Homework 3

Brandon Hosley

Mike Davis

Homework 3

1a

What is the output of the following PEP 8 program.
-202
99
cat

1b

Explain how each of the 5 outputs are produced from the 4 inputs:
The fourth input is a .WORD input of 2 bytes, and is comprised of the 'at' ASCII.

2a

Cut & paste the Assembler Listing including the symbol table at the bottom.

```
-------------------------------------------------------
Object
Addr  code   Symbol   Mnemon  Operand     Comment
------------------------------------------------------
0000  040007          BR      three
0003  0010   one:     .WORD   16
0005  0016   two:     .WORD   0x0016
;
0007  390003 three:   DECO    one,d
000A  50000A          CHARO   '\n',i
000D  390005          DECO    two,d
0010  50000A          CHARO   '\n',i
0013  390007          DECO    three,d
0016  00              STOP
0017                  .END
-------------------------------------------------------


Symbol table
--------------------------------------
Symbol    Value        Symbol    Value
--------------------------------------
one       0003         three     0007
two       0005
--------------------------------------
```

2b

Explain the values of the symbols one, two, and three in the symbol table.

The values described in the symbol table are addresses of the data that each symbol describes, or in the case of symbol three, the place in which the imperative part of the program will begin.

2c

Explain the values of the output of DECO one, DECO two & DECO three

DECO one prints 16_{10} represented as 16_{10} or 10_{16}

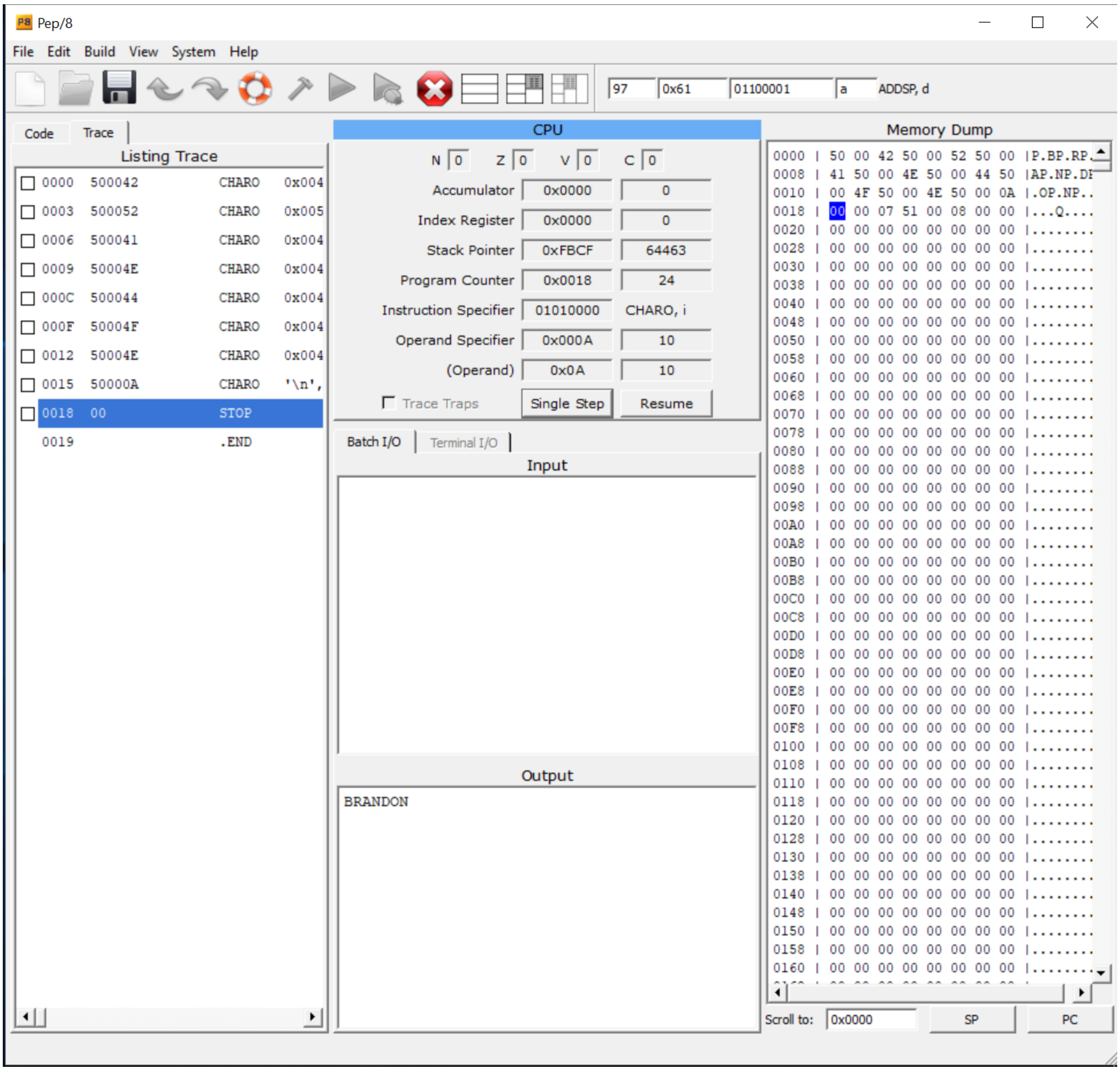
DECO two prints 22_{10} represented as 16_{16}

DECO three looks at the word stored at the location of symbol three. What it finds is the first 'print' machine code instruction. The instruction 39 00 03 becomes 3900_{16} and is printed as 14592_{10}

3

Write an assembly language program that prints your first name on the screen. Use immediate addressing with a hexadecimal constant to designate the operand of CHARO for each letter of your name. Comment each line except STOP & .END. Cut & paste the Assembler Listing into your document and paste a screen shot of the Output area of the PEP/8.

```
-------------------------------------------------------------
Object
Addr  code    Symbol   Mnemon  Operand      Comment
-------------------------------------------------------------
0000  500042           CHARO   0x0042,i     ;Print B
0003  500052           CHARO   0x0052,i     ;Print R
0006  500041           CHARO   0x0041,i     ;Print A
0009  50004E           CHARO   0x004E,i     ;Print N
000C  500044           CHARO   0x0044,i     ;Print D
000F  50004F           CHARO   0x004F,i     ;Print O
0012  50004E           CHARO   0x004E,i     ;Print N
0015  50000A           CHARO   '\n',i       ;Newline
0018  00               STOP
0019                   .END
--------------------------------------------------------------
```



