SECR 2033 Computer Organization and Architecture
2019/2020-2

Programming 6: CONDITIONAL STRUCTURES

Part A – Programming review

A) Block-Structured IF Statements

Assembly language programmers can easily translate logical statements written in C++/Java into assembly language

```
if( op1 == op2 )
  X = 1;
else
  X = 2;
```

```
      mov eax,op1
      cmp eax,op2
      jne L1
      mov X,1
      jmp L2
L1:   mov X,2
L2:
```

B) Compound Expressions

Logical AND Operator

When implementing the logical AND and OR operator, consider that high-level languages compilers for Java, C, and C++ use short-circuit evaluation for efficiency reasons.
The second expression is not evaluated if the first expression is false. (early exit)

```
if (al > bl) AND (bl > cl)
    X = 1;
```

```
   cmp al,bl        ; first expression...
   jbe next         ; quit if false
   cmp bl,cl        ; second expression...
   jbe next         ; quit if false
   mov X,1          ; both are true
next:
```

Logical OR Operator

```
if (al > bl) OR (bl > cl)
    X = 1;
```

```
    cmp al, bl          ; is AL > BL?
    ja  L1              ; yes
    cmp bl, cl          ; no: is BL > CL?
    jbe next            ; no: skip next statement
L1:
    mov X,1             ; set X to 1
next:
```

C) WHILE Loops

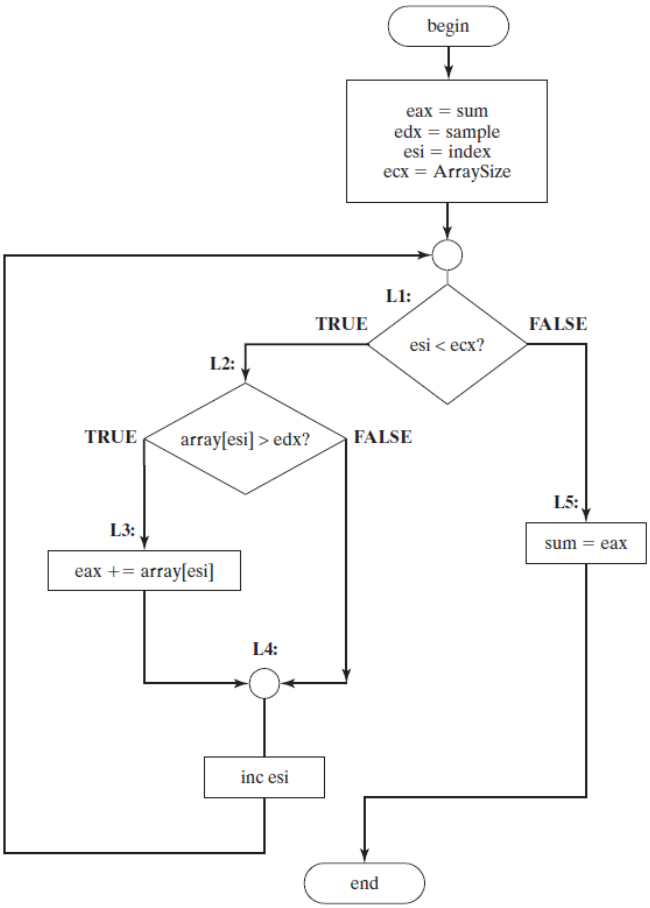
The WHILE structure tests a condition first before performing a block of statement. As long as the loop condition remains true, the statement is repeated.

```
while(eax < ebx)
  eax = eax + 1;
```

```
top:
    cmp eax,ebx     ; check loop condition
    jae next        ; false? exit loop
    inc eax         ; body of loop
    jmp top         ; repeat the loop
next:
```

D) Assembly Code

The easiest way to generate the assembly code from a flowchart is to implement the code for each shape. Note the direct correlation between the flowchart labels and the labels used in the source code.




```
.data
sum DWORD 0
sample DWORD 50
array DWORD 10,60,20,33,72,89,45,65,72,18
ArraySize = ($ - Array) / TYPE array

.code
main PROC
     mov   eax,0                ; sum
     mov   edx,sample
     mov   esi,0                ; index
     mov   ecx,ArraySize

L1:  cmp   esi,ecx              ; if esi < ecx
     jl    L2
     jmp   L5

L2:  cmp   array[esi*4], edx    ; if array[esi] > edx
     jg    L3
     jmp   L4
L3:  add   eax,array[esi*4]

L4:  inc   esi
     jmp   L1

L5:  mov   sum,eax
```

Part B – Let's do a little programming on your own

1. Implement the following pseudocode in assembly language, using unsigned values. Please use short-circuit evaluation in your code.

```
IF (MY_MONEY < YOUR_MONEY)
   YOU_DONATE = 20;
ELSE
{
   YOU_DONATE = 10;
   I_DONATE   = 10;
}
```

2. Implement the following pseudocode in assembly language, using unsigned values. Please use short-circuit evaluation in your code.

```
IF (EBX > ECX && ECX <= EDX)
{
   EAX = 9;
   EDX = 10;
}
```

3. Implement the following loop, using unsigned 32-bit integers. Please use the reverse condition to make the code shorter.

```
WHILE (VAR <= TRIAL)
{
   VAR++;
   PRINTF("Sorry your trial period has expired");
}
```

4. Rewrite the code from section (D) so it is functionally equivalent but uses fewer instructions.

5. Suppose a student wants to decide to go home or stay in UTM during online teaching & learning phase. Two criteria are used to determine whether the student will go home or not:
   - First is the student's hometown current zone, whether it is in green or red zone.
   - Second is the internet speed in the student's hometown. The speed must at least be 2MBps to have a decent teaching & learning experience.

   a. Draw a flowchart of the program.

   b. Write an assembly language program that ask the user to enter the inputs and suggest a decision to the student.