Lecture 03

Operators and Expressions

Objectives

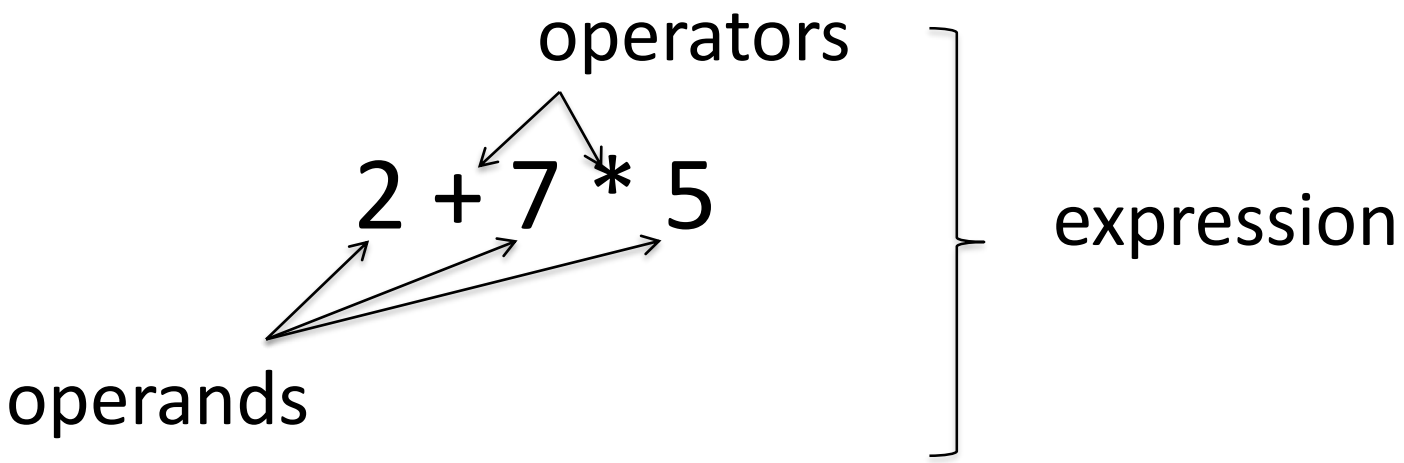
- Assignment operator
- Expression definition
- Arithmetic operators
- Relational operators
- Logical operators
- Binary operators

Assignment operator

- Assignment operator (=) is used to assign a value to a variable
- Usage LHS = RHS;
  - LHS: is a variable used to store the value
  - RHS: is a value or an expression to be evaluated as a value before assigning to the variable

Expression

- An expression is a combination between operands and operators
- An expression can be evaluated as a value



Operators

- Besides assignment operator, there are more important operators
  - Arithmetic operators: +, -, *, /, %
  - Unary operators: ++, --
  - Logical operators: &&, ||, !
  - Relational operators: >, >=, ==, <, <=, !=
  - Bitwise (binary) operators: &, |

Arithmetic operators

- Used to make arithmetic expressions
  - Work with numeric operands
- Several operators are
  - Add: +
  - Minus: -
  - Multiply: *
  - Division: /
  - Modulus: %
- Modulus example: 10%3 = 1;

Activity

- Create a C Program to test the modulus operator. E.g., display the result of
  - 1%3
  - 2%3
  - 3%3
  - 4%3

Unary operators: ++, --

- Plus plus operator (++)/minus minus operator (--) is used to increase/decrease the operand by 1
  - E.g., x++ means x = x + 1;
  - E.g., x-- means x = x – 1;
- The place of these operators matters

```c
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char** argv) {
    int x = 1;
    //print first then increase
    printf("%d\n", x++);
    printf("%d\n", x);
    return (EXIT_SUCCESS);
}
```

```
1
2
```

```c
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char** argv) {
    int x = 1;
    //increase first then print
    printf("%d\n", ++x);
    printf("%d\n", x);
    return (EXIT_SUCCESS);
}
```

```
2
2
```

Logical operators

- Used to make logical expressions
  - Work with logical operands (true, false)
- Several operators are
  - AND operator: &&
  - OR operator: ||
  - NOT operator: !
- TRUTH table

X	Y	X && Y	X || Y	!Y
true	true	true	true	false
true	false	false	true	true
false	true	false	true	
false	false	false	false	

Note that in C programming, *false* is represented as 0 and *true* is everything else

Activity

- Create a C program to show truth tables
  - Make use of logical operators
- E.g., will show the results of
  - 1 && 1
  - 1 && 0
  - 0 && 1
  - 0 && 0
  - So on and so forth (for other logical operators)

Relational/Comparison operators

- Used to make comparison expression
  - Will produce logical value (true/false)
- Several operators are

Operator	Meaning	Example
>	Greater than	12 > 11 => true
>=	Greater than or equal	9 >= 10 => false
==	Equal	10 == 9 => false
<=	Less than or equal	10 <= 10 => true
<	Less than	9 < 10 => true
!=	Different from	9 != 10 => true

Activity

- Create a C program to display the result of following expressions
  - 10 > 11
  - 10 >= 11
  - 10 == 11
  - 10 <= 11
  - 10 < 11
  - 10 != 11

Binary/bitwise operators (Advanced)

- It operates on the operands as binary format
  - It works on a bit at a time

AND (NUM1 & NUM2)	Return 1 if both the operands are 1
OR (NUM1 | NUM2)	Returns 1 if bits of either of the operand are 1
NOT (~ NUM1)	Reverses the bits of its operand (from 0 to 1 and 1 to 0)
XOR (NUM1 ^ NUM2)	Returns 1 if either of the bits in an operand is 1 but not both

Activity

Make a program to test these

10 & 15→ 1010 & 1111→1010 → 10

10 | 15→ 1010 | 1111→1111 → 15

10 ^ 15→ 1010 ^ 1111→0101 → 5

~ 10 → ~1010 →1011 → -11

Summaries

- Assignment operator
- Expression definition
- Arithmetic operators
- Relational operators
- Logical operators
- Binary operators
- Operator precedence