Loops - while

- while statement
  
  while (condition) {
    statement_{1};
    
    ... 
    
    statement_{n};
  }

- How it works
  1. Check condition
  2. If true (1) execute statements and go back to 1
  3. If false (0) exit out of while

Need to have a statement that changes the condition so that we are not in an infinite loop

Example

```
/* Calculates the factorial of the number entered */
#include <stdio.h>

int main(void)
{
    int Number; /* The number user enters */
    int Factorial = 1; /* The factorial of the number entered */
    int LpIndx = 1; /* The loop index */

    /* Print instruction to enter a integer */
    printf("This program finds the factorial of an integer that is entered\n");
    printf("Please enter an integer: ");

    /* Read the number */
    scanf("%d", &Number);

    /* Calculate the factorial */
    while (LpIndx <= Number) {
        Factorial = Factorial * LpIndx;
        LpIndx = LpIndx + 1;
    }

    /* Print the output */
    printf("The factorial of %d is %d\n", Number, Factorial);
    return(0);
}
```

Number = 3

<table>
<thead>
<tr>
<th>Loop</th>
<th>Factorial</th>
<th>LpIndx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Init</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>4</td>
</tr>
</tbody>
</table>

Number = 0

<table>
<thead>
<tr>
<th>Loop</th>
<th>Factorial</th>
<th>LpIndx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Init</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Loops – do-while

- do-while statement
  
  do {
    statement_1;
    
    ;
    statement_n;
  } while (condition);

- How it works
  1. Execute Statements
  2. Check condition
  3. If true (1) go back to 1
  4. If false (0) exit out of while

Need to have a statement that changes the condition so that we are not in an infinite loop

---

Example

/* Calculates the factorial of the number entered */
#include <stdio.h>

int main(void)
{
  int Number; /* The number user enters */
  int Factorial = 1; /* The factorial of the number entered */
  int LpIndx = 1; /* The loop index */

  /* Print instruction to enter a integer */
  printf("This program finds the factorial of an integer that is entered\n");
  printf("Please enter an integer: ");

  /* Read the number */
  scanf("%d", &Number);

  /* Calculate the factorial */
  do {
    Factorial = Factorial * LpIndx;
    LpIndx = LpIndx + 1;
  } while (LpIndx <= Number);

  /* Print the output */
  printf("The factorial of %d is %d\n", Number, Factorial);
  return(0);
}
Example 2

What does the following loop do?

```c
char ans = 'n';
while (ans != 'y') {
    printf("Would you like to quit [y/n]: ");
    scanf("%c", &ans);
}
```

Compound Assignment

- When a variable appears on both sides of an assignment, shorthand assignment can be used:
  
  ```c
  LpIndx = LpIndx + 1
  
  can be written as
  
  LpIndx += 1
  ```

- Same shorthand can be applied to -, *, /, and %
  
  ```c
  x /= n + 2
  
  means
  
  x = x / (n + 2)
  ```
Increment/Decrement Operators

- **Increment operator** `++`
  - Used before a variable
    - `++k`
    - Means variable is incremented by 1, then used
  - Used after a variable
    - `k++`
    - Means variable is used, then incremented by 1

<table>
<thead>
<tr>
<th>Before</th>
<th>x</th>
<th>k</th>
<th>After</th>
<th>x</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>k = 1; x = 0;</td>
<td>0</td>
<td>1</td>
<td>k = 1; x = 0;</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>x = 2 * ++k;</td>
<td>4</td>
<td>2</td>
<td>x = 2 * k++;</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

- **Decrement operator** `--`
  - Used before a variable
    - `--k`
    - Means variable is decremented by 1, then used
  - Used after a variable
    - `k--`
    - Means variable is used, then decremented by 1

<table>
<thead>
<tr>
<th>Before</th>
<th>x</th>
<th>k</th>
<th>After</th>
<th>x</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>k = 1; x = 0;</td>
<td>0</td>
<td>1</td>
<td>k = 1; x = 0;</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>x = 2 * --k;</td>
<td>0</td>
<td>0</td>
<td>x = 2 * k--;</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>
Loops - for

for statement

for (init cond; loop cond; upd exp) {
    statement_1;
    ...
    statement_n;
}

How it works
1. Initial condition is set
2. Loop condition is checked
3. If false (0) exit out of for
4. If true (1) statements are executed
5. update expression is calculated
6. Loop back to 2.

Example
/* Calculates the factorial of the number entered */
#include <stdio.h>
int main(void) {
    int Number; /* The number user enters */
    int Factorial = 1; /* The factorial of the number entered */
    int LpIndx = 1; /* The loop index */
    /* Print instruction to enter a integer */
    printf("This program finds the factorial of an integer that is entered\n");
    printf("Please enter an integer: ");
    /* Read the number */
    scanf("%d", &Number);
    /* Calculate the factorial */
    for (LpIndx = 1; LpIndx <= Number; ++LpIndx) {
        Factorial = Factorial * LpIndx;
    }
    /* Print the output */
    printf("The factorial of %d is %d\n", Number, Factorial);
    return(0);
}