Microprocessor and Interfacing

LAB MANUAL: 5

Example:

This instruction compares two operands, discarding the results and setting the flags

Syntax: cmp, op1, op2

op1: register or memory

op2: register, memory, or immediate

Flags Affected: OF, SF, ZF, AF, PF, CF

Solution:

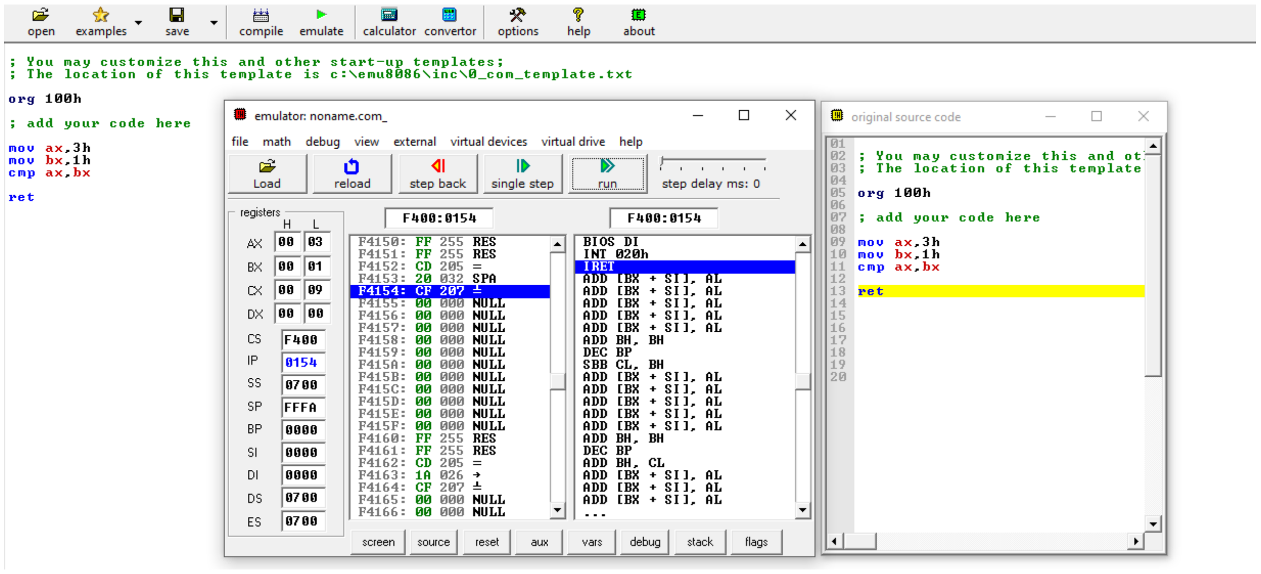
In case When op1 > op2:

In this case when op1>op2, the two registers ax having 3h value and bx having 1h value are compared using cmp command which will subtract the bx register value from ax register value and in the given case the result is 2h it will not affect the execution of program but cmp command will update the status flags on the basis of comparison between them but here the result is non zero so no flag is affected at the end as shown in following figure:

SOURCE CODE:

```
mov ax,3h
mov bx,1h
cmp ax,bx
```

Output:



```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

; add your code here

mov ax,3h
mov bx,1h
cmp ax,bx

ret
```

```
registers
     H   L
AX  00  03
BX  00  01
CX  00  09
DX  00  00
CS  F400
IP  0154
SS  0700
SP  FFFA
BP  0000
SI  0000
DI  0000
DS  0700
ES  0700
```

```
F400:0154

F4150: FF 255 RES
F4151: FF 255 RES
F4152: CD 205 =
F4153: 20 032 SPA
F4154: CF 207 ±
F4155: 00 000 NULL
F4156: 00 000 NULL
F4157: 00 000 NULL
F4158: 00 000 NULL
F4159: 00 000 NULL
F415A: 00 000 NULL
F415B: 00 000 NULL
F415C: 00 000 NULL
F415D: 00 000 NULL
F415E: 00 000 NULL
F415F: 00 000 NULL
F4160: FF 255 RES
F4161: FF 255 RES
F4162: CD 205 =
F4163: 1A 026 →
F4164: CF 207 ±
F4165: 00 000 NULL
F4166: 00 000 NULL
```

```
F400:0154

BIOS DI
INT 020h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
SBB CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
; You may customize this and ot
; The location of this template

org 100h

; add your code here

mov ax,3h
mov bx,1h
cmp ax,bx

ret
```

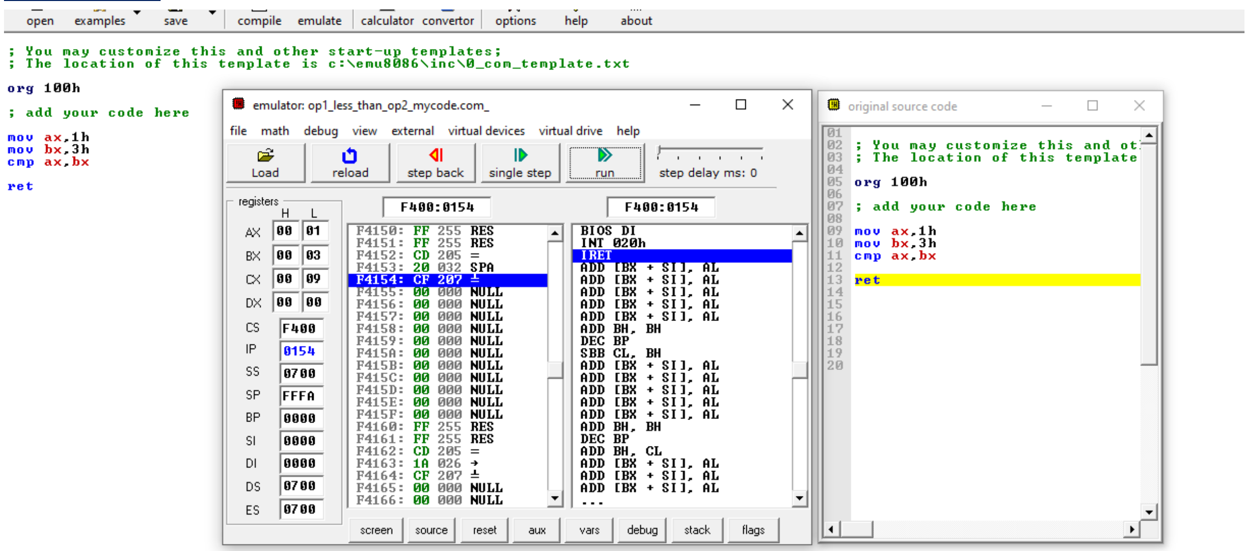
In case When op1<op2:

Since Auxiliary Flag is used as CF but when working with BCD. So AF will be set when we have overflow or underflow on in BCD calculations here is the underflow condition and carry flag after comparison will be set to 1 also due to logical cmp operation over the register ax and bx and signed flag will also set to 1 due to signed (negative) output value after comparison between ax and bx as shown in following figure:

SOURCE CODE:

mov ax,1h
mov bx,3h
cmp ax,bx

Output:


open examples save compile emulate calculator convertor options help about
emulator: op1_less_than_op2_mycode.com_
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
screen source reset aux vars debug stack flags
original source code

```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

; add your code here

mov ax,1h
mov bx,3h
cmp ax,bx

ret
```

```
registers
     H   L
AX   00  01
BX   00  03
CX   00  09
DX   00  00
CS   F400
IP   0154
SS   0700
SP   FFFA
BP   0000
SI   0000
DI   0000
DS   0700
ES   0700
```

```
F400:0154
F4150: FF 255 RES
F4151: FF 255 RES
F4152: CD 205 =
F4153: 20 032 SPA
F4154: CF 207 ╧
F4155: 00 000 NULL
F4156: 00 000 NULL
F4157: 00 000 NULL
F4158: 00 000 NULL
F4159: 00 000 NULL
F415A: 00 000 NULL
F415B: 00 000 NULL
F415C: 00 000 NULL
F415D: 00 000 NULL
F415E: 00 000 NULL
F415F: 00 000 NULL
F4160: FF 255 RES
F4161: FF 255 RES
F4162: CD 205 =
F4163: 1A 026 →
F4164: CF 207 ╧
F4165: 00 000 NULL
F4166: 00 000 NULL
```

```
F400:0154
BIOS DI
INT 020h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
SBB CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
; You may customize this and ot
; The location of this template

org 100h

; add your code here

mov ax,1h
mov bx,3h
cmp ax,bx

ret
```

In case When op1=op2:

In this case the zero flag at the end will set to 1 because zero flag in case of cmp command setted to 1 if after comparison between two operands the resulting output is equal to zero since in given case this condition is fulfilled so it will be set to 1 and here due to parity even the parity flag is setted to 1 as shown in following figure:

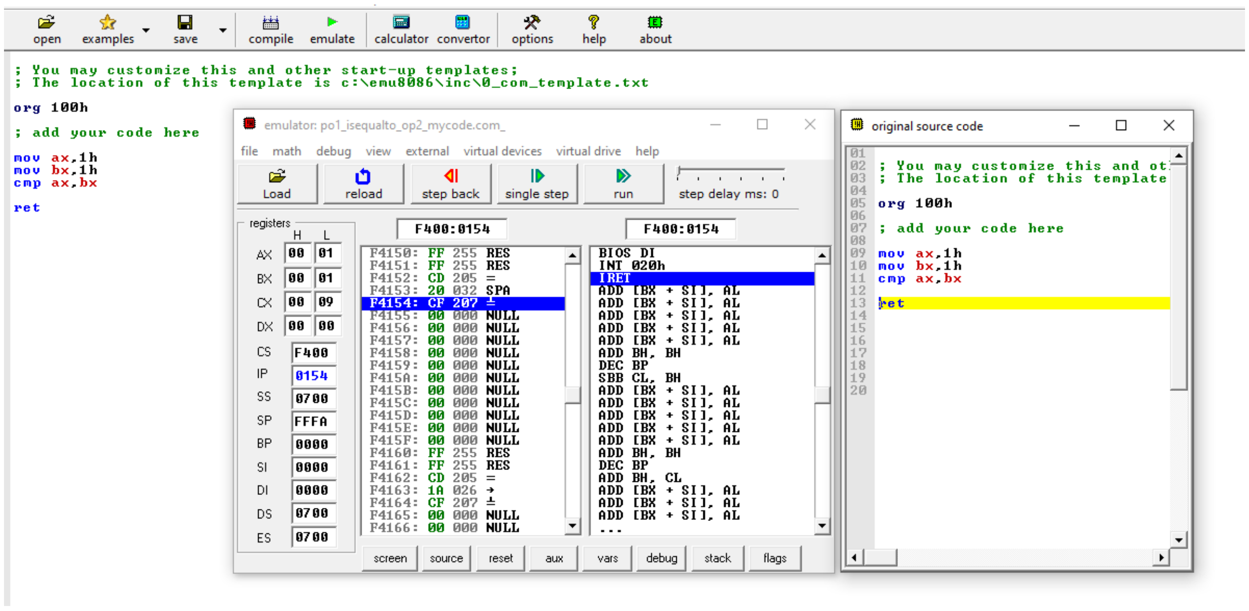
SOURCE CODE:

mov ax,1h

mov bx,1h

cmp ax,bx

Output:



```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

; add your code here

mov ax,1h
mov bx,1h
cmp ax,bx

ret
```

```
registers
     H   L
AX  00  01
BX  00  01
CX  00  09
DX  00  00
CS  F400
IP  0154
SS  0700
SP  FFFA
BP  0000
SI  0000
DI  0000
DS  0700
ES  0700
```

```
F400:0154
F4150: FF 255 RES
F4151: FF 255 RES
F4152: CD 205 =
F4153: 20 032 SPA
F4154: CF 207 ±
F4155: 00 000 NULL
F4156: 00 000 NULL
F4157: 00 000 NULL
F4158: 00 000 NULL
F4159: 00 000 NULL
F415A: 00 000 NULL
F415B: 00 000 NULL
F415C: 00 000 NULL
F415D: 00 000 NULL
F415E: 00 000 NULL
F415F: 00 000 NULL
F4160: FF 255 RES
F4161: FF 255 RES
F4162: CD 205 =
F4163: 1A 026 →
F4164: CF 207 ±
F4165: 00 000 NULL
F4166: 00 000 NULL
```

```
F400:0154
BIOS DI
INT 020h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
SBB CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
; You may customize this and ot
; The location of this template

org 100h

; add your code here

mov ax,1h
mov bx,1h
cmp ax,bx

ret
```

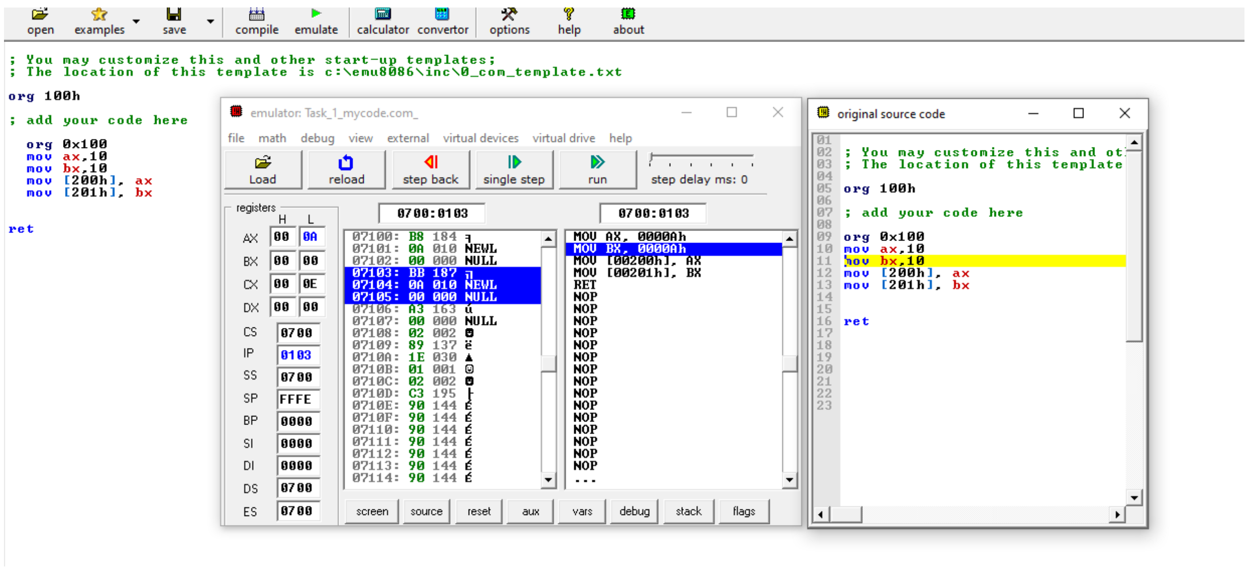
Execute the following tasks.

Task1:

Run the following code and observe the registers/ memory locations:

```
org   0x100
mov   ax,10
mov bx,10

mov  [200h],  ax
mov  [201h],  bx
ret
```

Step#1:



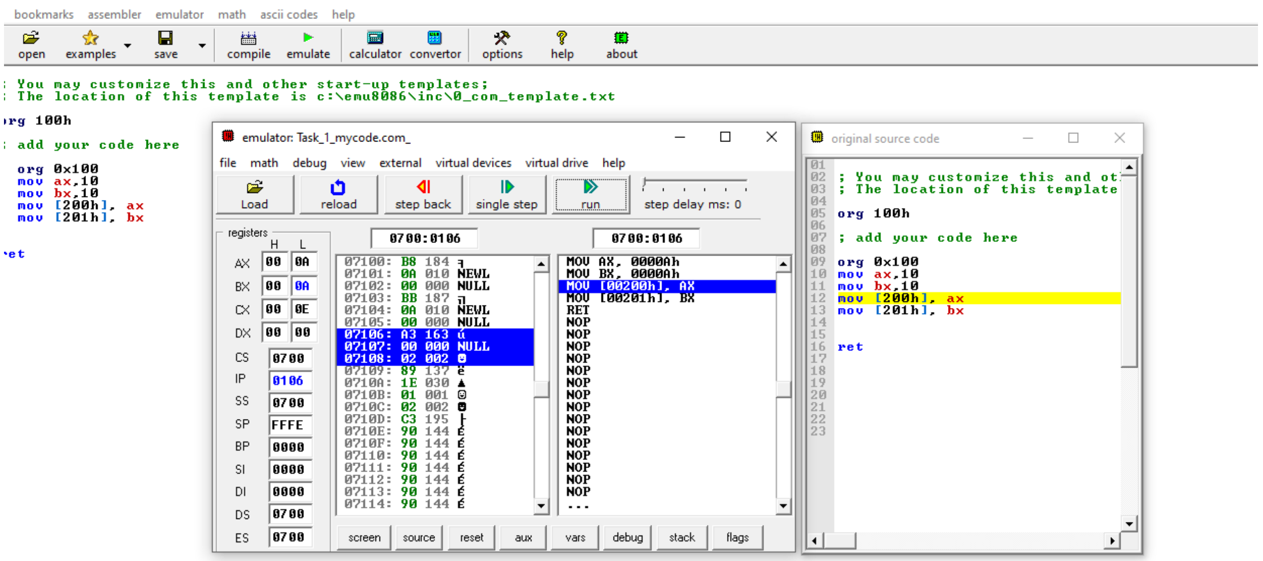
```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

; add your code here

  org 0x100
  mov ax,10
  mov bx,10
  mov [200h], ax
  mov [201h], bx


ret
```

```
registers   H   L
AX         00  0A
BX         00  00
CX         00  0E
DX         00  00
CS         0700
IP         0103
SS         0700
SP         FFFE
BP         0000
SI         0000
DI         0000
DS         0700
ES         0700
```

```
0700:0103                     0700:0103

07100: B8 184 ╕               MOV AX, 0000Ah
07101: 0A 010 NEWL            MOV BX, 0000Ah
07102: 00 000 NULL            MOV [00200h], AX
07103: BB 187 ╗               MOV [00201h], BX
07104: 0A 010 NEWL            RET
07105: 00 000 NULL            NOP
07106: A3 163 ú               NOP
07107: 00 000 NULL            NOP
07108: 02 002 ☻               NOP
07109: 89 137 ë               NOP
0710A: 1E 030 ▲               NOP
0710B: 01 001 ☺               NOP
0710C: 02 002 ☻               NOP
0710D: C3 195 ├               NOP
0710E: 90 144 É               NOP
0710F: 90 144 É               NOP
07110: 90 144 É               NOP
07111: 90 144 É               NOP
07112: 90 144 É               NOP
07113: 90 144 É               NOP
07114: 90 144 É               ...
```

```
; You may customize this and ot
; The location of this template

org 100h

; add your code here

org 0x100
mov ax,10
mov bx,10
mov [200h], ax
mov [201h], bx


ret
```

