C PROGRAMMING FOR ENGINEERS

Control Structures
(while loop, do/while loop, for loop)

Types of Structured Programming

- Sequence: set of steps to be performed in order.
- Selection: set of steps to be performed if a logical condition is true; another set of steps if the logical condition is false.
- Repetition: set of steps to be repeated if a condition is true.

Loop Structure Types

- Repetition structures (loops) allow statements that are placed inside the structure to be repeated as long as the testing condition is true.
- There are three categories:
  - while loop
  - do/while loop
  - for loop

NOTE: The while loop incorporates the concept of decision before logic. The do/while and for loops incorporate the concept of decision after logic.

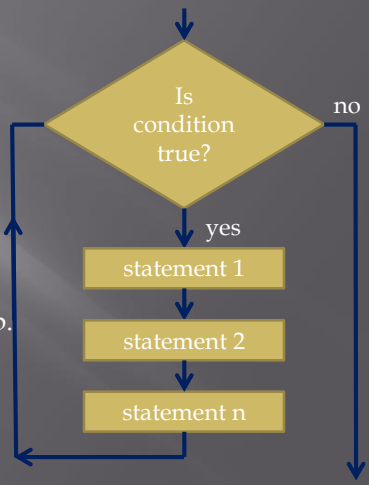
while Loop

- General form:

```
while(condition)
{
    statements;
}
```

If the condition is false, the program will skip the loop statements and exit the loop.

If the condition is true, the program will execute the loop statements and repeat them until the condition turns false.



while Loop Example

Since time is the loop control variable in the while loop shown below, time needs a starting value.

```
time = 0.0;
while(time <= 25.0)
{
   acceleration = 4.25–0.015*pow(time,2);
   velocity = 4.25*time-0.005*pow(time,3);
   distance = 90.0+2.125*pow(time,2)-0.00125*pow(time,4);
   printf("%4.1f    %7.2f    %7.2f    %8.2f\n",time,
          acceleration, velocity, distance);
   time = time + 1.0;
}
```

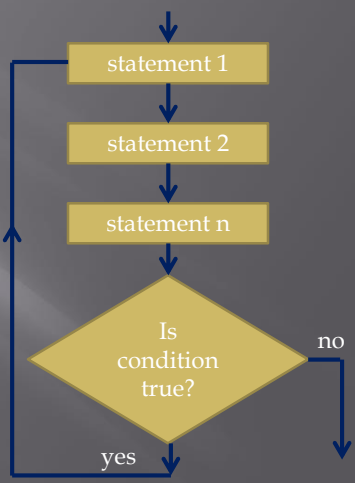
Loop controlling condition.

Time needs to be incremented for the loop to continue.

do/while Loop

- General Form:

```
do
{
   statements;
} while(condition);
```



Notice that the loop control condition is evaluated at the end of the loop. This will guarantee that the loop statements will be executed at least once.

do/while Loop Example

```
time = 0.0;
do
{
    acceleration = 4.25–0.015*pow(time,2);
    velocity = 4.25*time-0.005*pow(time,3);
    distance = 90.0+2.125*pow(time,2)-0.00125*pow(time,4);
    printf("%4.1f     %7.2f     %7.2f     %8.2f\n",time,
            acceleration, velocity, distance);
    time = time + 1.0;
}while(time <= 25.0);
```

Time needs a starting value.

Time needs to be incremented for the loop to continue.

Loop controlling condition.

for Loop

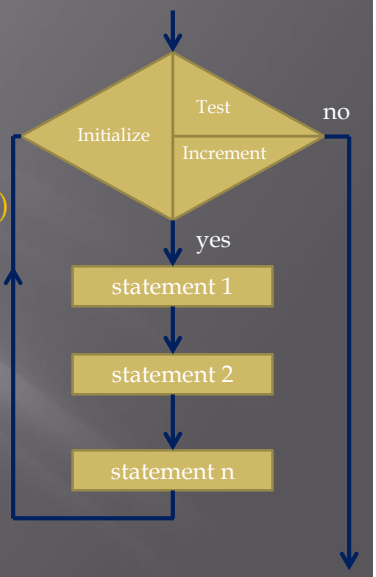
- General form:

```
for(expression1; expression2; expression3)
{
    statements;
}
```

Initialize the loop control variable.

Test condition.

Increment or decrement the loop control variable.



for Loop Example

```
pi = 3.14159;
for(angle = 0.0; angle <= 180.0; angle += 5.0)
{
    radians = angle*pi/180.0;
    sine = sin(radians);
    cosine = cos(radians);
    tangent = tan(radians);
}
```

Initialize the loop control variable.

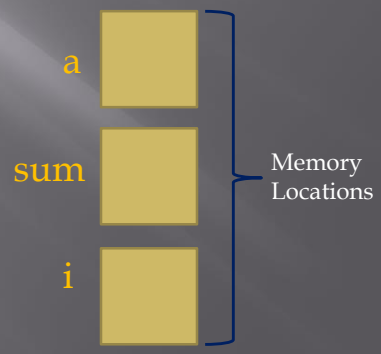
Increment or decrement the loop control variable.

Test condition.

Notice that semicolons must be used in the for statement, not commas. This is a common mistake for beginning programmers.

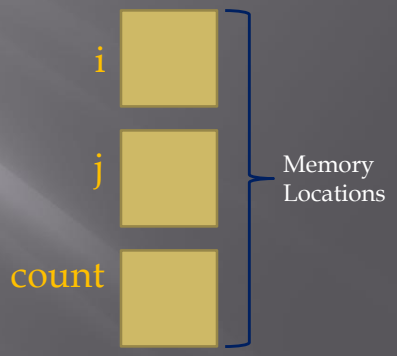
Another for Loop Example

```
int sum = 0, a = 2, i;
for(i = 0; i <= 3; i++)
{
    a = a + 2;
    sum = sum + a;
}
```



Nested for Loops Example

```
int i, j, count=0;
for(i = -1; i <= 1; i++)
{
    for(j = 0; j <= 2; j++)
    {
        count ++;
    }
}
```



Calculating for Loop Iterations

- The number of iterations that a for loop will be executed is determined by:

$$floor\left[\frac{final\ value - initial\ value}{increment}\right] + 1$$

The floor command rounds to the nearest integer toward negative infinity.

Example: floor(180.0 – 0.0)/5.0 + 1 = 37

Infinite Loops

Infinite loops occur when the loop condition has not been met. An example would be when the condition in a while loop is always true.

Computer systems will have several methods to stop infinite loops:

1. System-defined limit exceeded (generally time defined on main-frame computers).
2. Ctrl C (stops or aborts program execution on PCs).

The break & continue Statements

- break;
  - Used to exit the inner most loop or operation.
- continue;
  - Skips the remaining statements in a current iteration and then continues with the next iteration of the loop or operation.

break Example

Example:

```
int i, sum = 0, y;
for(i = 1; i <= 5; i++)
{
    scanf("%i", &y);
    if(y > 10)
        break;
    sum += y;
}
```

What is the value of sum after the following values are entered via the scanf command?

2
5
11
20
1

Program will break out of the loop when y > 10.

continue Example

Example:

```
int i, sum = 0, y;
for(i = 1; i <= 5; i++)
{
    scanf("%i", &y);
    if(y > 10)
        continue;
    sum += y;
}
```

What is the value of sum after the following values are entered via the scanf command?

2
5
11
20
1

When y > 10 the program will skip the rest of the statements in the current iteration and continue to the next iteration.