Listing Trace

```
□ 0000  500042   CHARO  0x004
□ 0003  500052   CHARO  0x005
□ 0006  500041   CHARO  0x004
□ 0009  50004E   CHARO  0x004
□ 000C  500044   CHARO  0x004
□ 000F  50004F   CHARO  0x004
□ 0012  50004E   CHARO  0x004
□ 0015  50000A   CHARO  '\n',
□ 0018  00       STOP
  0019           .END
```

CPU

N 0 Z 0 V 0 C 0

Accumulator	0x0000	0
Index Register	0x0000	0
Stack Pointer	0xFBCF	64463
Program Counter	0x0018	24
Instruction Specifier	01010000	CHARO, i
Operand Specifier	0x000A	10
(Operand)	0x0A	10

□ Trace Traps

Input

Output

```
BRANDON
```

Memory Dump

```
0000 | 50 00 42 50 00 52 50 00 |P.BP.RP.
0008 | 41 50 00 4E 50 00 44 50 |AP.NP.DP
0010 | 00 4F 50 00 4E 50 00 0A |.OP.NP..
0018 | 00 00 07 51 00 08 00 00 |...Q....
0020 | 00 00 00 00 00 00 00 00 |........
0028 | 00 00 00 00 00 00 00 00 |........
0030 | 00 00 00 00 00 00 00 00 |........
0038 | 00 00 00 00 00 00 00 00 |........
0040 | 00 00 00 00 00 00 00 00 |........
0048 | 00 00 00 00 00 00 00 00 |........
0050 | 00 00 00 00 00 00 00 00 |........
0058 | 00 00 00 00 00 00 00 00 |........
0060 | 00 00 00 00 00 00 00 00 |........
0068 | 00 00 00 00 00 00 00 00 |........
0070 | 00 00 00 00 00 00 00 00 |........
0078 | 00 00 00 00 00 00 00 00 |........
0080 | 00 00 00 00 00 00 00 00 |........
0088 | 00 00 00 00 00 00 00 00 |........
0090 | 00 00 00 00 00 00 00 00 |........
0098 | 00 00 00 00 00 00 00 00 |........
00A0 | 00 00 00 00 00 00 00 00 |........
00A8 | 00 00 00 00 00 00 00 00 |........
00B0 | 00 00 00 00 00 00 00 00 |........
00B8 | 00 00 00 00 00 00 00 00 |........
00C0 | 00 00 00 00 00 00 00 00 |........
00C8 | 00 00 00 00 00 00 00 00 |........
00D0 | 00 00 00 00 00 00 00 00 |........
00D8 | 00 00 00 00 00 00 00 00 |........
00E0 | 00 00 00 00 00 00 00 00 |........
00E8 | 00 00 00 00 00 00 00 00 |........
00F0 | 00 00 00 00 00 00 00 00 |........
00F8 | 00 00 00 00 00 00 00 00 |........
0100 | 00 00 00 00 00 00 00 00 |........
0108 | 00 00 00 00 00 00 00 00 |........
0110 | 00 00 00 00 00 00 00 00 |........
0118 | 00 00 00 00 00 00 00 00 |........
0120 | 00 00 00 00 00 00 00 00 |........
0128 | 00 00 00 00 00 00 00 00 |........
0130 | 00 00 00 00 00 00 00 00 |........
0138 | 00 00 00 00 00 00 00 00 |........
0140 | 00 00 00 00 00 00 00 00 |........
0148 | 00 00 00 00 00 00 00 00 |........
0150 | 00 00 00 00 00 00 00 00 |........
0158 | 00 00 00 00 00 00 00 00 |........
0160 | 00 00 00 00 00 00 00 00 |........
```

