System Programming
< LAB 2 >

Prepared Date :

March, 25, 2019

Prepared by :

Hagar Usama 4970

Contents:

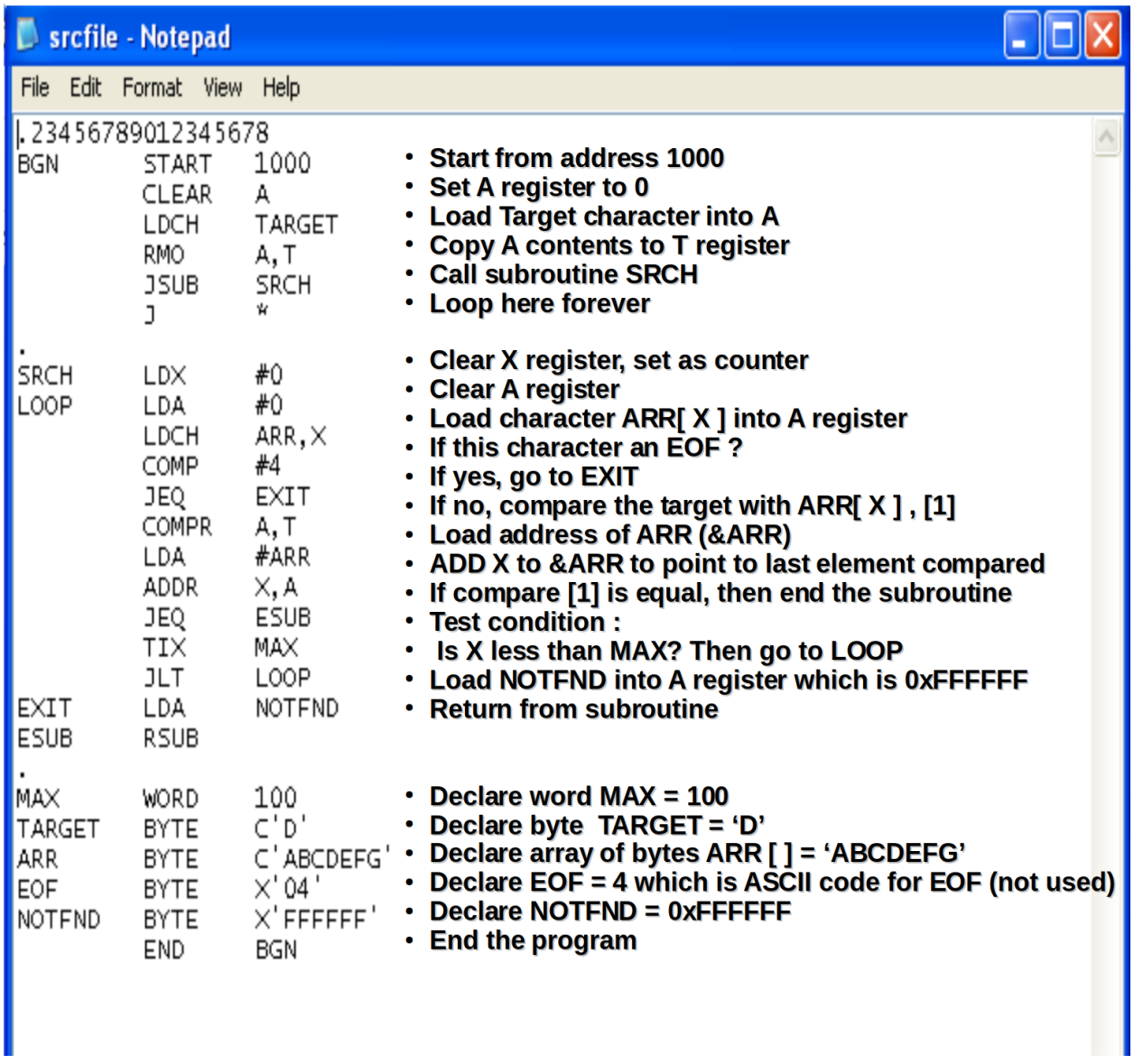
Code Description :

The lab is mainly divided into 4 problems. Each one has its own source file attached to the report :

- SRCHAR
- read_write_dev
- upper_i_o
- bubble_char

SRCHAR:

Function : to search for a certain byte in a string, return the address of the byte else return 0xFFFFFF into A register.

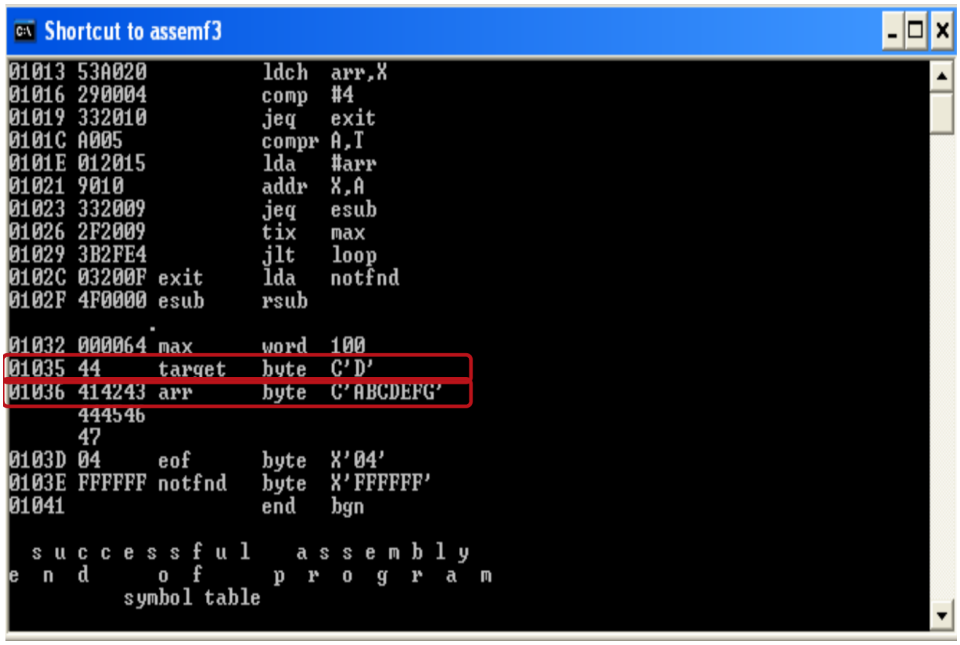


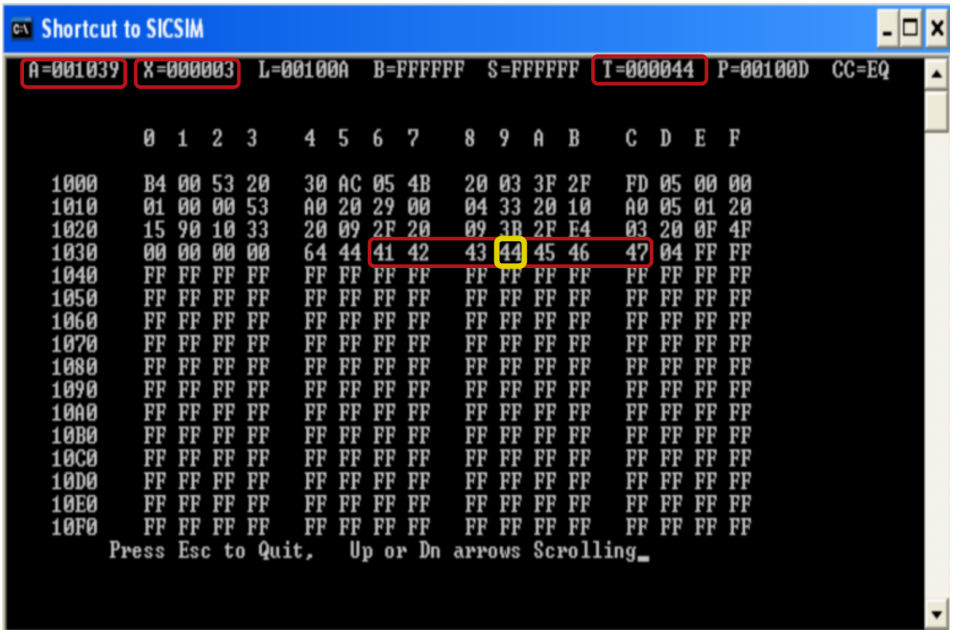
Assumptions :

