**PRACTICAL # 06**

**OBJECT:**

Loops in assembly.

**THEORY:**

Loops in assembly are formed using the conditionals and jumps statements. These are the very same jumps that are used for if/else branching. Until the condition is true, the loop iterations continue. When the condition results false, we break out of the loop.

**Program:**

The program implements a loop to print digits from 0 to 9. Finally outputs the result.

*.model small*

*.stack 100h*

*.data*

*.code*

*main proc*

*mov cl, 48 ; initialize the counter with ASCII value of 0*

*mov ah, 2 ; character output interrupt number*

*next:*

*mov dl, cl ; print current digit*

*int 21h*

*inc cl ; increment counter*

*cmp cl, 57; ascii value for '9'*

*jle next ; loop until counter is less than or equal to 9*

*main endp*

*end main*

The program initializes a counter variable with value ‘0’ (remember ASCII value of ‘0’ is 48). It then prints the digit in loop body and increments the counter by 1. Next statement compares the counter with ‘9’ and if it is not greater than ‘9’, the jump statement transfers control back to next label for the next iteration. When the condition is not satisfied, the loop breaks.

Note that there is a little optimization used in program.

There are dedicated opcodes for the loops, like loopz, loopnz. The difference is they manipulate the counter register CX automatically.

|  |  |
| --- | --- |
| **Instruction** | **operation and jump condition** |
| LOOP | decrease cx, jump to label if cx not zero. |
| LOOPE | decrease cx, jump to label if cx not zero and equal (zf = 1). |
| LOOPNE | decrease cx, jump to label if cx not zero and not equal (zf = 0). |
| LOOPNZ | decrease cx, jump to label if cx not zero and zf = 0. |
| LOOPZ | decrease cx, jump to label if cx not zero and zf = 1. |
| JCXZ | jump to label if cx is zero. |

The program below prints numbers 0 to 9 using loop.

*.model small*

*.stack 100h*

*.data*

*.code*

*main proc*

*mov cx, 9 ; initialize loop counter*

*mov dl, 48 ; initialize the counter with ASCII value of 0*

*mov ah, 2 ; character output interrupt number*

*next:*

*int 21h*

*loop next ;* *loop until counter is equal to 0*

*main endp*

*end main*

Here is another program calculating sum of numbers from 0-3.

*main proc*

*mov cx, 3 ; initialize loop counter*

*mov ax, 0 ; initialize the counter with ASCII value of 0*

*next:*

*add ax, cx*

*loop next ; loop until counter is equal to 0*

*main endp*

*end main*

Although these can be used to implement the loop, you can use whatever way you like. Furthermore, these dedicated loop opcodes may be bit slow performance-wise, so you may choose to implement loops with the conditional jumps.

**ACTIVITIES**

**Activity 1**

Write a similar program to print digits in reverse order that is from 9 to 0.

**Activity 2**

Write a similar program to that outputs odd numbers from 1 to 9.

**Activity 3**

Write a program to output capital alphabets from ‘A’ to ‘Z’.

**Activity 4**

Rewrite the above program using the dedicated loop opcodes like **loop** , **loopnz**;

**REVIEW QUESTIONS**

1. How is a loop different from if conditional?
2. What will happen if the counter value is not updated in loop body?
3. What will be the result if the loop counter/index is initialized within the loop body?

1. What is the difference between a loop with conditional jump and a loop using opcodes like **loopz** , **loopnz**;