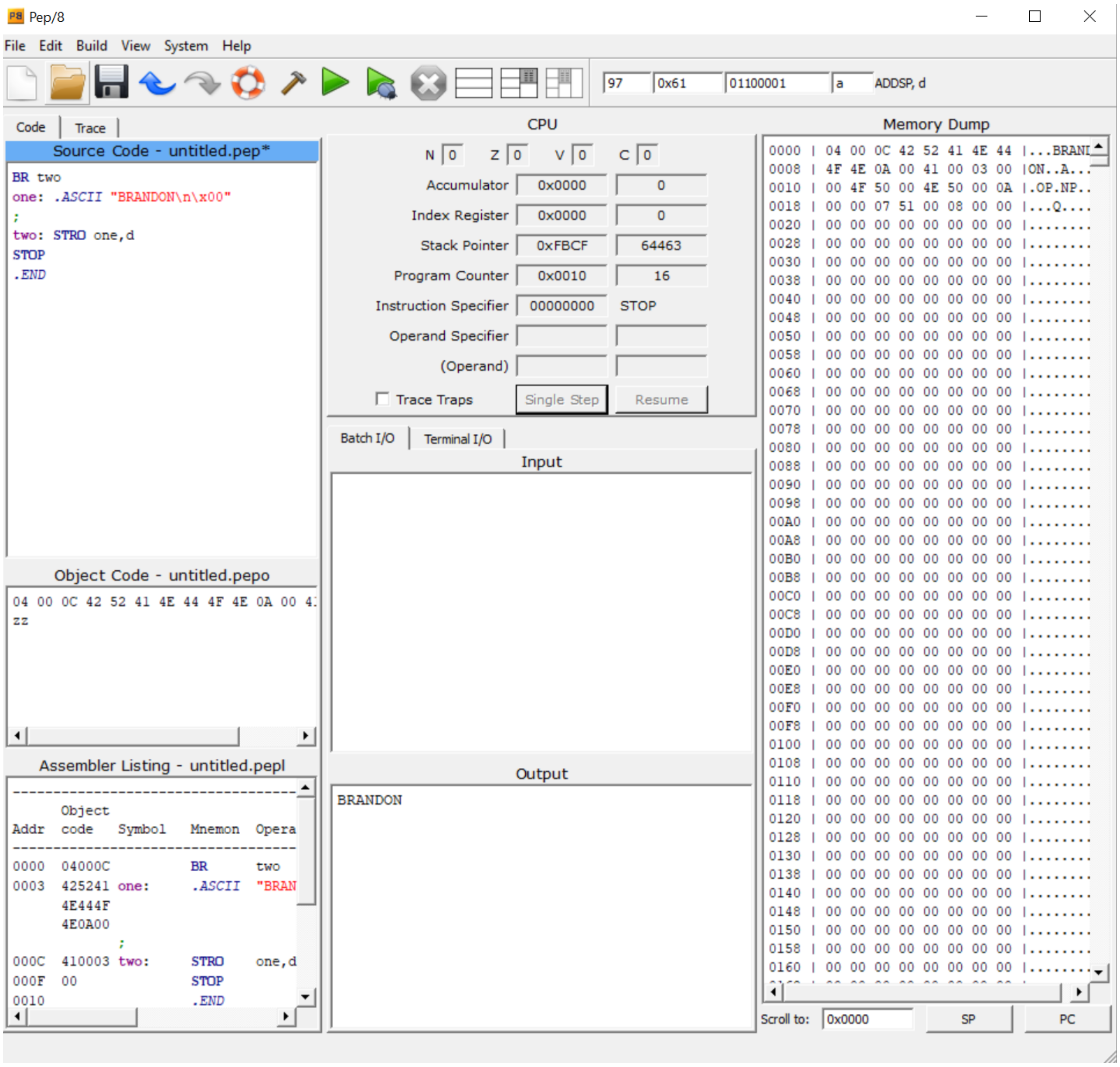
4

Write an assembly language program that prints your full name on the screen. Use .ASCII pseudo-op to store the characters at the top of your program. Use BR to branch around the characters and use STRO to output your name. Comment each line except STOP & .END. Cut & paste the Assembler Listing into your document and paste a screen shot of the Output area of the PEP/8.

```
-----------------------------------------------------------
      Object
Addr  code    Symbol   Mnemon  Operand     Comment
-----------------------------------------------------------
0000  04000C           BR      two
0003  425241  one:     .ASCII  "BRANDON\n\x00"
      4E444F
      4E0A00
               ;
000C  410003  two:     STRO    one,d
```

```
000F   00                  STOP
0010                       .END
----------------------------------------------------------
```

```
Symbol table
--------------------------------------
Symbol    Value        Symbol    Value
--------------------------------------
one       0003         two       000C
--------------------------------------
```



97 | 0x61 | 01100001 | a | ADDSP, d

Source Code - untitled.pep*

```
BR two
one: .ASCII "BRANDON\n\x00"
;
two: STRO one,d
STOP
.END
```

N 0 Z 0 V 0 C 0

Accumulator 0x0000 0

Index Register 0x0000 0

Stack Pointer 0xFBCF 64463

Program Counter 0x0010 16

Instruction Specifier 00000000 STOP

Operand Specifier

(Operand)

□ Trace Traps

Object Code - untitled.pepo

```
04 00 0C 42 52 41 4E 44 4F 4E 0A 00 4
zz
```

Assembler Listing - untitled.pepl

```
-------------------------------------
      Object
Addr  code    Symbol   Mnemon  Opera
-------------------------------------
0000  04000C           BR      two
0003  425241 one:      .ASCII  "BRAN
      4E444F
      4E0A00
              ;
000C  410003 two:      STRO    one,d
000F  00               STOP
0010                   .END
```

Output

```
BRANDON
```

Memory Dump

```
0000 | 04 00 0C 42 52 41 4E 44 |...BRANI
0008 | 4F 4E 0A 00 41 00 03 00 |ON..A...
0010 | 00 4F 50 00 4E 50 00 0A |.OP.NP..
0018 | 00 00 07 51 00 08 00 00 |...Q....
0020 | 00 00 00 00 00 00 00 00 |........
0028 | 00 00 00 00 00 00 00 00 |........
0030 | 00 00 00 00 00 00 00 00 |........
0038 | 00 00 00 00 00 00 00 00 |........
0040 | 00 00 00 00 00 00 00 00 |........
0048 | 00 00 00 00 00 00 00 00 |........
0050 | 00 00 00 00 00 00 00 00 |........
0058 | 00 00 00 00 00 00 00 00 |........
0060 | 00 00 00 00 00 00 00 00 |........
0068 | 00 00 00 00 00 00 00 00 |........
0070 | 00 00 00 00 00 00 00 00 |........
0078 | 00 00 00 00 00 00 00 00 |........
0080 | 00 00 00 00 00 00 00 00 |........
0088 | 00 00 00 00 00 00 00 00 |........
0090 | 00 00 00 00 00 00 00 00 |........
0098 | 00 00 00 00 00 00 00 00 |........
00A0 | 00 00 00 00 00 00 00 00 |........
00A8 | 00 00 00 00 00 00 00 00 |........
00B0 | 00 00 00 00 00 00 00 00 |........
00B8 | 00 00 00 00 00 00 00 00 |........
00C0 | 00 00 00 00 00 00 00 00 |........
00C8 | 00 00 00 00 00 00 00 00 |........
00D0 | 00 00 00 00 00 00 00 00 |........
00D8 | 00 00 00 00 00 00 00 00 |........
00E0 | 00 00 00 00 00 00 00 00 |........
00E8 | 00 00 00 00 00 00 00 00 |........
00F0 | 00 00 00 00 00 00 00 00 |........
00F8 | 00 00 00 00 00 00 00 00 |........
0100 | 00 00 00 00 00 00 00 00 |........
0108 | 00 00 00 00 00 00 00 00 |........
0110 | 00 00 00 00 00 00 00 00 |........
0118 | 00 00 00 00 00 00 00 00 |........
0120 | 00 00 00 00 00 00 00 00 |........
0128 | 00 00 00 00 00 00 00 00 |........
0130 | 00 00 00 00 00 00 00 00 |........
0138 | 00 00 00 00 00 00 00 00 |........
0140 | 00 00 00 00 00 00 00 00 |........
0148 | 00 00 00 00 00 00 00 00 |........
0150 | 00 00 00 00 00 00 00 00 |........
0158 | 00 00 00 00 00 00 00 00 |........
0160 | 00 00 00 00 00 00 00 00 |........
```

Scroll to: 0x0000

5a

Write an assembly language program (no loops!) that starts at 8 and counts down by 2 to 0. The C++ program is shown below. Comment each line except STOP & .END. Add something to the output that makes this program uniquely yours. Cut &