Maximum Size for our array is 100

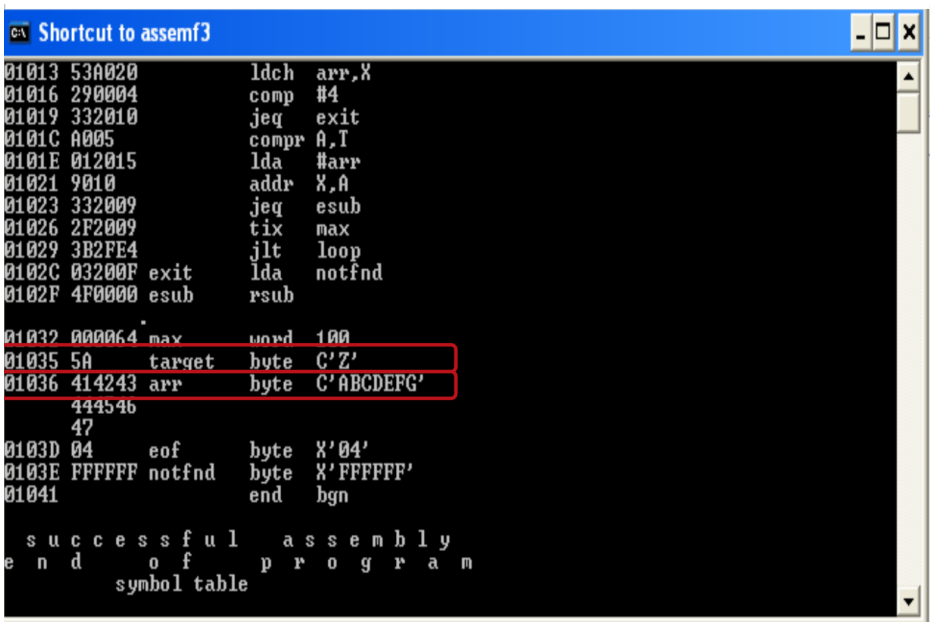
Samples :

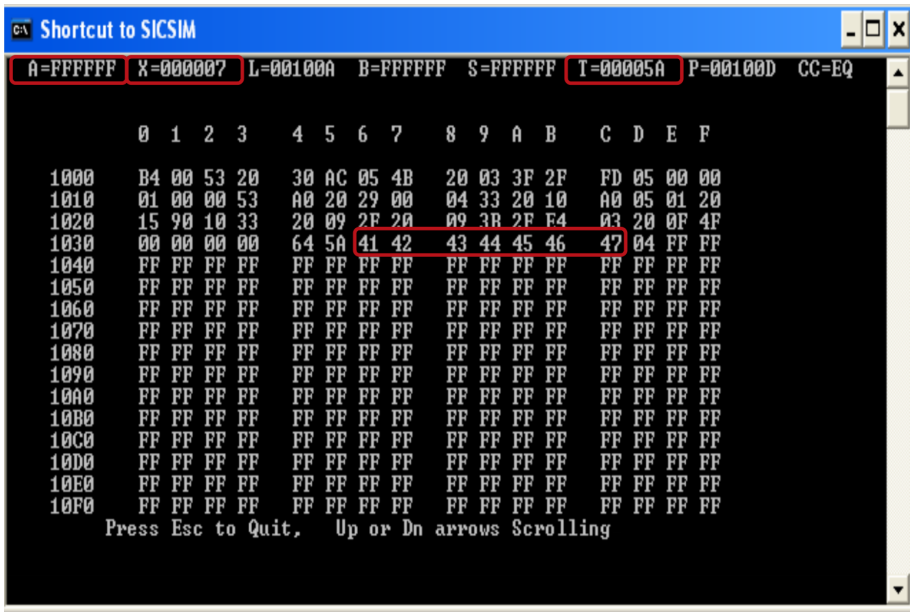
Case 1 :





Case 2:





Read_write_dev :

Function : to read a string from an input device and print it reversed to an output device.

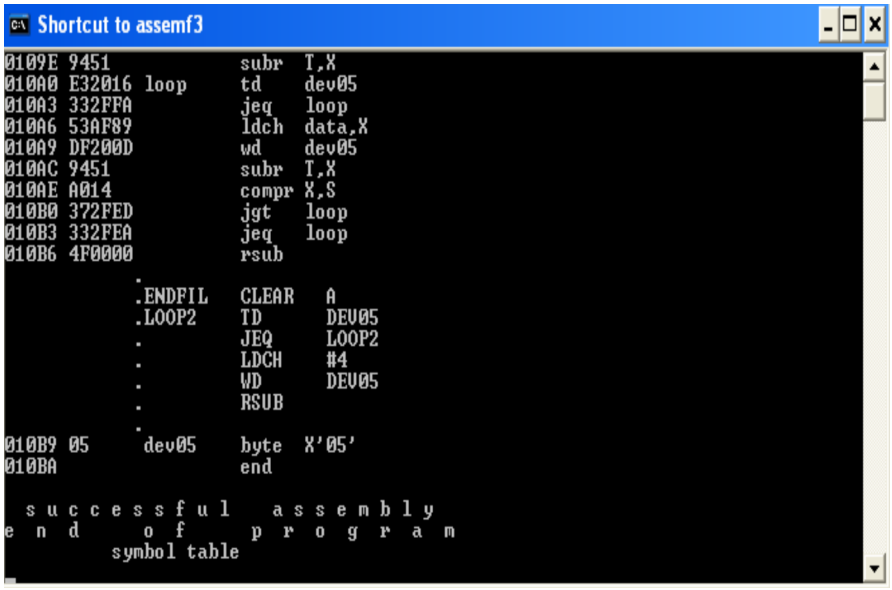
srcfile - Notepad

File Edit Format View Help

```
.234567890123456789
.LABEL...OPCODE..OPERAND
         START   1000
         JSUB    READ
         JSUB    WRITE
.        JSUB    ENDFIL
         J       *
.
.SUBROUTINE FOR READING FROM DEV F3
.
READ     CLEAR   X
         CLEAR   A
LPR      TD      DEVF3
         JEQ     LPR
         RD      DEVF3
         COMP    #04
         JEQ     EXIT
         STCH    DATA,X
         TIX     MAX
         JLT     LPR
EXIT     STX     LENGTH
         RSUB
.
DEVF3    BYTE    X'F3'
LENGTH   RESW    1
MAX      WORD    100
DATA     RESB    100
.
.SUBROUTINE FOR WRITING INTO DEV
.
WRITE    CLEAR   A
         LDT     #1
         LDS     #0
         SUBR    T,X
LOOP     TD      DEV05
         JEQ     LOOP
         LDCH    DATA,X
         WD      DEV05

         SUBR    T,X
         COMPR   X,S
         JGT     LOOP
         JEQ     LOOP
         RSUB
.
.ENDFIL   CLEAR   A
.LOOP2    TD      DEV05
.         JEQ     LOOP2
.         LDCH    #4
.         WD      DEV05
.         RSUB
.
DEV05    BYTE    X'05'
         END
```

