ESc 101: Fundamentals of Computing

Lecture 9

Jan 20, 2010

++ AND --

- ++ and -- are unary operators.
- They increment and decrement the value of variable, unlike other operators.
- In addition, the operations also have a result, exactly like the other operators.
- The result of `n++` is the initial value of `n`.
- The result of `++n` is the final value `n`.
- Similarly for `--n` and `n--`.

++ AND --

Consider following program fragment:

```
n = 5;
m = ((n++) - 7) + n;
```

- The value of `n` after the execution is 6.
- The value of `m` after execution is 3.

Caution: do not use ++ or -- inside expressions!

Boolean Expressions

Use the following operators:
`&&`, `||`, `!`, `<=`, `>=`, `==`, `!=`

The precedence is ensured by `(` and `)`.

Boolean Expressions

Examples:

- `((n <= 5) && (n >= 2)) || ((n >= 10) && (n != 20))`
- `!((n <= 5) && (n >= 2))`
- `&&`: AND, `||`: OR, `!`: NOT

Arithmetic with Large Numbers

- C does not provide a way of working with large numbers.
- So one needs to develop programs to do this.
- Let us define the problem first.

The Problem Statement

Write programs that read two large integers, and output their

- addition,
- subtraction,
- multiplication, and
- division

respectively.

Storing Large Numbers

Large numbers are stored using arrays:

```
int number[100];
```

Reserves 100 memory locations, each storing an integer

Arrays

- The memory locations are named `number[0]`, `number[1]`, ..., `number[99]`.
- Allows for ease of access: `number[i]` can be used where `i` is a variable storing value between 0 and 99.
- Caution: Error occurs when `number[i]` is referred with value of `i` bigger than 99!