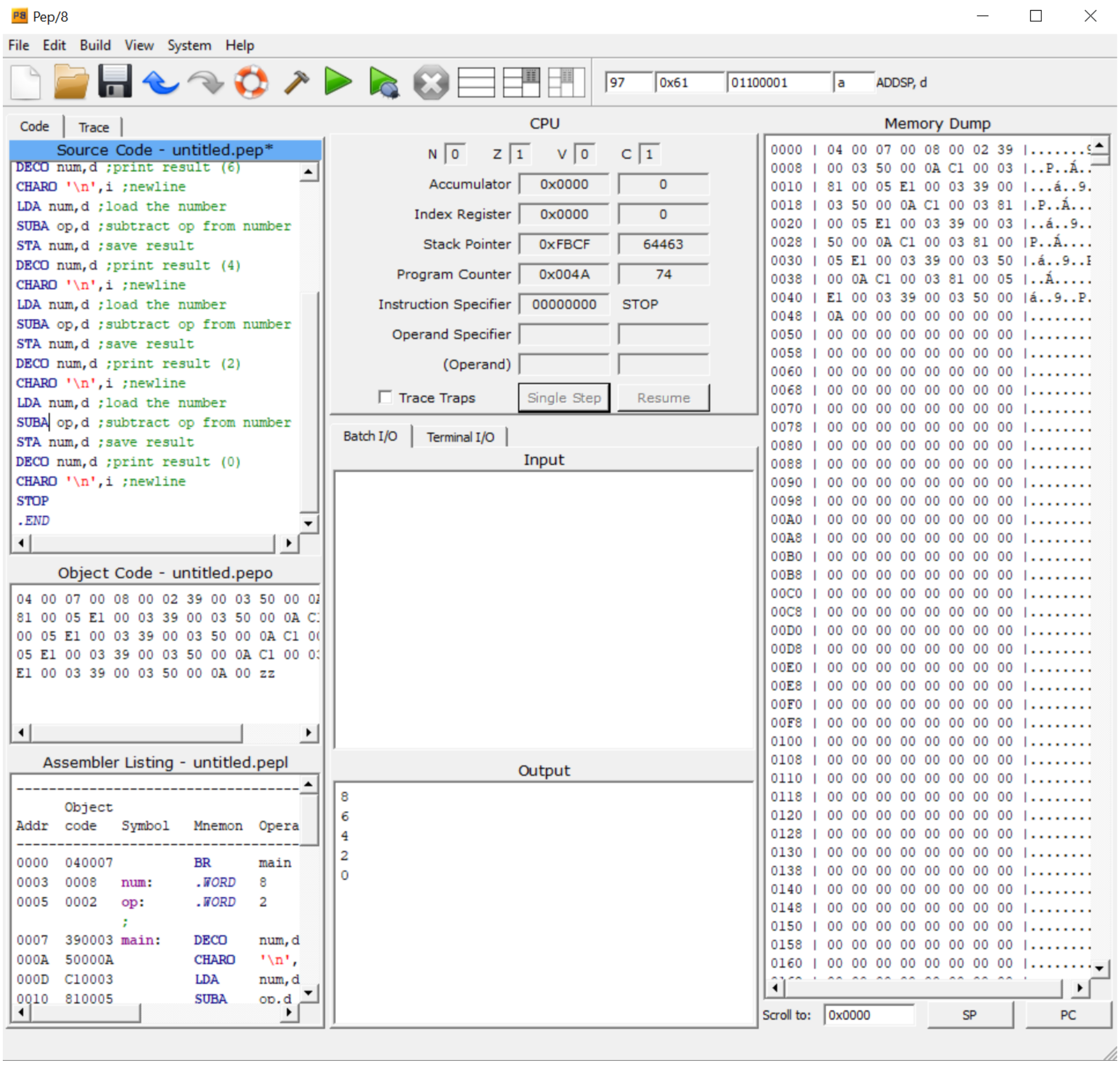
paste the Assembler Listing into your document and paste a screen shot of the Output area of the PEP/8.

```
-----------------------------------------------------------------------------
Object
Addr  code   Symbol   Mnemon  Operand     Comment
-----------------------------------------------------------------------------
0000  040007          BR      main
0003  0008   num:     .WORD   8
0005  0002   op:      .WORD   2
;
0007  390003 main:    DECO    num,d       ;Print Original(8)
000A  50000A          CHARO   '\n',i      ;newline
000D  C10003          LDA     num,d       ;load the number
0010  810005          SUBA    op,d        ;subtract op from number
0013  E10003          STA     num,d       ;save result
0016  390003          DECO    num,d       ;print result (6)
0019  50000A          CHARO   '\n',i      ;newline
001C  C10003          LDA     num,d       ;load the number
001F  810005          SUBA    op,d        ;subtract op from number
0022  E10003          STA     num,d       ;save result
0025  390003          DECO    num,d       ;print result (4)
0028  50000A          CHARO   '\n',i      ;newline
002B  C10003          LDA     num,d       ;load the number
002E  810005          SUBA    op,d        ;subtract op from number
0031  E10003          STA     num,d       ;save result
0034  390003          DECO    num,d       ;print result (2)
0037  50000A          CHARO   '\n',i      ;newline
003A  C10003          LDA     num,d       ;load the number
003D  810005          SUBA    op,d        ;subtract op from number
0040  E10003          STA     num,d       ;save result
0043  390003          DECO    num,d       ;print result (0)
0046  50000A          CHARO   '\n',i      ;newline
0049  00              STOP
004A                  .END
-----------------------------------------------------------------------------


Symbol table
--------------------------------------
Symbol    Value        Symbol    Value
--------------------------------------
main      0007         num       0003
op        0005
--------------------------------------
```

5b

Cut and paste a screen shot of the Output of the PEP/8



97 0x61 01100001 a ADDSP, d

Source Code - untitled.pep*

```
DECO num,d ;print result (6)
CHARO '\n',i ;newline
LDA num,d ;load the number
SUBA op,d ;subtract op from number
STA num,d ;save result
DECO num,d ;print result (4)
CHARO '\n',i ;newline
LDA num,d ;load the number
SUBA op,d ;subtract op from number
STA num,d ;save result
DECO num,d ;print result (2)
CHARO '\n',i ;newline
LDA num,d ;load the number
SUBA op,d ;subtract op from number
STA num,d ;save result
DECO num,d ;print result (0)
CHARO '\n',i ;newline
STOP
.END
```

Object Code - untitled.pepo

```
04 00 07 00 08 00 02 39 00 03 50 00 0A
81 00 05 E1 00 03 39 00 03 50 00 0A C1
00 05 E1 00 03 39 00 03 50 00 0A C1 00
05 E1 00 03 39 00 03 50 00 0A C1 00 03
E1 00 03 39 00 03 50 00 0A 00 zz
```

Assembler Listing - untitled.pepl

```
-----------------------------------------
      Object
Addr  code    Symbol   Mnemon  Opera
-----------------------------------------
0000  040007           BR      main
0003  0008    num:     .WORD   8
0005  0002    op:      .WORD   2
              ;
0007  390003  main:    DECO    num,d
000A  50000A           CHARO   '\n',
000D  C10003           LDA     num,d
0010  810005           SUBA    op,d
```

N 0 Z 1 V 0 C 1

Accumulator	0x0000	0
Index Register	0x0000	0
Stack Pointer	0xFBCF	64463
Program Counter	0x004A	74
Instruction Specifier	00000000	STOP
Operand Specifier		
(Operand)		