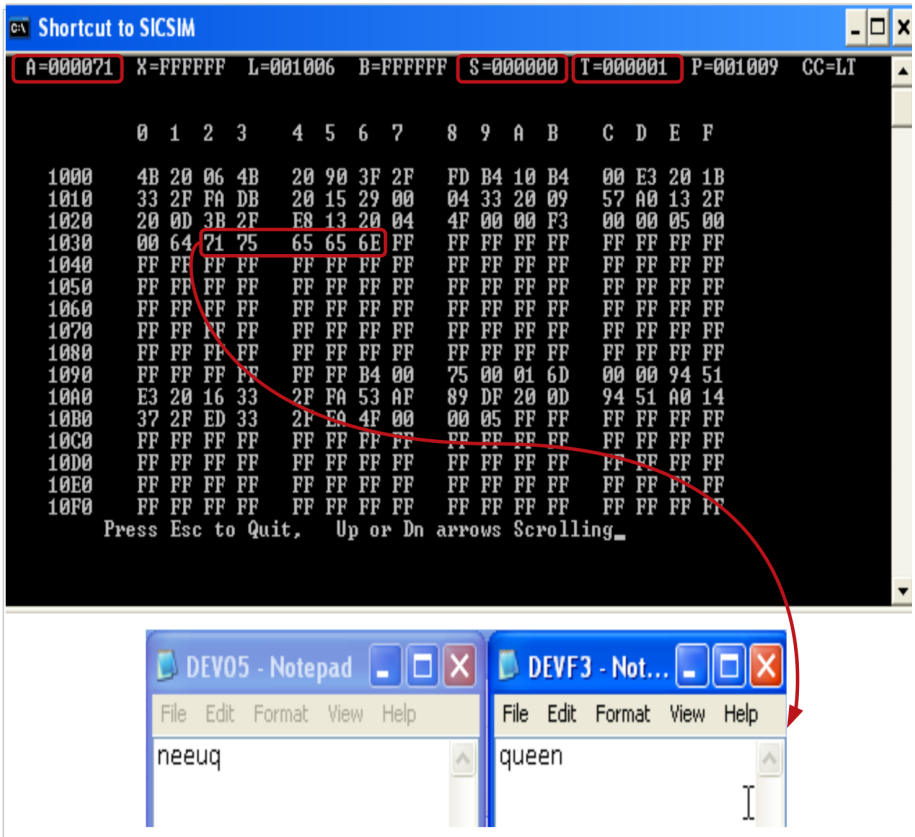
- Start from address 1000
- Call subroutine READ
- Call subroutine WRITE
- Not used (Call subroutine ENDFIL)
- Loop here forever
-
-
-
- Clear register X to set as a counter
- Clear register A
- Test device DEVF3
- Is Ready?
  - If no, go to LPR to test again
  - If yes, read a byte from the device into A
- Compare the byte with EOF 0x04
- If equal, go to EXIT
  - If no, store the character into array DATA[X]
- Increment X and Compare to MAX
  - If less than, loop (go to LPR)
  - If not, store the X into LENGTH
- Return from subroutine
-
- Input device number
- One-word variable
- One-word constant = 100
- Array varaible 100-byte
-
- Clear register A
- Load 1 into register T
- Load 0 into register S (loop limit)
- X = X – T
- Test output device
- Is Ready?
  - If no, go to loop to test again
  - If yes, load a character from DATA[X] into A
- Write one-byte into output device
- X = X – T (to read from last to first)
- Compare register X to 0
  - If NOT less than 0
    - Go to loop (continue looping)
- Return from subroutine
-
- ENDFIL : (not used) : to add EOF character into the output device at the end
-
-
-
-
-
- Output device number
- End program



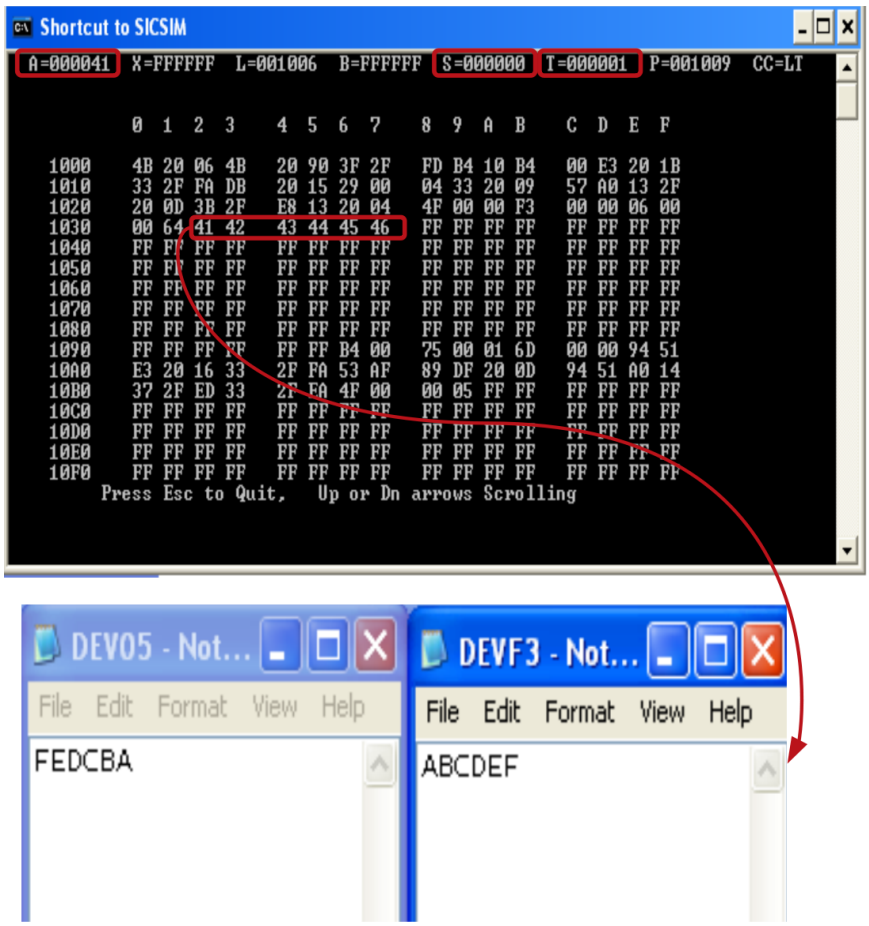
Assumptions :

DATA: array to read from input device maximum 100-byte

Case 1 :



Case 2:



---

upper_i_o:

Function : to read a string from an input device and print the string to an output device capitalized.

srcfile - Notepad

File Edit Format View Help

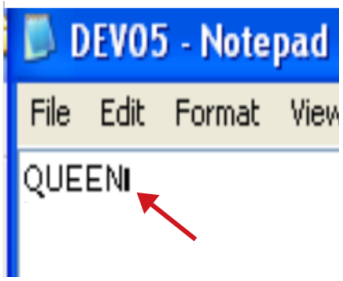
```
.23456789012345678
.LABEL...opcode..OPERAND
BGN       START   1000
          JSUB    UPPER
          JSUB    WRITE
.         JSUB    ENDFIL
          J       *
.
UPPER     LDX     #0
          LDA     #0
LOOP1     TD      DEVF3
          JEQ     LOOP1
          RD      DEVF3
          COMP    #4
          JEQ     EXIT
          SUB     #32
          STCH    DATA,X
          TIX     MAX
          JLT     LOOP1
EXIT      STX     LENGTH
          RSUB
LENGTH    RESW    1
MAX       WORD    15
DATA      RESW    100
DEVF3     BYTE    X'F3'
.
. WRITE SUBROUTINE TO WRITE INTO DEVICE
.
WRITE     CLEAR   A
          CLEAR   X
LOOP2     TD      DEV05
          JEQ     LOOP2
          LDCH    DATA,X
          WD      DEV05
          TIX     LENGTH
          JLT     LOOP2
          RSUB
.
.ENDFIL   CLEAR   A
.LOOP3    TD      DEV05
.         JEQ     LOOP3
.         LDCH    #4
.         WD      DEV05
.         RSUB
.
DEV05     BYTE    X'05'
          END     BGN
```

