Assembly - Loops

The JMP instruction can be used for implementing loops. For example, the following code snippet can be used for executing the loop-body 10 times.

```
MOV     CL, 10
L1:
<LOOP-BODY>
DEC     CL
JNZ     L1
```

The processor instruction set, however, includes a group of loop instructions for implementing iteration. The basic LOOP instruction has the following syntax −

```
LOOP      label
```

Where, label is the target label that identifies the target instruction as in the jump instructions. The LOOP instruction assumes that the **ECX register contains the loop count**. When the loop instruction is executed, the ECX register is decremented and the control jumps to the target label, until the ECX register value, i.e., the counter reaches the value zero.

The above code snippet could be written as −

```
mov ECX,10
l1:
<loop body>
loop l1
```

Example

The following program prints the number 1 to 9 on the screen −

Live Demo

```
section .text
   global _start        ;must be declared for using gcc

_start:                 ;tell linker entry point
   mov ecx,10
   mov eax, '1'
```

```
l1:
   mov [num], eax
   mov eax, 4
   mov ebx, 1
   push ecx

   mov ecx, num
   mov edx, 1
   int 0x80

   mov eax, [num]
   sub eax, '0'
   inc eax
   add eax, '0'
   pop ecx
   loop l1

   mov eax,1             ;system call number (sys_exit)
   int 0x80              ;call kernel
section .bss
num resb 1
```

When the above code is compiled and executed, it produces the following result –

```
123456789:
```