□ Trace Traps Single Step Resume

Output

```
8
6
4
2
0
```

Memory Dump

```
0000 | 04 00 07 00 08 00 02 39 |.......9
0008 | 00 03 50 00 0A C1 00 03 |..P..Á..
0010 | 81 00 05 E1 00 03 39 00 |...á..9.
0018 | 03 50 00 0A C1 00 03 81 |.P..Á...
0020 | 00 05 E1 00 03 39 00 03 |..á..9..
0028 | 50 00 0A C1 00 03 81 00 |P..Á....
0030 | 05 E1 00 03 39 00 03 50 |.á..9..P
0038 | 00 0A C1 00 03 81 00 05 |..Á.....
0040 | E1 00 03 39 00 03 50 00 |á..9..P.
0048 | 0A 00 00 00 00 00 00 00 |........
0050 | 00 00 00 00 00 00 00 00 |........
0058 | 00 00 00 00 00 00 00 00 |........
0060 | 00 00 00 00 00 00 00 00 |........
0068 | 00 00 00 00 00 00 00 00 |........
0070 | 00 00 00 00 00 00 00 00 |........
0078 | 00 00 00 00 00 00 00 00 |........
0080 | 00 00 00 00 00 00 00 00 |........
0088 | 00 00 00 00 00 00 00 00 |........
0090 | 00 00 00 00 00 00 00 00 |........
0098 | 00 00 00 00 00 00 00 00 |........
00A0 | 00 00 00 00 00 00 00 00 |........
00A8 | 00 00 00 00 00 00 00 00 |........
00B0 | 00 00 00 00 00 00 00 00 |........
00B8 | 00 00 00 00 00 00 00 00 |........
00C0 | 00 00 00 00 00 00 00 00 |........
00C8 | 00 00 00 00 00 00 00 00 |........
00D0 | 00 00 00 00 00 00 00 00 |........
00D8 | 00 00 00 00 00 00 00 00 |........
00E0 | 00 00 00 00 00 00 00 00 |........
00E8 | 00 00 00 00 00 00 00 00 |........
00F0 | 00 00 00 00 00 00 00 00 |........
00F8 | 00 00 00 00 00 00 00 00 |........
0100 | 00 00 00 00 00 00 00 00 |........
0108 | 00 00 00 00 00 00 00 00 |........
0110 | 00 00 00 00 00 00 00 00 |........
0118 | 00 00 00 00 00 00 00 00 |........
0120 | 00 00 00 00 00 00 00 00 |........
0128 | 00 00 00 00 00 00 00 00 |........
0130 | 00 00 00 00 00 00 00 00 |........
0138 | 00 00 00 00 00 00 00 00 |........
0140 | 00 00 00 00 00 00 00 00 |........
0148 | 00 00 00 00 00 00 00 00 |........
0150 | 00 00 00 00 00 00 00 00 |........
0158 | 00 00 00 00 00 00 00 00 |........
0160 | 00 00 00 00 00 00 00 00 |........
```

Scroll to: 0x0000

5c

Explain the status bit(s) NZVC at the point that STOP is loaded.
N = 0
Z = 1
V = 0
C = 1

6a

Write an assembly language program that corresponds to the following C++ program. Comment each line except STOP & .END. Add something to the output that makes this program uniquely yours. Cut & paste the Source Code into your document. (Hint: PEP/8 does not have a divide instruction; however we have discussed an instruction that divides by 2. Please use that instruction.)

Source Code:

```
BR main
numA: .WORD 0
```

```
numB: .WORD 0
numC: .WORD 0
numD: .WORD 0
sum: .WORD 0
avg: .WORD 0
labA: .ASCII "input a = \x00"
labB: .ASCII "input b = \x00"
labC: .ASCII "input c = \x00"
labD: .ASCII "input d = \x00"
labSum: .ASCII "sum = \x00"
labAvg: .ASCII "average = \x00"
;set the variables
main: LDA 2,i
STA numA,d ;set a = 2
LDA 4,i
STA numB,d ;set b = 4
LDA 5,i
STA numC,d ;set c = 5
LDA 1,i
STA numD,d ;set d = 1
;calculate results
LDA numA,d ;load a
ADDA numB,d ;add b
ADDA numC,d ;add c
ADDA numD,d ;add d
STA sum,d ;store sum
;calculate simple average
LDA sum,d ;load sum
ASRA ;Arithmetic left(proxy division by 2)
ASRA ;As above
STA avg,d ;store average
;tell the variables
STRO labA,d ;print a label
DECO numA,d ;Print a
CHARO '\n',i ;newline
STRO labB,d ;print b label
DECO numB,d ;Print b
CHARO '\n',i ;newline
STRO labC,d ;print c label
DECO numC,d ;Print c
CHARO '\n',i ;newline
STRO labD,d ;print d label
DECO numD,d ;Print d
CHARO '\n',i ;newline
STRO labSum,d ;print sum label
DECO sum,d ;Print sum
CHARO '\n',i ;newline
STRO labAvg,d ;print average label
```

```
DECO avg,d ;Print average
STOP
.END
```

Assembler Listing:

