1. Please implement the following steps with PUSH and POP in the .code section: (a) get values 1 and 2 into the stack; (b) save values 2 and 1 in EAX and EBX

```
.code
main proc
        mov eax, _____________ ;save value 1 in EAX
        mov ebx, _____________ ;save value 2 in EBX

        push _____________ ; get values 1 into the stack
        push _____________ ; get values 2 into the stack

        pop _____________ ;save value 2 in EAX
        pop _____________ ;save value 1 in EBX
        invoke ExitProcess,0
    main endp
end main
```

2. To (1) get the values 6, 4, and 2 into the stack; and (2) save values 2, 4 and 6 in EAX (values can be overwritten in EAX)

, please fill out blank lines in the .code section. (assume: array WORD 2,4,6)

```
.code
main proc
        mov eax,0
        mov ecx,3

        pushLoop:
            push array[_________________]  ; locate a proper index of array
        loop pushLoop

        ____________________ ; configure ECX for popLoop

        popLoop:
            pop _______________ ; save values 2, 4 and 6 in EAX
        loop popLoop

        invoke ExitProcess,0
```

```
   main endp
end main
```

3. Please predict the values in EDX in step ⓪-④. (assume: arrayVariable DWORD 3h, 6h, 9h)

```
.code
main proc
        mov eax,0
        mov ecx,3
        mov edx, arrayVariable[0]----------⓪  ; EDX =
        mov edx, arrayVariable[1]----------①  ; EDX =
        mov edx, arrayVariable[2]----------②  ; EDX =
        mov edx, arrayVariable[3]----------③  ; EDX =
        mov edx, arrayVariable[4]----------④  ; EDX =

        pushLoop:
            push arrayVariable[_____________________]; proper index for DWORD
        loop pushLoop

        mov ecx, 3
        popLoop:
             pop eax
        loop popLoop

        invoke ExitProcess,0
   main endp
end main
```

4. Reverse String. Please fill out blank lines with proper instructions.

```
.data
aName BYTE "Assembly Language",0

.code
main PROC
```

```
; Push the name on the stack.
        mov ecx, _________________  ; ecx = ?/Alternative to get a size
        mov esi, ___________ ; initialize ESI

L1:     movzx eax, __________       ; get character
        push _____          ; push on stack
        inc ______ ; update ESI
        Loop L1

; Pop the name from the stack, in reverse,
; and store in the aName array.
        mov ecx, _____________  ; configure ecx for loop L2 again
        mov esi, ________ ; configure esi for aName again

L2:     pop eax   ; get character
        mov aName[esi],______       ; store in string
        inc ______ ; update ESI
        Loop L2

; Display the name.
        ……..
    main ENDP
END main
```

5. Please use two procedures (pushProc and popProc) to rewrite Q2.

```
.code
main proc
        mov eax,0
        mov ecx,3
; Main program control procedure.
; Calls: pushProc and popProc.
        _____________   ; call pushProc procedure
        mov ecx, 3
        _____________  ; call popProc procedure
```

```
main ENDP

;-----------------------------------------------------
_________________
;
; Push values in array into stack
;-----------------------------------------------------
  pushLoop:
              push array[(ecx *2) - 2]
            loop pushLoop


            __________
_________________
;-----------------------------------------------------
_____________
;
; Pop each value one by one in EAX
;-----------------------------------------------------
            popLoop:
              pop eax
            loop popLoop


            __________
_________________
end main
```