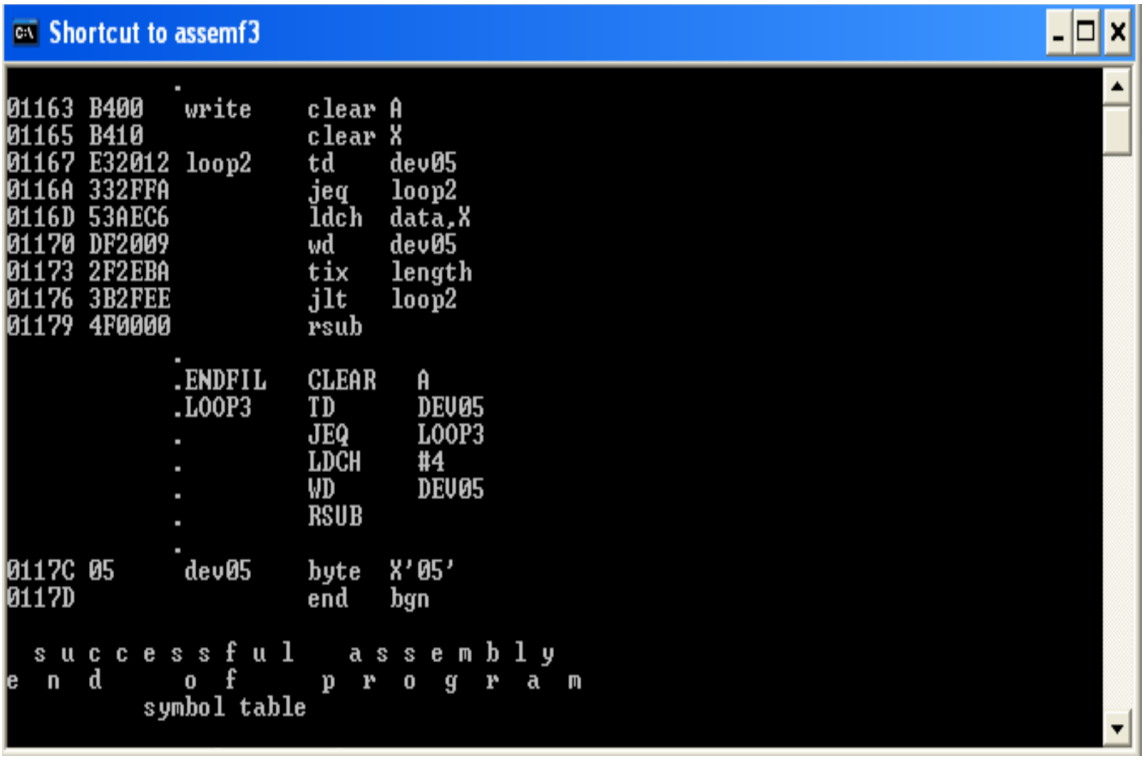
- Start from address 1000
- Call subroutine UPPER
- Call subroutine WRITE
- Call subroutine ENDFIL (not used)
- Loop here forever
- Clear register X as counter
- Clear register A to load charcters in
- Test Input device
- Is Ready?
  - If no, go to LOOP1 (loop again)
  - If yes, compare register A to EOF
    -
    - If equal, go to EXIT
- Subtract 32 from charcter stored in A to convert into uppercase
- Store the new charcter into DATA[X]
- X = X+1 , compare X to MAX
- If less than : go to LOOP1
- Else : store X into LENGTH
- Return from subroutine
- Declare a variable Length (one-word)
- Declare a constant MAX = 15
- Declare array of 100 bytes DATA[100]
- Input device number
- Clear register A
- Clear register X
- Test output device
- Is Ready?
  - If no, loop until device is ready
  - If yes, load a charcter of DATA[X] into register A
  - Write the character into output device
  - X = X + 1 , compare X to LENGTH
  - Is less than? Go to LOOP2
  - Return from subroutine
- ENDFIL : (not used) : to add EOF character
- Output device number
- End program

Assumptions :

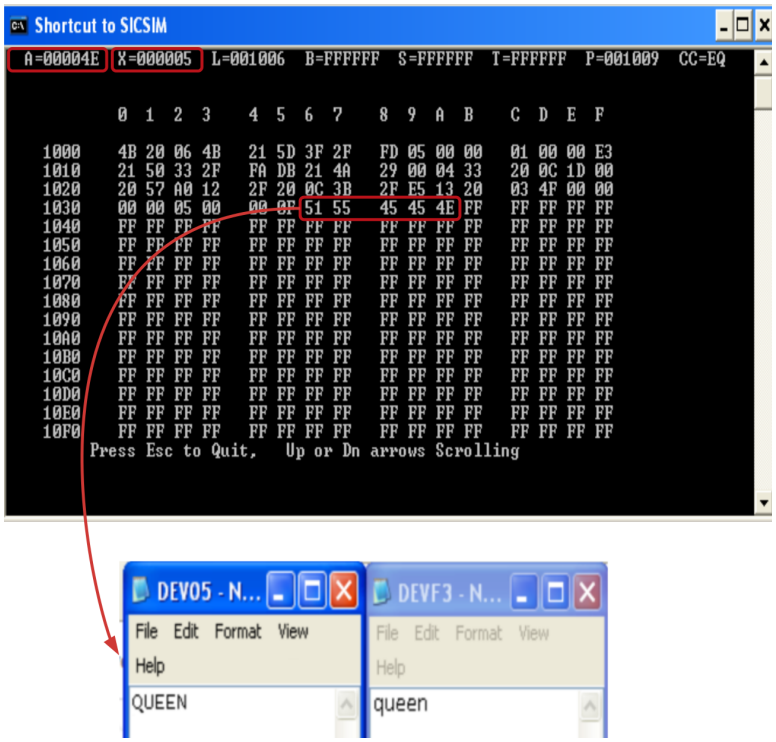
Assume input data are lowercase characters (special characters excluded)

NOTE : ENDFIL subroutine should be used to end output device file by EOF (both in read_write_dev & upper_i_o) , however the EOF character seems to be visible in windows XP, so I discarded it.

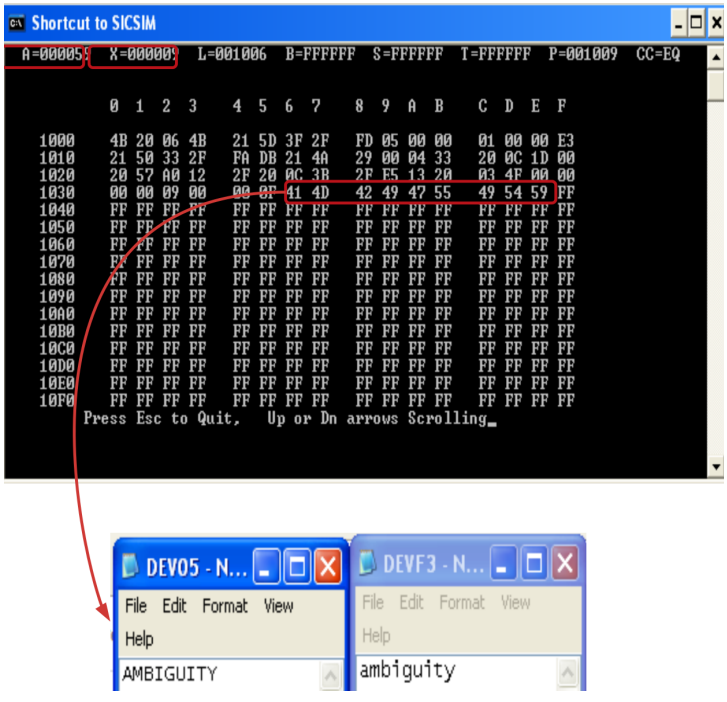




Case 1 :


