While Loops

2024 Winter APS 105: Computer Fundamentals

Jon Eyolfson

Lecture 8

1.0.0

While Loops Introduce Repetition

If statements allow execution to jump forward

While statements allow execution to jump backward (and repeat)

We Can Repeat Code When the Expression is True

The syntax of a while statement is:

```
while (<expr>) <stmt>
```

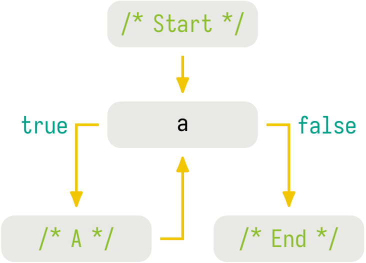
However, you should **always** write it like:

```
while (<expr>) {
    <stmts>
}
```

The Flow of a While Loop

```
/* Start */
while (a) {
    /* A */
}
/* End */
```

becomes



Write a Program to Countdown from 10 then Blast Off!

We have no input, our output should be:

```
Countdown: 10
Countdown: 09
Countdown: 08
Countdown: 07
Countdown: 06
Countdown: 05
Countdown: 04
Countdown: 03
Countdown: 02
Countdown: 01
Blast off!
```

Previous Solution

```c
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    int count = 10;
    while (count > 0) {
        printf("Countdown: %02d\n", count);
        --count;
    }
    printf("Blast off!\n");
    return EXIT_SUCCESS;
}
```

Beware of Infinite Loops!

The statements that run in the body of the loop need to eventually make the expression `false`

Otherwise, your program will not stop!

If this happens press `Crtl+C` in your terminal to stop it

Write a Program to Print the Number of Digits in an Integer

Input: a positive integer (> 0)

Examples:

```
1 → Number of digits: 1
10 → Number of digits: 2
123 → Number of digits: 3
54321 → Number of digits: 5
```

Previous Solution

```c
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    int input = 0;
    printf("Input a positive integer: ");
    scanf("%d", &input);
    int num_digits = 0;
    while (input > 0) {
        input /= 10;
        ++num_digits;
    }
    printf("Number of digits: %d\n", num_digits);
    return EXIT_SUCCESS;
}
```

We Can Run Code and Repeat It When the Expression is True

The syntax of a do while statement is:

```
do <stmt> while (expr);
```

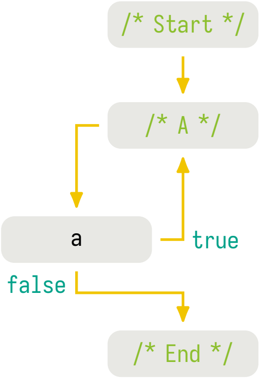
However, you should **always** write it like:

```
do {
    <stmts>
}
while (<expr>);
```

The Flow of a Do While Loop

```
/* Start */
do {
    /* A */
} while (a);
/* End */
```

becomes



Write a Program to Ensure an Integer is Positive

Input: any valid integer (that fits in an `int`)

Output: a positive integer (> 0)

`do while` loops are useful when you always want the statements to run at least once

Previous Solution

```c
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    int input = 0;
    do {
        printf("Input a positive integer: ");
        scanf("%d", &input);
    } while (input <= 0);
    printf("You entered: %d\n", input);
    return EXIT_SUCCESS;
}
```

Write a Program to Sum Positive Integers

Input: any number of positive integers (> 0)
an input of 0 means the end of input

Output: a single integer representing the sum of all inputs

Previous Solution

```c
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    int sum = 0;
    printf("Input a positive number: ");
    int input = 0;
    scanf("%d", &input);
    while (input != 0) {
        sum += input;
        printf("Input a positive integer (0 to stop): ");
        scanf("%d", &input);
    }
    printf("The sum is: %d\n", sum);
    return EXIT_SUCCESS;
}
```