Step#2:



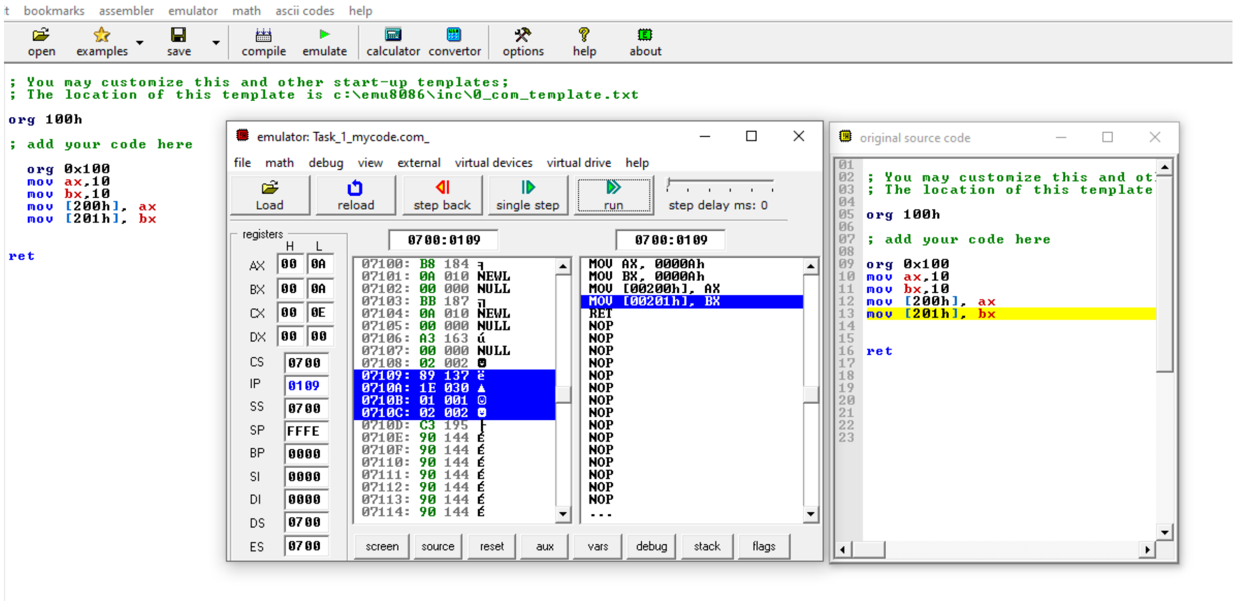
```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

; add your code here

  org 0x100
  mov ax,10
  mov bx,10
  mov [200h], ax
  mov [201h], bx


ret
```

```
registers   H   L
AX         00  0A
BX         00  0A
CX         00  0E
DX         00  00
CS         0700
IP         0106
SS         0700
SP         FFFE
BP         0000
SI         0000
DI         0000
DS         0700
ES         0700
```

```
0700:0106                     0700:0106

07100: B8 184 ╕               MOV AX, 0000Ah
07101: 0A 010 NEWL            MOV BX, 0000Ah
07102: 00 000 NULL            MOV [00200h], AX
07103: BB 187 ╗               MOV [00201h], BX
07104: 0A 010 NEWL            RET
07105: 00 000 NULL            NOP
07106: A3 163 ú               NOP
07107: 00 000 NULL            NOP
07108: 02 002 ☻               NOP
07109: 89 137 ë               NOP
0710A: 1E 030 ▲               NOP
0710B: 01 001 ☺               NOP
0710C: 02 002 ☻               NOP
0710D: C3 195 ├               NOP
0710E: 90 144 É               NOP
0710F: 90 144 É               NOP
07110: 90 144 É               NOP
07111: 90 144 É               NOP
07112: 90 144 É               NOP
07113: 90 144 É               NOP
07114: 90 144 É               ...
```

```
; You may customize this and ot
; The location of this template

org 100h

; add your code here

org 0x100
mov ax,10
mov bx,10
mov [200h], ax
mov [201h], bx


ret
```

