**WAQAR AHMED**

**20P-0750**

**Lab-Report 08**

**Subroutines:**

**[org 0x0100]**

**jmp start**

**num: dw 20,15,5,10**

**swap: db 0**

**bubblesort:**

**dec cx**

**shl cx,1**

**main:**

**mov si,0**

**mov byte[swap],0**

**innerloop:**

**mov ax,[ bx + si]**

**cmp ax,[bx + si + 2]**

**jbe noswap**

**mov dx,[bx+ si + 2]**

**mov [bx + si],dx**

**mov [bx + si +2],ax**

**mov byte[swap],1**

**noswap:**

**add si,2**

**cmp si , cx**

**jne innerloop**

**cmp byte[swap],1**

**je main**

**ret**

**start:**

**mov bx,num**

**mov cx,4**

**call bubblesort**

**mov ax,0x4c00**

**int 0x21**

# **call function:**

An instruction for temporary diversion is the CALL instruction.

The word call must be familiar to the Programmers from subroutine call in higher level languages . The CALL instruction allows temporary diversion and therefore reusability of code.

CALL takes a label as argument and execution starts from that label, until the RET instruction is encountered and it takes execution back to the instruction following the CALL.

# **Stack :**

The behavior of stack is first in last out manner. It may keep many elements and there is only one way “in” and “out” of the container. When an element is inserted it sits on top of all other elements and when an element is removed the one sitting at top of all others is removed first.