COAL_A_p200165_R9

- **Introduction:**

  In Lab No 9 we have studied about parameterized function and the concept of local variable. The subroutine must have a concept of abstraction.

- **Code No 1:**

  The first code is the code of bubble sort in the previous we have used global variable but now we are going to use local variable. We will push the local variable before calling the function then we access that variable by the use of bp pointer that we will push and with the help of that we will access our all local variable and we can use how much local variable we want.

```
c06-05.asm
1   [org 0x100]
2   jmp start
3   data:   dw  60, 55
4   ; swapflag:   db  0            ; Globals are bad! Let's make this local.
5   swap:
6       push ax  ; ----------------------;
7                                        ;
8       mov  ax, [bx + si]               ;
9       xchg ax, [bx + si + 2]           ;
10      mov  [bx + si], ax               ;
11                                       ;
12      pop ax   ; ----------------------;
13      ret
14  bubblesort:
15      ; handle stack issue for parameters ------------
16      push bp
17      mov  bp, sp
18      sub sp, 2          ; make space on the stack, just below BP
19                         ; only if you want to do local variables
20      push ax
21      push bx
22      push cx
23      push si
24      mov  bx, [bp + 6]   ; address of data to sort
25      mov  cx, [bp + 4]   ; number of elements to sort
26      ; same old code from here ----------------------
27      dec  cx
28      shl  cx, 1
29      mainloop:
30          mov  si, 0                    ; use as array index
31          ; mov  byte[swapflag], 0      ; reset swap flag for this iteration
32          mov  word [bp - 2], 0           ; has to be a word
33          innerloop:
34              mov  ax, [bx + si]
35              cmp  ax, [bx + si + 2]
36              jbe  noswap
37                call swap    ; another call here
38                ; mov  byte[swapflag], 1
39                mov  word [bp - 2], 1
40              noswap:
41              add  si, 2
42              cmp  si, cx
43              jne  innerloop
44          cmp  word [bp - 2],  1
45          je   mainloop
46      ; handle parameter stack issue at end again ------------------
47      pop si
48      pop cx
49      pop bx
50      pop ax
51      mov sp, bp   ; sp should be restored
52      pop bp       ; bp was the first thing pushed, so last popped!
53      ; stack cleared? ------------------------------------------
54      ret 4        ; what is this guy?
55  start:
56      mov  bx, data
57      mov  cx, 2
58      push bx
59      push cx
60      ; make a function call
61      call bubblesort
62      ; data is now sorted!
63      mov ax, 0x4c00
64      int 0x21
```

CODE NO 2:

In Practice by understanding the concept of stack I have implemented an recursive function. That recursive function basically find the GCD(Greatest common divisor). The subroutine(function) take 2 numbers as a parameter and find it's GCD.

a-3-q2.asm

```
1  [org 0x0100]
2  jmp start
3      X: dw 48
4      Y: dw 72
5      result: dw 0
6  GCD:
7      push bp
8      mov bp, sp
9      push ax
10     push bx
11
12     mov ax, [bp + 4]        ;Y Value
13     mov bx, [bp + 6]        ;X Value
14
15     ;if condition
16     cmp ax, 0
17     jne elif
18         mov [result], bx
19         pop bx
20         pop ax
21         pop bp
22         ret 4               ; 2 Parameters given
23     ;elif condition
24     elif:
25         cmp bx, ax
26         jae else
27             ;elif
28             push ax
29             push bx
30             call GCD
31             pop bx
32             pop ax
33             pop bp
34             ret 4
35     ;else condition
36     else:
37         sub bx, ax
38         push bx
39         push ax
40         call GCD
41         pop ax
42         pop bx
43         pop bp
44         ret 4
45
46 start:
47     push word [X]
48     push word [Y]
49     call GCD
50
51     mov ax, 0x4c00
52     int 0x21
```

Name : Jawad Ahmed
Roll No: 20P-0165
Section : 3A