Step#3:



```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

; add your code here

   org 0x100
   mov ax,10
   mov bx,10
   mov [200h], ax
   mov [201h], bx


ret
```

emulator: Task_1_mycode.com_

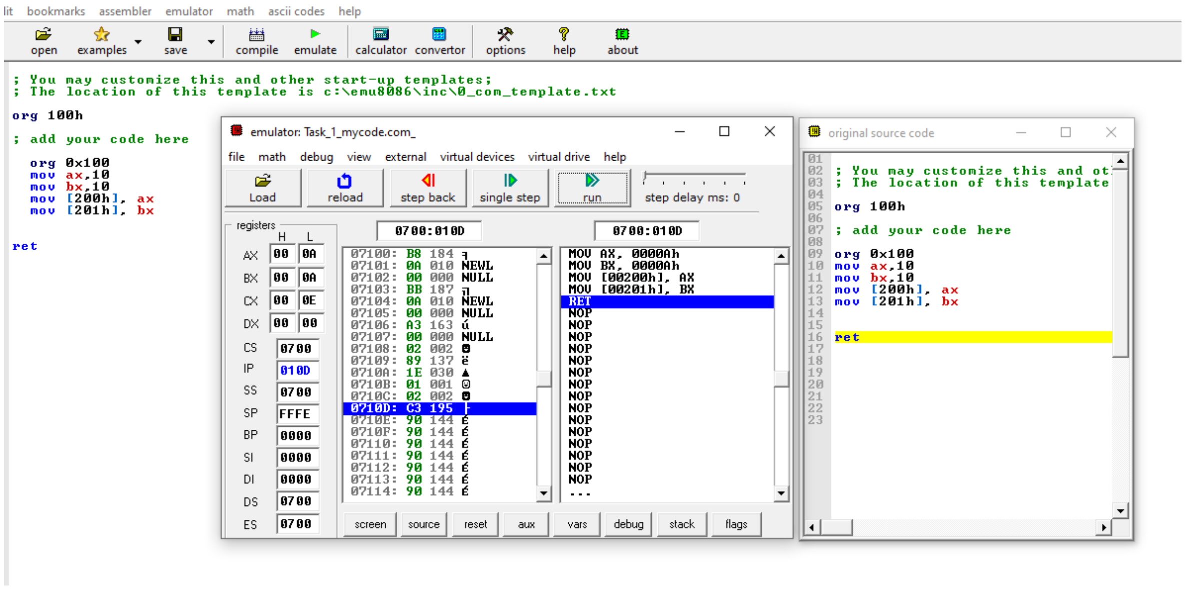
```
registers  H   L
AX        00  0A
BX        00  0A
CX        00  0E
DX        00  00
CS        0700
IP        0109
SS        0700
SP        FFFE
BP        0000
SI        0000
DI        0000
DS        0700
ES        0700
```

```
0700:0109                  0700:0109

07100: B8 184 ╕            MOV  AX, 0000Ah
07101: 0A 010 NEWL         MOV  BX, 0000Ah
07102: 00 000 NULL         MOV  [00200h], AX
07103: BB 187 ╗            MOV  [00201h], BX
07104: 0A 010 NEWL         RET
07105: 00 000 NULL         NOP
07106: A3 163 ú            NOP
07107: 00 000 NULL         NOP
07108: 02 002 ☻            NOP
07109: 89 137 ë            NOP
0710A: 1E 030 ▲            NOP
0710B: 01 001 ☺            NOP
0710C: 02 002 ☻            NOP
0710D: C3 195 ├            NOP
0710E: 90 144 É            NOP
0710F: 90 144 É            NOP
07110: 90 144 É            NOP
07111: 90 144 É            NOP
07112: 90 144 É            NOP
07113: 90 144 É            NOP
07114: 90 144 É            ...
```

original source code

```
; You may customize this and ot
; The location of this template

org 100h

; add your code here

org 0x100
mov ax,10
mov bx,10
mov [200h], ax
mov [201h], bx


ret
```

Step#4:



```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

; add your code here

   org 0x100
   mov ax,10
   mov bx,10
   mov [200h], ax
   mov [201h], bx


ret
```

emulator: Task_1_mycode.com_

```
registers  H   L
AX        00  0A
BX        00  0A
CX        00  0E
DX        00  00
CS        0700
IP        010D
SS        0700
SP        FFFE
BP        0000
SI        0000
DI        0000
DS        0700
ES        0700
```

```
0700:010D                  0700:010D

07100: B8 184 ╕            MOV  AX, 0000Ah
07101: 0A 010 NEWL         MOV  BX, 0000Ah
07102: 00 000 NULL         MOV  [00200h], AX
07103: BB 187 ╗            MOV  [00201h], BX
07104: 0A 010 NEWL         RET
07105: 00 000 NULL         NOP
07106: A3 163 ú            NOP
07107: 00 000 NULL         NOP
07108: 02 002 ☻            NOP
07109: 89 137 ë            NOP
0710A: 1E 030 ▲            NOP
0710B: 01 001 ☺            NOP
0710C: 02 002 ☻            NOP
0710D: C3 195 ├            NOP
0710E: 90 144 É            NOP
0710F: 90 144 É            NOP
07110: 90 144 É            NOP
07111: 90 144 É            NOP
07112: 90 144 É            NOP
07113: 90 144 É            NOP
07114: 90 144 É            ...
```

original source code

```
; You may customize this and ot
; The location of this template

org 100h

; add your code here

org 0x100
mov ax,10
mov bx,10
mov [200h], ax
mov [201h], bx


ret
```

TASK 2:

Write a code to add first ten natural numbers starting from the memory location 0200H?