Shortcut to SICSIM
A=00004E X=000005 L=001006 B=FFFFFF S=FFFFFF T=FFFFFF P=001009 CC=EQ
0 1 2 3 4 5 6 7 8 9 A B C D E F
1000 4B 20 06 4B 21 5D 3F 2F FD 05 00 00 01 00 00 E3
1010 21 50 33 2F FA DB 21 4A 29 00 04 33 20 0C 1D 00
1020 20 57 A0 12 2F 20 0C 3B 2F E5 13 20 03 4F 00 00
1030 00 00 05 00 00 00 51 55 45 45 4E FF FF FF FF FF
Press Esc to Quit, Up or Dn arrows Scrolling
DEV05 - N...
File Edit Format View
Help
QUEEN
DEVF3 - N...
File Edit Format View
Help
queen

Case 2 :


Shortcut to SICSIM
A=000059 X=000009 L=001006 B=FFFFFF S=FFFFFF T=FFFFFF P=001009 CC=EQ
0 1 2 3 4 5 6 7 8 9 A B C D E F
1000 4B 20 06 4B 21 5D 3F 2F FD 05 00 00 01 00 00 E3
1010 21 50 33 2F FA DB 21 4A 29 00 04 33 20 0C 1D 00
1020 20 57 A0 12 2F 20 0C 3B 2F E5 13 20 03 4F 00 00
1030 00 00 09 00 00 00 41 4D 42 49 47 55 49 54 59 FF
Press Esc to Quit, Up or Dn arrows Scrolling
DEV05 - N...
File Edit Format View
Help
AMBIGUITY
DEVF3 - N...
File Edit Format View
Help
ambiguity

bubble_char :

The algorithm is :

```
 for( i=0; i<n-1; i++)
        for(j=0; j<n-i-1 ; j++)
              If (arr[j] > arr[j+1])
                 swap(arr[j] , arr[j+1])
```

Where :