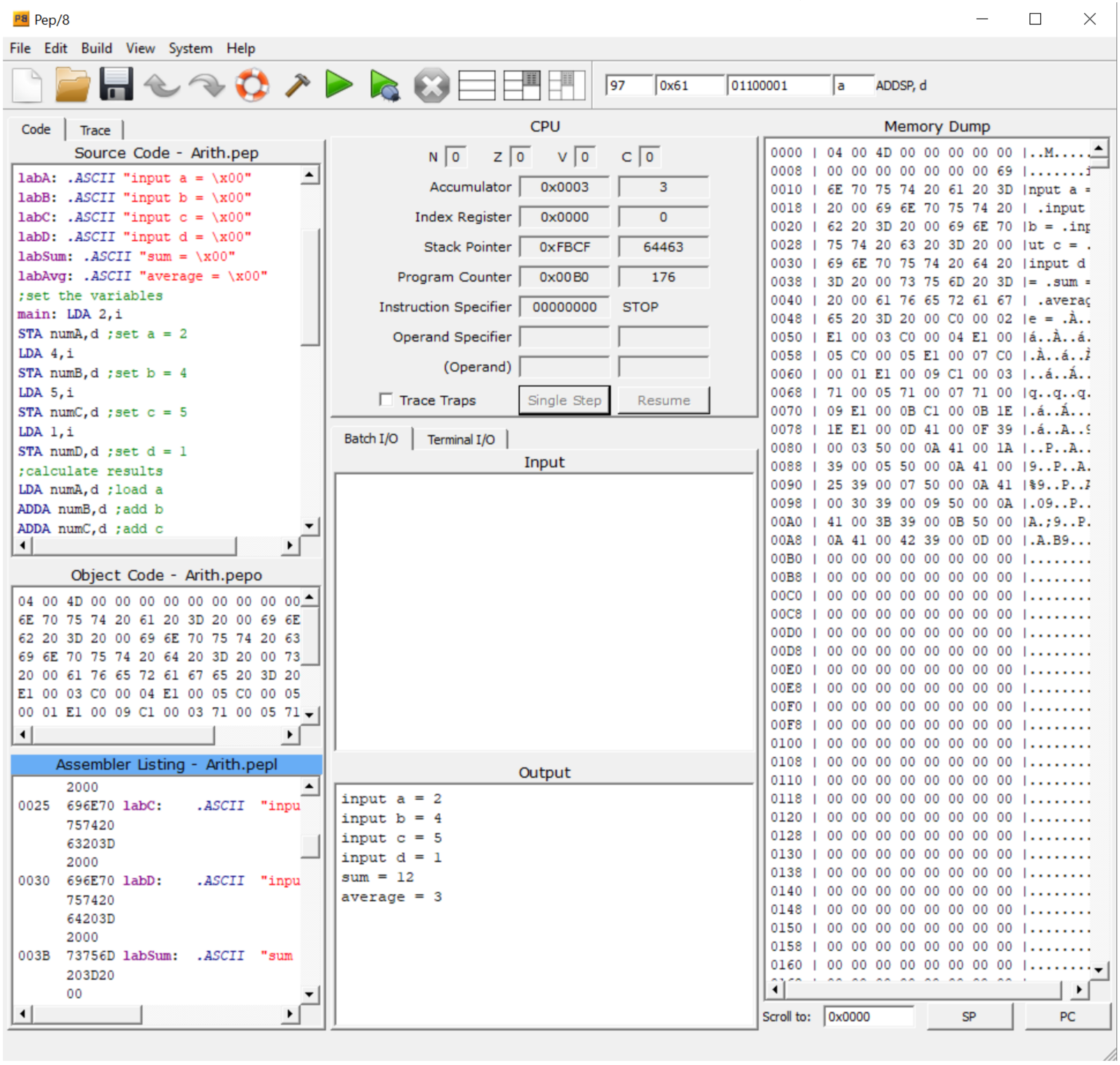
```
--------------------------------------------------------------------------
Object
Addr  code   Symbol   Mnemon  Operand     Comment
--------------------------------------------------------------------------
0000  04004D          BR      main
0003  0000   numA:    .WORD   0
0005  0000   numB:    .WORD   0
0007  0000   numC:    .WORD   0
0009  0000   numD:    .WORD   0
000B  0000   sum:     .WORD   0
000D  0000   avg:     .WORD   0
000F  696E70 labA:    .ASCII  "input a = \x00"
757420
61203D
2000
001A  696E70 labB:    .ASCII  "input b = \x00"
757420
62203D
2000
0025  696E70 labC:    .ASCII  "input c = \x00"
757420
63203D
2000
0030  696E70 labD:    .ASCII  "input d = \x00"
757420
64203D
2000
003B  73756D labSum:  .ASCII  "sum = \x00"
203D20
00
0042  617665 labAvg:  .ASCII  "average = \x00"
726167
65203D
2000
;set the variables
004D  C00002 main:    LDA     2,i
0050  E10003          STA     numA,d      ;set a = 2
0053  C00004          LDA     4,i
0056  E10005          STA     numB,d      ;set b = 4
0059  C00005          LDA     5,i
005C  E10007          STA     numC,d      ;set c = 5
005F  C00001          LDA     1,i
0062  E10009          STA     numD,d      ;set d = 1
```

```
;calculate results
0065  C10003          LDA     numA,d       ;load a
0068  710005          ADDA    numB,d       ;add b
006B  710007          ADDA    numC,d       ;add c
006E  710009          ADDA    numD,d       ;add d
0071  E1000B          STA     sum,d        ;store sum
;calculate simple average
0074  C1000B          LDA     sum,d        ;load sum
0077  1E              ASRA                 ;Arithmetic left(proxy division by 2)
0078  1E              ASRA                 ;As above
0079  E1000D          STA     avg,d        ;store average
;tell the variables
007C  41000F          STRO    labA,d       ;print a label
007F  390003          DECO    numA,d       ;Print a
0082  50000A          CHARO   '\n',i       ;newline
0085  41001A          STRO    labB,d       ;print b label
0088  390005          DECO    numB,d       ;Print b
008B  50000A          CHARO   '\n',i       ;newline
008E  410025          STRO    labC,d       ;print c label
0091  390007          DECO    numC,d       ;Print c
0094  50000A          CHARO   '\n',i       ;newline
0097  410030          STRO    labD,d       ;print d label
009A  390009          DECO    numD,d       ;Print d
009D  50000A          CHARO   '\n',i       ;newline
00A0  41003B          STRO    labSum,d     ;print sum label
00A3  39000B          DECO    sum,d        ;Print sum
00A6  50000A          CHARO   '\n',i       ;newline
00A9  410042          STRO    labAvg,d     ;print average label
00AC  39000D          DECO    avg,d        ;Print average
00AF  00              STOP
00B0                  .END
---------------------------------------------------------------------------


Symbol table
--------------------------------------
Symbol    Value        Symbol    Value
--------------------------------------
avg       000D         labA      000F
labAvg    0042         labB      001A
labC      0025         labD      0030
labSum    003B         main      004D
numA      0003         numB      0005
numC      0007         numD      0009
sum       000B
--------------------------------------
```

6b

Run it twice – once with values that yield a output that is within the range of the PEP/8 and once with values that yield a output that is outside the range of the PEP/8. Explain the limits. Paste screen shots of the Output area of the PEP/8 for both runs.