SOURCE CODE:

```
MOV BX,200h
MOV [BX],  01
MOV [BX+1],02
MOV [BX+2],03
MOV [BX+3],04
MOV [BX+4],05
MOV [BX+5],06
MOV [BX+6],07
MOV [BX+7],08
MOV [BX+8],09
MOV [BX+9],10
ADD AL,[BX]
ADD AL,[BX+1]
ADD AL,[BX+2]
ADD AL,[BX+3]
ADD AL,[BX+4]
ADD AL,[BX+5]
ADD AL,[BX+6]
ADD AL,[BX+7]
```

ADD AL,[BX+8]

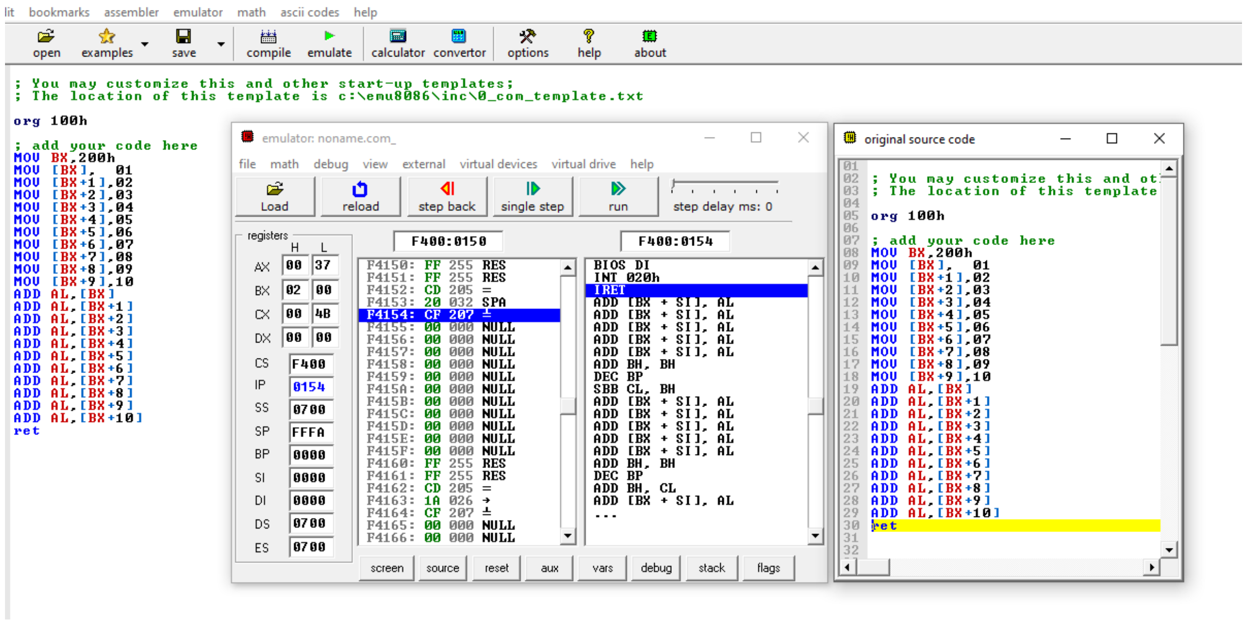
ADD AL,[BX+9]

ADD AL,[BX+10]

Step:

At the end the Al part of AX register has Final value as shown in following figure;

Output:



```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

; add your code here
MOV BX,200h
MOV [BX],  01
MOV [BX+1],02
MOV [BX+2],03
MOV [BX+3],04
MOV [BX+4],05
MOV [BX+5],06
MOV [BX+6],07
MOV [BX+7],08
MOV [BX+8],09
MOV [BX+9],10
ADD AL,[BX]
ADD AL,[BX+1]
ADD AL,[BX+2]
ADD AL,[BX+3]
ADD AL,[BX+4]
ADD AL,[BX+5]
ADD AL,[BX+6]
ADD AL,[BX+7]
ADD AL,[BX+8]
ADD AL,[BX+9]
ADD AL,[BX+10]
ret
```

emulator: noname.com_

```
registers
     H   L
AX   00  37
BX   02  00
CX   00  4B
DX   00  00
CS   F400
IP   0154
SS   0700
SP   FFFA
BP   0000
SI   0000
DI   0000
DS   0700
ES   0700
```

```
F400:0150

F4150: FF 255 RES
F4151: FF 255 RES
F4152: CD 205 =
F4153: 20 032 SPA
F4154: CF 207 ±
F4155: 00 000 NULL
F4156: 00 000 NULL
F4157: 00 000 NULL
F4158: 00 000 NULL
F4159: 00 000 NULL
F415A: 00 000 NULL
F415B: 00 000 NULL
F415C: 00 000 NULL
F415D: 00 000 NULL
F415E: 00 000 NULL
F415F: 00 000 NULL
F4160: FF 255 RES
F4161: FF 255 RES
F4162: CD 205 =
F4163: 1A 026 →
F4164: CF 207 ±
F4165: 00 000 NULL
F4166: 00 000 NULL
```

```
F400:0154

BIOS DI
INT 020h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
SBB CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
...
```

original source code

```
; You may customize this and ot
; The location of this template

org 100h

; add your code here
MOV BX,200h
MOV [BX],  01
MOV [BX+1],02
MOV [BX+2],03
MOV [BX+3],04
MOV [BX+4],05
MOV [BX+5],06
MOV [BX+6],07
MOV [BX+7],08
MOV [BX+8],09
MOV [BX+9],10
ADD AL,[BX]
ADD AL,[BX+1]
ADD AL,[BX+2]
ADD AL,[BX+3]
ADD AL,[BX+4]
ADD AL,[BX+5]
ADD AL,[BX+6]
ADD AL,[BX+7]
ADD AL,[BX+8]
ADD AL,[BX+9]
ADD AL,[BX+10]
ret
```

TASK 3:

Write a code to transfer a block of data from source memory block to destination memory block. Use any starting address.