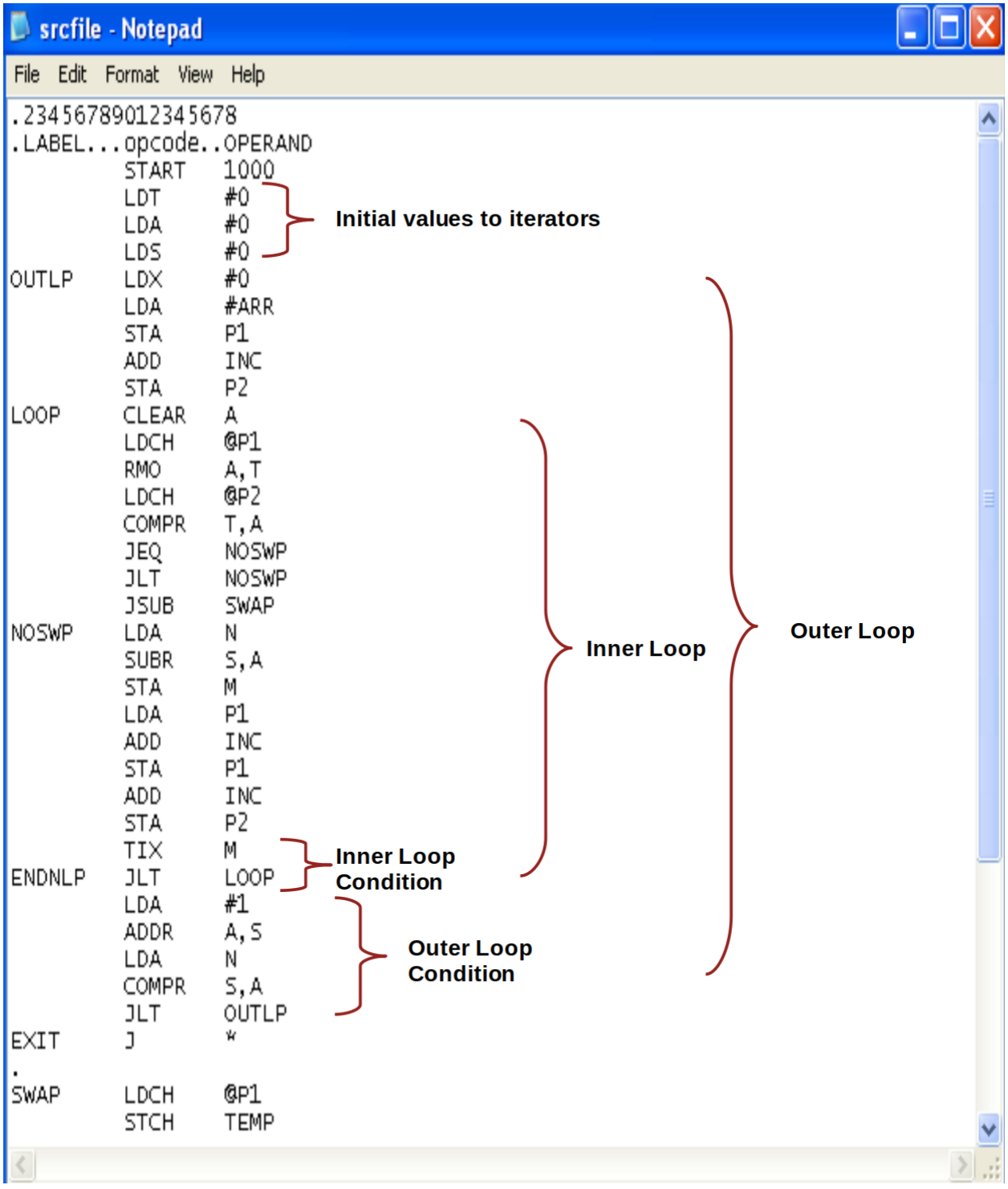
i : X register

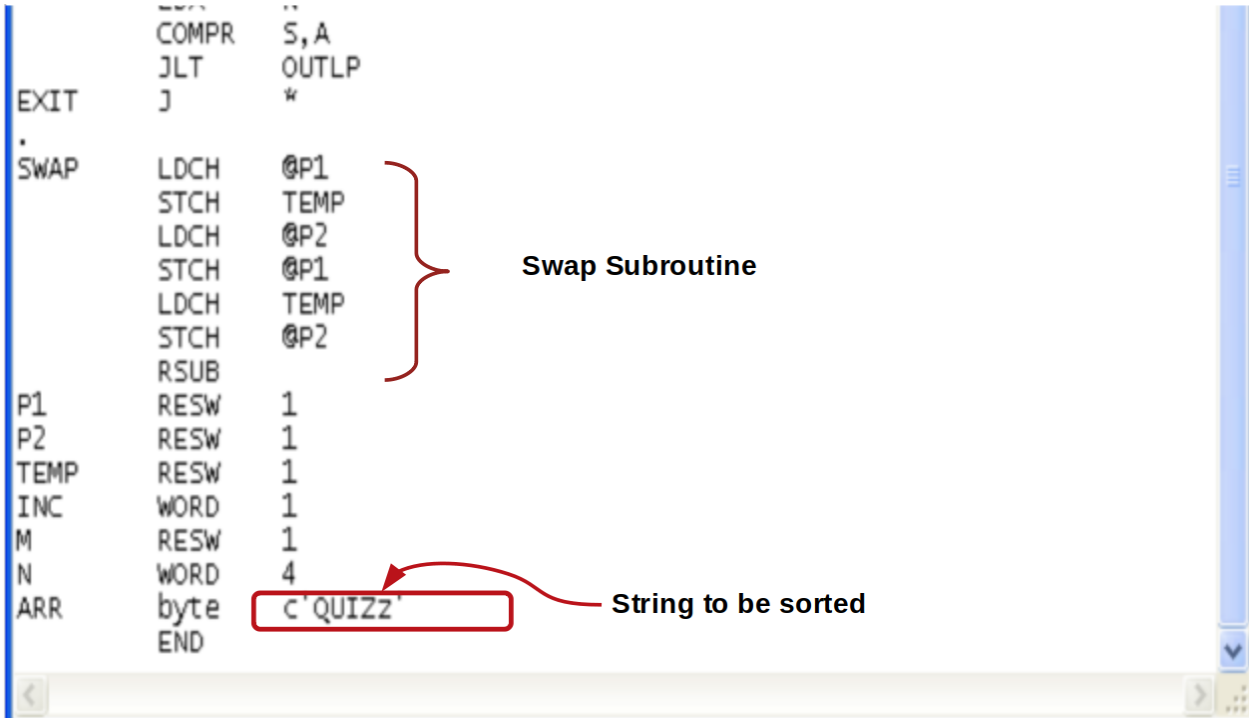
&arr[j] : P1

&arr[j+1] : P2

n -1 : N

n -i-1 : M


srcfile - Notepad
File Edit Format View Help
.23456789012345678
.LABEL...opcode..OPERAND
START 1000
LDT #0
LDA #0
LDS #0
OUTLP LDX #0
LDA #ARR
STA P1
ADD INC
STA P2
LOOP CLEAR A
LDCH @P1
RMO A,T
LDCH @P2
COMPR T,A
JEQ NOSWP
JLT NOSWP
JSUB SWAP
NOSWP LDA N
SUBR S,A
STA M
LDA P1
ADD INC
STA P1
ADD INC
STA P2
TIX M
ENDNLP JLT LOOP
LDA #1
ADDR A,S
LDA N
COMPR S,A
JLT OUTLP
EXIT J *
.
SWAP LDCH @P1
STCH TEMP
Initial values to iterators
Inner Loop
Outer Loop
Inner Loop Condition
Outer Loop Condition

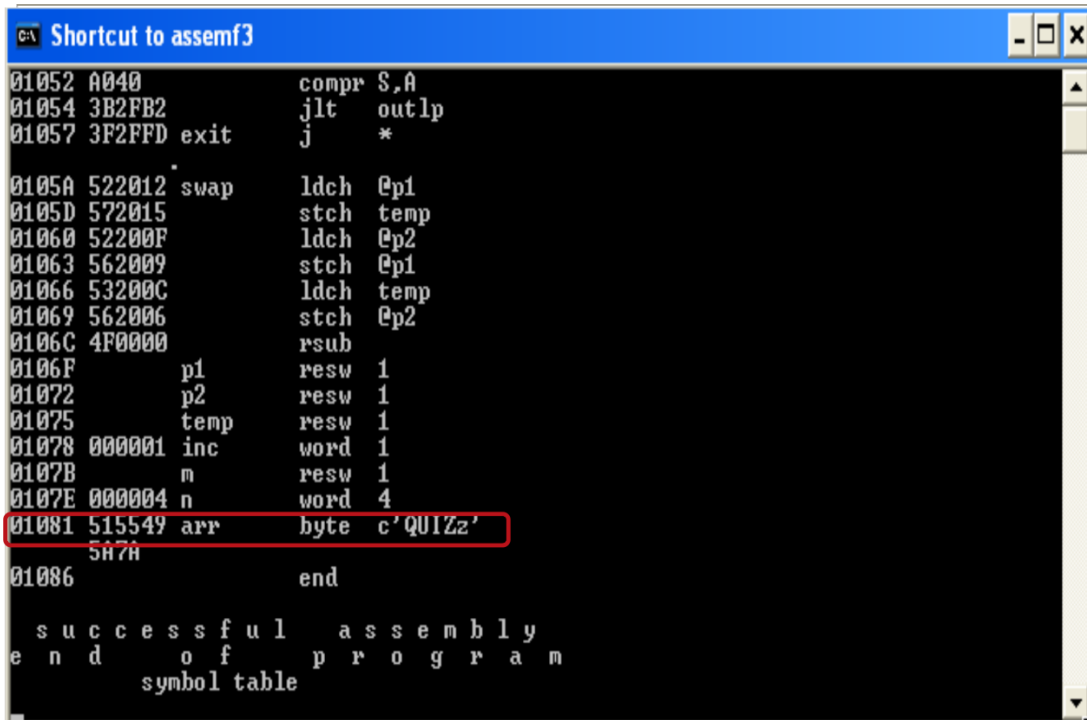


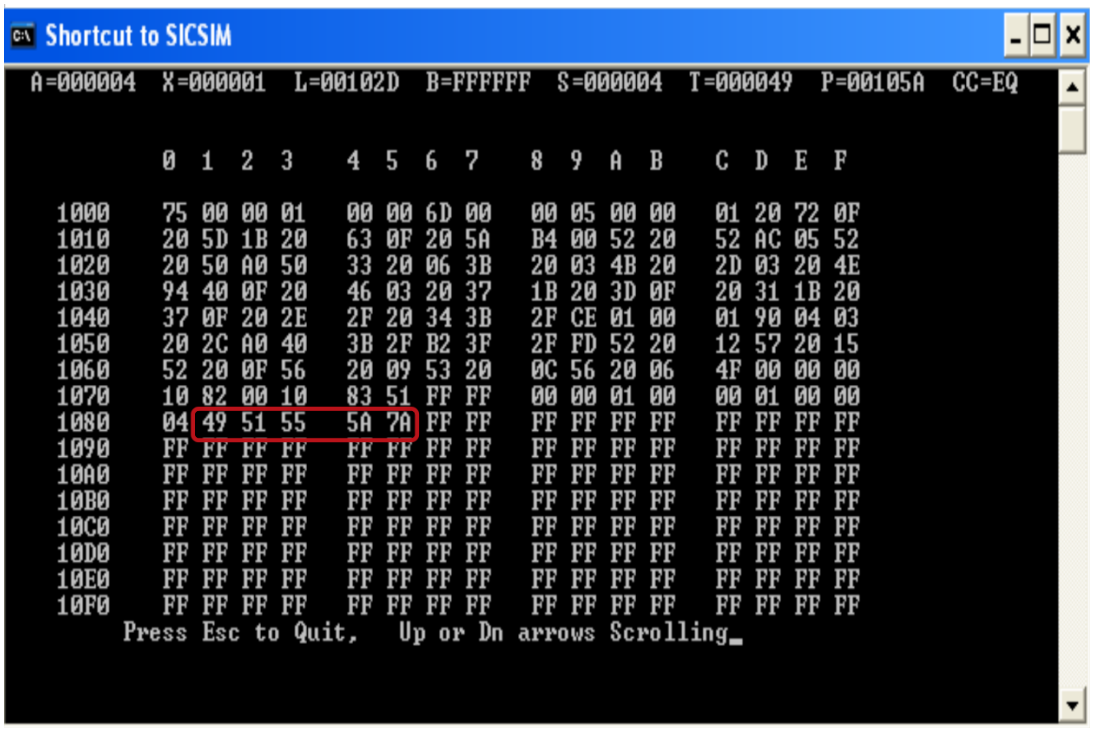
Assumptions :

Assume the length of the string to be sorted is already given, so n-1 (N) is given.