97 0x61 01100001 a ADDSP, d

Source Code - Arith.pep

```
labA: .ASCII "input a = \x00"
labB: .ASCII "input b = \x00"
labC: .ASCII "input c = \x00"
labD: .ASCII "input d = \x00"
labSum: .ASCII "sum = \x00"
labAvg: .ASCII "average = \x00"
;set the variables
main: LDA 2,i
STA numA,d ;set a = 2
LDA 4,i
STA numB,d ;set b = 4
LDA 5,i
STA numC,d ;set c = 5
LDA 1,i
STA numD,d ;set d = 1
;calculate results
LDA numA,d ;load a
ADDA numB,d ;add b
ADDA numC,d ;add c
```

Object Code - Arith.pepo

```
04 00 4D 00 00 00 00 00 00 00 00 00
6E 70 75 74 20 61 20 3D 20 00 69 6E
62 20 3D 20 00 69 6E 70 75 74 20 63
69 6E 70 75 74 20 64 20 3D 20 00 73
20 00 61 76 65 72 61 67 65 20 3D 20
E1 00 03 C0 00 04 E1 00 05 C0 00 05
00 01 E1 00 09 C1 00 03 71 00 05 71
```

Assembler Listing - Arith.pepl

```
     2000
0025 696E70 labC:    .ASCII  "inpu
     757420
     63203D
     2000
0030 696E70 labD:    .ASCII  "inpu
     757420
     64203D
     2000
003B 73756D labSum:  .ASCII  "sum
     203D20
     00
```

CPU

N 0 Z 0 V 0 C 0

Accumulator 0x0003 3

Index Register 0x0000 0

Stack Pointer 0xFBCF 64463

Program Counter 0x00B0 176

Instruction Specifier 00000000 STOP

Operand Specifier

(Operand)

□ Trace Traps

Input

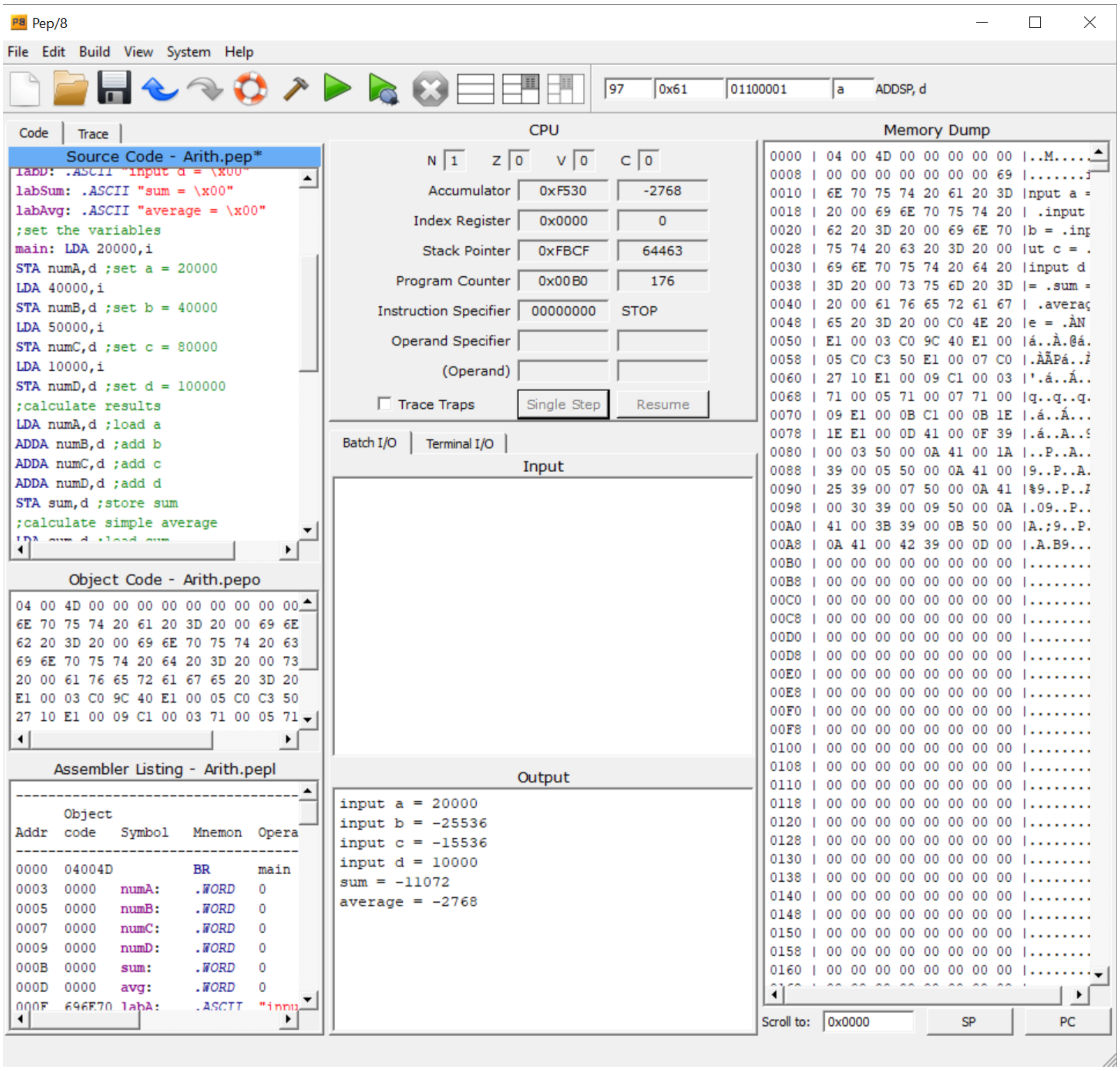
Output

```
input a = 2
input b = 4
input c = 5
input d = 1
sum = 12
average = 3
```

Memory Dump

```
0000 | 04 00 4D 00 00 00 00 00 |..M.....
0008 | 00 00 00 00 00 00 00 69 |.......i
0010 | 6E 70 75 74 20 61 20 3D |nput a =
0018 | 20 00 69 6E 70 75 74 20 | .input
0020 | 62 20 3D 20 00 69 6E 70 |b = .inp
0028 | 75 74 20 63 20 3D 20 00 |ut c = .
0030 | 69 6E 70 75 74 20 64 20 |input d
0038 | 3D 20 00 73 75 6D 20 3D |= .sum =
0040 | 20 00 61 76 65 72 61 67 | .averag
0048 | 65 20 3D 20 00 C0 00 02 |e = .À..
0050 | E1 00 03 C0 00 04 E1 00 |á..À..á.
0058 | 05 C0 00 05 E1 00 07 C0 |.À..á..À
0060 | 00 01 E1 00 09 C1 00 03 |..á..Á..
0068 | 71 00 05 71 00 07 71 00 |q..q..q.
0070 | 09 E1 00 0B C1 00 0B 1E |.á..Á...
0078 | 1E E1 00 0D 41 00 0F 39 |.á..A..9
0080 | 00 03 50 00 0A 41 00 1A |..P..A..
0088 | 39 00 05 50 00 0A 41 00 |9..P..A.
0090 | 25 39 00 07 50 00 0A 41 |%9..P..A
0098 | 00 30 39 00 09 50 00 0A |.09..P..
00A0 | 41 00 3B 39 00 0B 50 00 |A.;9..P.
00A8 | 0A 41 00 42 39 00 0D 00 |.A.B9...
00B0 | 00 00 00 00 00 00 00 00 |........
00B8 | 00 00 00 00 00 00 00 00 |........
00C0 | 00 00 00 00 00 00 00 00 |........
00C8 | 00 00 00 00 00 00 00 00 |........
00D0 | 00 00 00 00 00 00 00 00 |........
00D8 | 00 00 00 00 00 00 00 00 |........
00E0 | 00 00 00 00 00 00 00 00 |........
00E8 | 00 00 00 00 00 00 00 00 |........
00F0 | 00 00 00 00 00 00 00 00 |........
00F8 | 00 00 00 00 00 00 00 00 |........
0100 | 00 00 00 00 00 00 00 00 |........
0108 | 00 00 00 00 00 00 00 00 |........
0110 | 00 00 00 00 00 00 00 00 |........
0118 | 00 00 00 00 00 00 00 00 |........
0120 | 00 00 00 00 00 00 00 00 |........
0128 | 00 00 00 00 00 00 00 00 |........
0130 | 00 00 00 00 00 00 00 00 |........
0138 | 00 00 00 00 00 00 00 00 |........
0140 | 00 00 00 00 00 00 00 00 |........
0148 | 00 00 00 00 00 00 00 00 |........
0150 | 00 00 00 00 00 00 00 00 |........
0158 | 00 00 00 00 00 00 00 00 |........
0160 | 00 00 00 00 00 00 00 00 |........
```