SOURCE CODE:

LEA SI,Array1

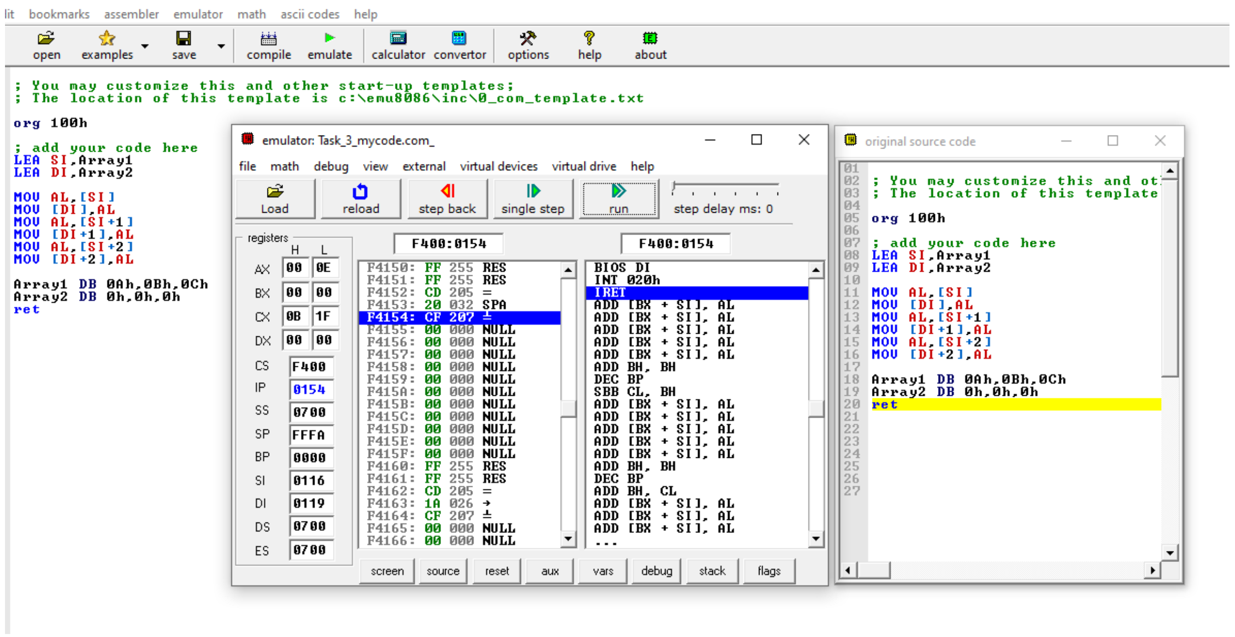
LEA DI,Array2

```
MOV AL,[SI]

MOV [DI],AL

MOV AL,[SI+1]

MOV [DI+1],AL

MOV AL,[SI+2]

MOV [DI+2],AL

ret

Array1 DB 0Ah,0Bh,0Ch

Array2 DB 0h,0h,0h
```

Step:

In the following figure we can see that the block data from array1 moved to array 2 having starting address 011A in memory

Output:


lit bookmarks assembler emulator math ascii codes help
open examples save compile emulate calculator convertor options help about
emulator: Task_3_mycode.com_
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
screen source reset aux vars debug stack flags
original source code

```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

; add your code here
LEA SI,Array1
LEA DI,Array2

MOV AL,[SI]
MOV [DI],AL
MOV AL,[SI+1]
MOV [DI+1],AL
MOV AL,[SI+2]
MOV [DI+2],AL

Array1 DB 0Ah,0Bh,0Ch
Array2 DB 0h,0h,0h
ret
```

```
registers
     H   L
AX   00  0E
BX   00  00
CX   0B  1F
DX   00  00
CS   F400
IP   0154
SS   0700
SP   FFFA
BP   0000
SI   0116
DI   0119
DS   0700
ES   0700
```

```
F400:0154
F4150: FF 255 RES
F4151: FF 255 RES
F4152: CD 205 =
F4153: 20 032 SPA
F4154: CF 207 ±
F4155: 00 000 NULL
F4156: 00 000 NULL
F4157: 00 000 NULL
F4158: 00 000 NULL
F4159: 00 000 NULL
F415A: 00 000 NULL
F415B: 00 000 NULL
F415C: 00 000 NULL
F415D: 00 000 NULL
F415E: 00 000 NULL
F415F: 00 000 NULL
F4160: FF 255 RES
F4161: FF 255 RES
F4162: CD 205 =
F4163: 1A 026 →
F4164: CF 207 ±
F4165: 00 000 NULL
F4166: 00 000 NULL
```

```
F400:0154
BIOS DI
INT 020h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
SBB CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
; You may customize this and ot
; The location of this template

org 100h

; add your code here
LEA SI,Array1
LEA DI,Array2

MOV AL,[SI]
MOV [DI],AL
MOV AL,[SI+1]
MOV [DI+1],AL
MOV AL,[SI+2]
MOV [DI+2],AL

Array1 DB 0Ah,0Bh,0Ch
Array2 DB 0h,0h,0h
ret
```

TASK 4:

Write a code to perform addition on 8-bit data stored in consecutive memory locations (10) and store result in next memory locations. Use any starting address.