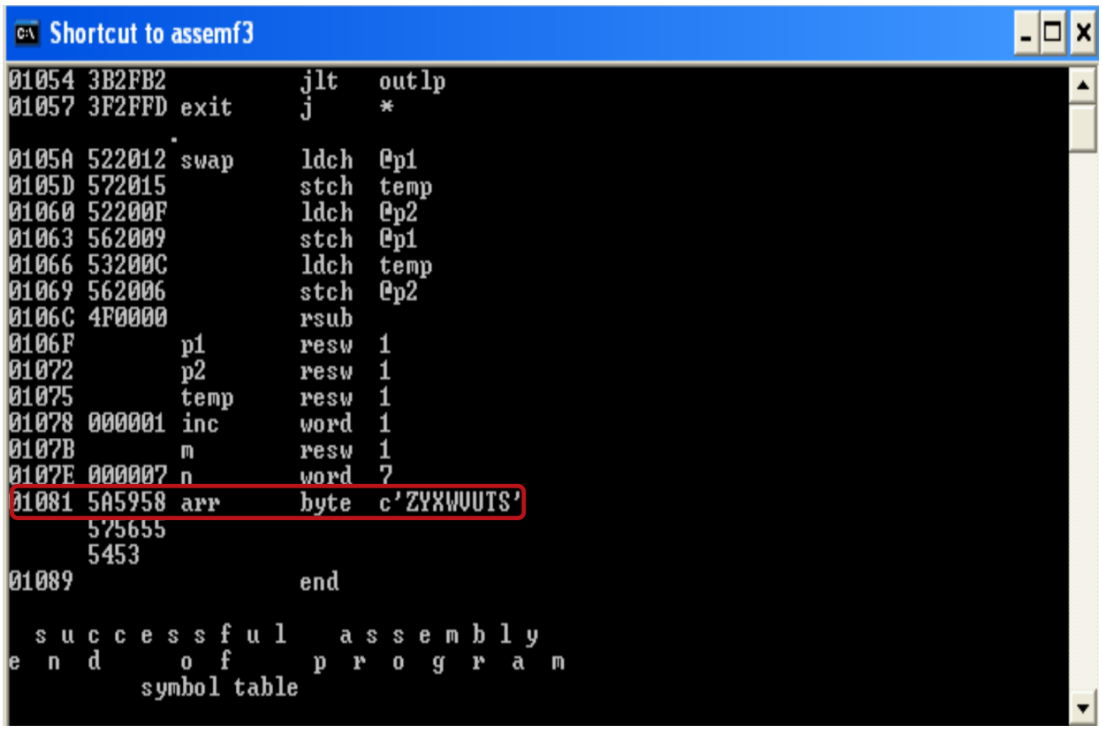
A string is declared in the program, and not read from input device.

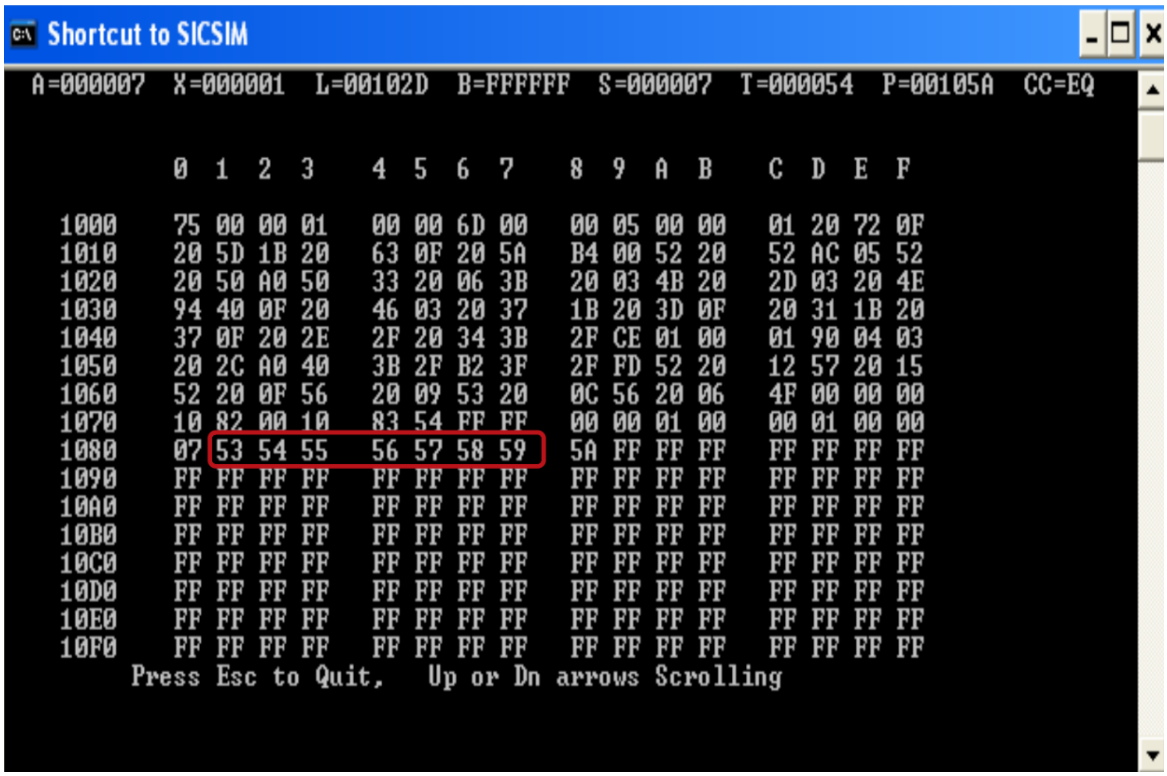
Case 1 :


Shortcut to assemf3
01052 A040 compr S,A
01054 3B2FB2 jlt outlp
01057 3F2FFD exit j *
0105A 522012 swap ldch @p1
0105D 572015 stch temp
01060 52200F ldch @p2
01063 562009 stch @p1
01066 53200C ldch temp
01069 562006 stch @p2
0106C 4F0000 rsub
0106F p1 resw 1
01072 p2 resw 1
01075 temp resw 1
01078 000001 inc word 1
0107B m resw 1
0107E 000004 n word 4
01081 515549 arr byte c'QUIZz'
01086 end
successful assembly
end of program
symbol table

