Increment and decrement operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Called</th>
<th>Sample expression</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>++</td>
<td>preincrement</td>
<td>++a</td>
<td>Increment a by 1, then use the new value of a in the expression in which a resides.</td>
</tr>
<tr>
<td>++</td>
<td>postincrement</td>
<td>a++</td>
<td>Use the current value of a in the expression in which a resides, then increment a by 1.</td>
</tr>
<tr>
<td>--</td>
<td>predecrement</td>
<td>--b</td>
<td>Decrement b by 1, then use the new value of b in the expression in which b resides.</td>
</tr>
<tr>
<td>--</td>
<td>postdecrement</td>
<td>b--</td>
<td>Use the current value of b in the expression in which b resides, then decrement b by 1.</td>
</tr>
</tbody>
</table>

Arithmetic assignment operators

<table>
<thead>
<tr>
<th>Assignment operator</th>
<th>Sample expression</th>
<th>Explanation</th>
<th>Assigns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assume: int c = 3, d = 5, e = 4, f = 6, g = 12;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+=</td>
<td>c += 7</td>
<td>c = c + 7</td>
<td>10 to c</td>
</tr>
<tr>
<td>-=</td>
<td>d -= 4</td>
<td>d = d - 4</td>
<td>1 to d</td>
</tr>
<tr>
<td>*=</td>
<td>e *= 5</td>
<td>e = e * 5</td>
<td>20 to e</td>
</tr>
<tr>
<td>/=</td>
<td>f /= 3</td>
<td>f = f / 3</td>
<td>2 to f</td>
</tr>
<tr>
<td>%=</td>
<td>g %= 9</td>
<td>g = g % 9</td>
<td>3 to g</td>
</tr>
</tbody>
</table>
```c
#include "stdio.h"
#include "conio.h"
#include "math.h"

int main()
{
    int i=1;
    long int N;
    printf("give number ");
    scanf("%ld", &N);
    while (pow(10, i) <= N)
    {
        i = i + 1;
        printf("%ld has %d decimals", N, i);
        getch();
    }
    return 0;
}
```
The while statement can be used to implement any counter-controlled loop.

C++ also provides the **for** repetition statement, which specifies the counter-controlled repetition details in a single line of code.
When the program reach the for loop, first executes 1, and then checks 2 whether true or not. If 2 is true then executes 4. After 4 is done, 3 will be executed. So one loop is completed. If 2 is still true, program executes, in order, 4,3. Therefore 4 will be executed until 2 is false.

4 - 3 - 2; 4 - 3 - 2; 4 - 3 - 2...
for keyword
Control variable name
Required semicolon separator
Final value of control variable for which the condition is true
Required semicolon separator

for (int counter = 1; counter <= 10; counter++)

Initial value of control variable
Loop-continuation condition
Increment of control variable

- Initialize control variable
  - int counter = 1

- Determine whether looping should continue
  - [counter <= 10]
    - Display the counter value
      - Increment the control variable
        - counter++
Let's write program of factorial using for loop

```
1 #include"stdio.h"
2 #include"conio.h"
3 #include"math.h"
4 int main()
5 {
6     float f=1;
7     printf("enter number ");
8     scanf("%d",&N);
9     for (i=2 ; i<=N ; i++) {
10         f*=i;  // f=f*i;
11     }
12     printf("%d!=%.0f",N,f);
13     getch();
14     return 0;
15 }
```

Write a program to find the maximum term in first 100 terms of the following sequence

\[ \{-n^2 + 10n - 21\}_{n=0}^{\infty} \]

```
1 #include"stdio.h"
2 #include"conio.h"
3
4 int main() {
5     int max=-21;
6     int Aen;
7     for (int n=1;n<100;n++) {
8         Aen=-n*n+10*n-21;
9         if (Aen>max) max=Aen;
10     }
11     printf("Max term is %d",max);
12     getch();
13     return 0;
14 }
```