**Flow Control Instructions and Looping Structures**

## 

### **📘 Overview**

Here we explored **repetition structures** in assembly language. These loops allow a section of code to run **multiple times**, depending on conditions or counters—just like for, while, and do-while loops in high-level languages.

I also learned to:

* Control flow explicitly and implicitly
* Use registers like CX, SI, DI, AX, etc., to hold counters and accumulators
* Take **character input** and process it with loops
* Implement **nested loops** and **conditional summation**

## **🔁 Looping Concepts Learned**

### **1️⃣ Explicit Loops (WHILE-style)**

#### **🔍 Concept:**

* Uses CMP, JMP, and labels to form loops
* Manual comparison and jump conditions
* Similar to while(condition) in C

#### **💡 Example:**

MOV DL, 30h ; ASCII '0'

PrintLoop:

CMP DL, 35h ; Compare with ASCII '5'

JGE EndLoop ; Exit loop if DL >= '5'

MOV AH, 02h

INT 21h ; Print current digit

INC DL

JMP PrintLoop ; Repeat

EndLoop:

### **2️⃣ Implicit Loops (FOR-style using LOOP)**

#### **🔍 Concept:**

* Uses CX as a counter and LOOP instruction
* Automatically decrements CX and checks if it’s 0
* Like for(i=0; i<n; i++)

#### **💡 Example:**

MOV CX, 5

MOV DL, 30h

LoopStart:

MOV AH, 02h

INT 21h ; Print DL

INC DL

LOOP LoopStart ; CX--; jump if CX ≠ 0

### **3️⃣ Repeat-Until Loops (DO-WHILE style)**

#### **🔍 Concept:**

* Executes at least once
* Uses CMP + JNE or JE to loop until condition is met

#### **💡 Example:**

RepeatInput:

MOV AH, 01h

INT 21h ; Read char into AL

MOV DL, AL

MOV AH, 02h

INT 21h ; Echo char

CMP AL, 20h ; Check if SPACE (ASCII 32)

JNE RepeatInput ; Repeat until SPACE pressed

### **4️⃣ Nested Loops**

#### **🔍 Concept:**

* One loop inside another
* Used for printing patterns or processing matrices
* Inner loop resets every time the outer loop runs

#### **💡 Example: Print triangle of numbers**

MOV CX, 5 ; Outer loop = number of rows

OuterLoop:

PUSH CX

MOV BL, 1 ; Start number

MOV CL, 5

SUB CL, CH ; Print decreasing number of elements

InnerLoop:

CMP BL, CL

JG EndInner

MOV DL, BL

ADD DL, 30h

MOV AH, 02h

INT 21h

INC BL

JMP InnerLoop

EndInner:

POP CX

LEA DX, newline

MOV AH, 09h

INT 21h

LOOP OuterLoop

### **5️⃣ Character Input with Looping**

#### **🔍 Concept:**

* User types characters one-by-one
* Loop continues until a certain condition is met (e.g., SPACE pressed)

#### **💡 Example:**

MOV AH, 01h

INT 21h ; Read char

CMP AL, 20h ; SPACE = 32

JNE RepeatInput

### **6️⃣ Summation Using Conditional Logic Inside Loops**

#### **🔍 Concept:**

* Loop through numbers and add to total if a condition is met
* Separate accumulators used (e.g., for divisible and non-divisible values)

#### **💡 Example: Sum of 1–100 divisible by 5**

MOV CX, 100

MOV SI, 0 ; Divisible sum

MOV BX, 5

SumLoop:

MOV AX, 101

SUB AX, CX ; Get current number

MOV DX, 0

DIV BX ; Divide by 5

CMP DX, 0

JE AddToSI

JMP SkipAdd

AddToSI:

ADD SI, AX

SkipAdd:

LOOP SumLoop

## **🧠 Registers Used**

| **Register** | **Purpose** |
| --- | --- |
| AX, BX | General purpose |
| CX | Loop counter for LOOP |
| DX | Used with output, division |
| SI, DI | Used as accumulators or for summation |

## **✅ Final Takeaways**

| **What I Can Now Do** | **How** |
| --- | --- |
| Repeat code | Using LOOP, JMP, and conditions |
| Take user input in a loop | INT 21h, AH=01h with CMP |
| Perform pattern printing | Using nested loops |
| Do math with conditions | DIV, CMP, conditional jumps |
| Simulate for, while, do-while | With labels, loops, flags, and registers |