Scroll to: 0x0000



```
labD: .ASCII "input d = \x00"
labSum: .ASCII "sum = \x00"
labAvg: .ASCII "average = \x00"
;set the variables
main: LDA 20000,i
STA numA,d ;set a = 20000
LDA 40000,i
STA numB,d ;set b = 40000
LDA 50000,i
STA numC,d ;set c = 80000
LDA 10000,i
STA numD,d ;set d = 100000
;calculate results
LDA numA,d ;load a
ADDA numB,d ;add b
ADDA numC,d ;add c
ADDA numD,d ;add d
STA sum,d ;store sum
;calculate simple average
LDA sum,d ;load sum
```

```
04 00 4D 00 00 00 00 00 00 00 00 00
6E 70 75 74 20 61 20 3D 20 00 69 6E
62 20 3D 20 00 69 6E 70 75 74 20 63
69 6E 70 75 74 20 64 20 3D 20 00 73
20 00 61 76 65 72 61 67 65 20 3D 20
E1 00 03 C0 9C 40 E1 00 05 C0 C3 50
27 10 E1 00 09 C1 00 03 71 00 05 71
```

```
-----------------------------------------
     Object
Addr  code    Symbol    Mnemon   Opera
-----------------------------------------
0000  04004D            BR       main
0003  0000    numA:     .WORD    0
0005  0000    numB:     .WORD    0
0007  0000    numC:     .WORD    0
0009  0000    numD:     .WORD    0
000B  0000    sum:      .WORD    0
000D  0000    avg:      .WORD    0
000F  696E70  labA:     .ASCII   "inpu
```

N 1 Z 0 V 0 C 0

Accumulator	0xF530	-2768
Index Register	0x0000	0
Stack Pointer	0xFBCF	64463
Program Counter	0x00B0	176
Instruction Specifier	00000000	STOP
Operand Specifier		
(Operand)		

```
input a = 20000
input b = -25536
input c = -15536
input d = 10000
sum = -11072
average = -2768
```

```
0000 | 04 00 4D 00 00 00 00 00 |..M.....
0008 | 00 00 00 00 00 00 00 69 |.......i
0010 | 6E 70 75 74 20 61 20 3D |nput a =
0018 | 20 00 69 6E 70 75 74 20 | .input
0020 | 62 20 3D 20 00 69 6E 70 |b = .inp
0028 | 75 74 20 63 20 3D 20 00 |ut c = .
0030 | 69 6E 70 75 74 20 64 20 |input d
0038 | 3D 20 00 73 75 6D 20 3D |= .sum =
0040 | 20 00 61 76 65 72 61 67 | .averag
0048 | 65 20 3D 20 00 C0 4E 20 |e = .ÀN
0050 | E1 00 03 C0 9C 40 E1 00 |á..À.@á.
0058 | 05 C0 C3 50 E1 00 07 C0 |.ÀÃPá..À
0060 | 27 10 E1 00 09 C1 00 03 |'.á..Á..
0068 | 71 00 05 71 00 07 71 00 |q..q..q.
0070 | 09 E1 00 0B C1 00 0B 1E |.á..Á...
0078 | 1E E1 00 0D 41 00 0F 39 |.á..A..9
0080 | 00 03 50 00 0A 41 00 1A |..P..A..
0088 | 39 00 05 50 00 0A 41 00 |9..P..A.
0090 | 25 39 00 07 50 00 0A 41 |%9..P..A
0098 | 00 30 39 00 09 50 00 0A |.09..P..
00A0 | 41 00 3B 39 00 0B 50 00 |A.;9..P.
00A8 | 0A 41 00 42 39 00 0D 00 |.A.B9...
00B0 | 00 00 00 00 00 00 00 00 |........
00B8 | 00 00 00 00 00 00 00 00 |........
00C0 | 00 00 00 00 00 00 00 00 |........
00C8 | 00 00 00 00 00 00 00 00 |........
00D0 | 00 00 00 00 00 00 00 00 |........
00D8 | 00 00 00 00 00 00 00 00 |........
00E0 | 00 00 00 00 00 00 00 00 |........
00E8 | 00 00 00 00 00 00 00 00 |........
00F0 | 00 00 00 00 00 00 00 00 |........
00F8 | 00 00 00 00 00 00 00 00 |........
0100 | 00 00 00 00 00 00 00 00 |........
0108 | 00 00 00 00 00 00 00 00 |........
0110 | 00 00 00 00 00 00 00 00 |........
0118 | 00 00 00 00 00 00 00 00 |........
0120 | 00 00 00 00 00 00 00 00 |........
0128 | 00 00 00 00 00 00 00 00 |........
0130 | 00 00 00 00 00 00 00 00 |........
0138 | 00 00 00 00 00 00 00 00 |........
0140 | 00 00 00 00 00 00 00 00 |........
0148 | 00 00 00 00 00 00 00 00 |........
0150 | 00 00 00 00 00 00 00 00 |........
0158 | 00 00 00 00 00 00 00 00 |........
0160 | 00 00 00 00 00 00 00 00 |........
```

6c

Explain the status bit(s) NZVC at the point that STOP is loaded for the invalid run.

$N = 1$
$Z = 0$
$V = 1$
$C = 0$

References

Warford, J. (2009). *Computer systems* (4th ed.). Jones and Bartlett.