbol Def Usage

000a while [25] []  
0012 else [30] []  
0013 done [31] []  
001c n [41] []  
001d x [42] []  
0028 possum [53] []  
0029 negcount [54] []