SOURCE CODE:

```
MOV  BX,200h

MOV [BX] , 00000001B

MOV [BX+1],0FH

MOV [BX+2],0AH

MOV [BX+3],00000101B

MOV [BX+4],00000011B

MOV [BX+5],0CH

MOV [BX+6],0BH

MOV [BX+7],11H

MOV [BX+8],12H

MOV [BX+9],16H

ADD AL,[BX]

ADD AL,[BX+1]

ADD AL,[BX+2]

ADD AL,[BX+3]

ADD AL,[BX+4]

ADD AL,[BX+5]
```

ADD AL,[BX+6]

ADD AL,[BX+7]

ADD AL,[BX+8]

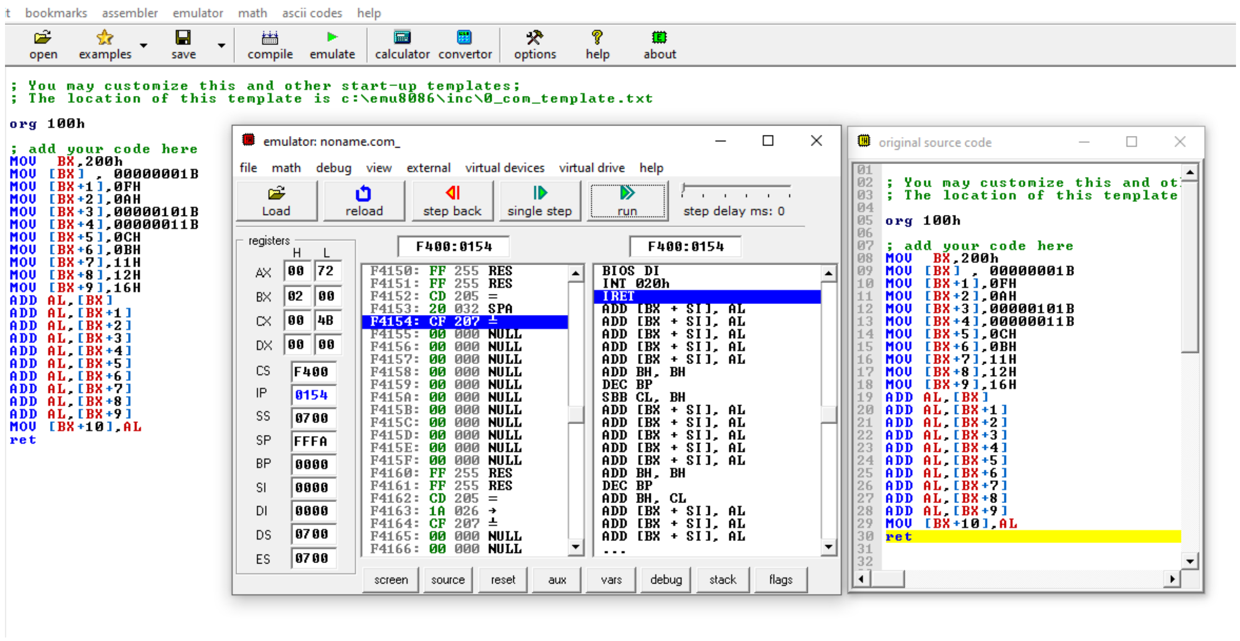
ADD AL,[BX+9]

MOV [BX+10],AL

Step:

In the following figure we can see that we have performed addition on 8-bit data stored in consecutive memory locations (10) and store result in next memory locations;

Output:



t bookmarks assembler emulator math ascii codes help
open examples save compile emulate calculator convertor options help about
emulator: noname.com_
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
screen source reset aux vars debug stack flags
original source code

```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

; add your code here
MOV  BX,200h
MOV [BX] , 00000001B
MOV [BX+1],0FH
MOV [BX+2],0AH
MOV [BX+3],00000101B
MOV [BX+4],00000011B
MOV [BX+5],0CH
MOV [BX+6],0BH
MOV [BX+7],11H
MOV [BX+8],12H
MOV [BX+9],16H
ADD AL,[BX]
ADD AL,[BX+1]
ADD AL,[BX+2]
ADD AL,[BX+3]
ADD AL,[BX+4]
ADD AL,[BX+5]
ADD AL,[BX+6]
ADD AL,[BX+7]
ADD AL,[BX+8]
ADD AL,[BX+9]
MOV [BX+10],AL
ret
```

```
registers
     H   L
AX  00  72
BX  02  00
CX  00  4B
DX  00  00
CS  F400
IP  0154
SS  0700
SP  FFFA
BP  0000
SI  0000
DI  0000
DS  0700
ES  0700
```

```
F400:0154
F4150: FF 255 RES
F4151: FF 255 RES
F4152: CD 205 =
F4153: 20 032 SPA
F4154: CF 207 ╧
F4155: 00 000 NULL
F4156: 00 000 NULL
F4157: 00 000 NULL
F4158: 00 000 NULL
F4159: 00 000 NULL
F415A: 00 000 NULL
F415B: 00 000 NULL
F415C: 00 000 NULL
F415D: 00 000 NULL
F415E: 00 000 NULL
F415F: 00 000 NULL
F4160: FF 255 RES
F4161: FF 255 RES
F4162: CD 205 =
F4163: 1A 026 →
F4164: CF 207 ╧
F4165: 00 000 NULL
F4166: 00 000 NULL
```

```
F400:0154
BIOS DI
INT 020h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
SBB CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
; You may customize this and ot
; The location of this template

org 100h

; add your code here
MOV  BX,200h
MOV [BX] , 00000001B
MOV [BX+1],0FH
MOV [BX+2],0AH
MOV [BX+3],00000101B
MOV [BX+4],00000011B
MOV [BX+5],0CH
MOV [BX+6],0BH
MOV [BX+7],11H
MOV [BX+8],12H
MOV [BX+9],16H
ADD AL,[BX]
ADD AL,[BX+1]
ADD AL,[BX+2]
ADD AL,[BX+3]
ADD AL,[BX+4]
ADD AL,[BX+5]
ADD AL,[BX+6]
ADD AL,[BX+7]
ADD AL,[BX+8]
ADD AL,[BX+9]
MOV [BX+10],AL
ret
```

--THE END-------------------------------------