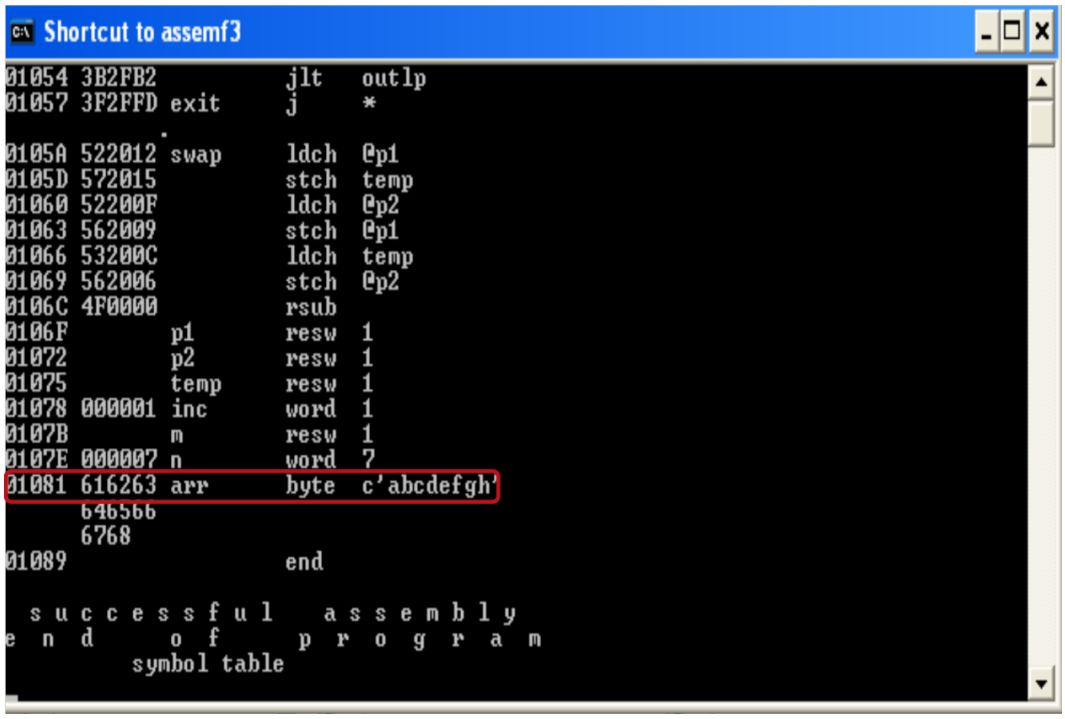

Shortcut to SICSIM
A=000004 X=000001 L=00102D B=FFFFFF S=000004 T=000049 P=00105A CC=EQ
0 1 2 3 4 5 6 7 8 9 A B C D E F
1000 75 00 00 01 00 00 6D 00 00 05 00 00 01 20 72 0F
1010 20 5D 1B 20 63 0F 20 5A B4 00 52 20 52 AC 05 52
1020 20 50 A0 50 33 20 06 3B 20 03 4B 20 2D 03 20 4E
1030 94 40 0F 20 46 03 20 37 1B 20 3D 0F 20 31 1B 20
1040 37 0F 20 2E 2F 20 34 3B 2F CE 01 00 01 90 04 03
1050 20 2C A0 40 3B 2F B2 3F 2F FD 52 20 12 57 20 15
1060 52 20 0F 56 20 09 53 20 0C 56 20 06 4F 00 00 00
1070 10 82 00 10 83 51 FF FF 00 00 01 00 00 01 00 00
1080 04 49 51 55 5A 7A FF FF FF FF FF FF FF FF FF FF
1090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Press Esc to Quit, Up or Dn arrows Scrolling

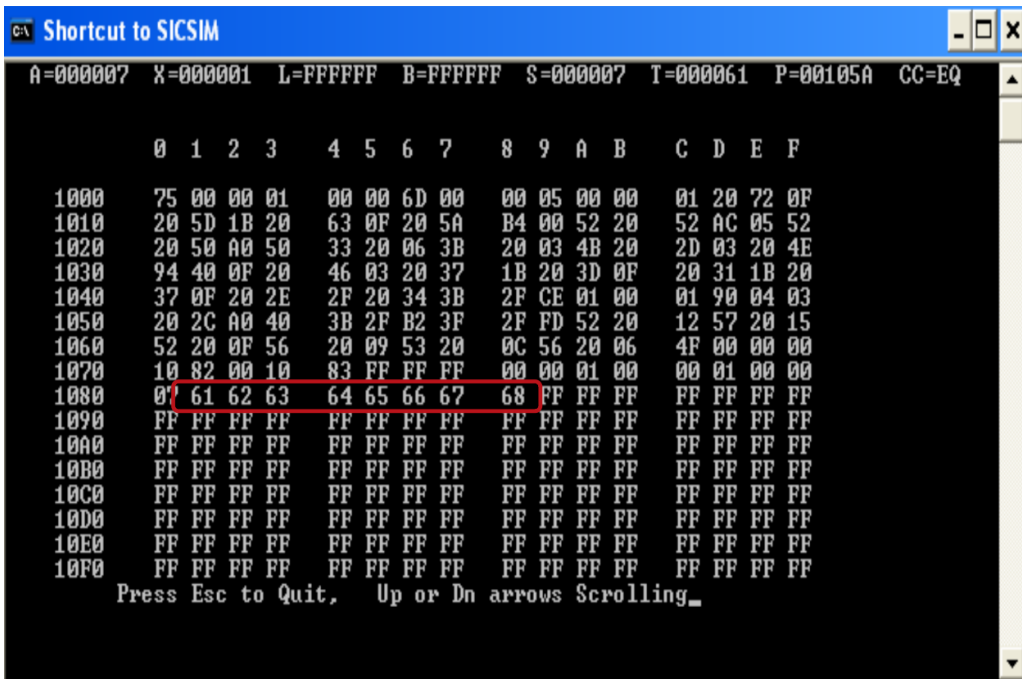
Case 2 :


Shortcut to assemf3
01054 3B2FB2 jlt outlp
01057 3F2FFD exit j *
01054 522012 swap ldch @p1
01057 572015 stch temp
01060 52200F ldch @p2
01063 562009 stch @p1
01066 53200C ldch temp
01069 562006 stch @p2
0106C 4F0000 rsub
0106F p1 resw 1
01072 p2 resw 1
01075 temp resw 1
01078 000001 inc word 1
0107B m resw 1
0107E 000007 n word 7
01081 5A5958 arr byte c'ZYXWVUTS'
575655
5453
01089 end
successful assembly
end of program
symbol table


Shortcut to SICSIM
A=000007 X=000001 L=00102D B=FFFFFF S=000007 T=000054 P=00105A CC=EQ
0 1 2 3 4 5 6 7 8 9 A B C D E F
1000 75 00 00 01 00 00 6D 00 00 05 00 00 01 20 72 0F
1010 20 5D 1B 20 63 0F 20 5A B4 00 52 20 52 AC 05 52
1020 20 50 A0 50 33 20 06 3B 20 03 4B 20 2D 03 20 4E
1030 94 40 0F 20 46 03 20 37 1B 20 3D 0F 20 31 1B 20
1040 37 0F 20 2E 2F 20 34 3B 2F CE 01 00 01 90 04 03
1050 20 2C A0 40 3B 2F B2 3F 2F FD 52 20 12 57 20 15
1060 52 20 0F 56 20 09 53 20 0C 56 20 06 4F 00 00 00
1070 10 82 00 10 83 54 FF FF 00 00 01 00 00 01 00 00
1080 07 53 54 55 56 57 58 59 5A FF FF FF FF FF FF FF
1090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Press Esc to Quit, Up or Dn arrows Scrolling

Case 3 :


Shortcut to assemf3
01054 3B2FB2 jlt outlp
01057 3F2FFD exit j *
01060 52200F ldch @p2
01063 562009 stch @p1
01066 53200C ldch temp
01069 562006 stch @p2
0106C 4F0000 rsub
0106F p1 resw 1
01072 p2 resw 1
01075 temp resw 1
01078 000001 inc word 1
0107B m resw 1
0107E 000007 n word 7
01081 616263 arr byte c'abcdefgh'
646566
6768
01089 end
s u c c e s s f u l a s s e m b l y
e n d o f p r o g r a m
symbol table


Shortcut to SICSIM
A=000007 X=000001 L=FFFFFF B=FFFFFF S=000007 T=000061 P=00105A CC=EQ
0 1 2 3 4 5 6 7 8 9 A B C D E F
1000 75 00 00 01 00 00 6D 00 00 05 00 00 01 20 72 0F
1010 20 5D 1B 20 63 0F 20 5A B4 00 52 20 52 AC 05 52
1020 20 50 A0 50 33 20 06 3B 20 03 4B 20 2D 03 20 4E
1030 94 40 0F 20 46 03 20 37 1B 20 3D 0F 20 31 1B 20
1040 37 0F 20 2E 2F 20 34 3B 2F CE 01 00 01 90 04 03
1050 20 2C A0 40 3B 2F B2 3F 2F FD 52 20 12 57 20 15
1060 52 20 0F 56 20 09 53 20 0C 56 20 06 4F 00 00 00
1070 10 82 00 10 83 FF FF FF 00 00 01 00 00 01 00 00
1080 07 61 62 63 64 65 66 67 68 FF FF FF FF FF FF FF
1090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Press Esc to Quit, Up or Dn arrows Scrolling