Introduction to Programming
C and Julia
Lecture 05: Operators- Part I

Dr Ram Prasad Krishnamoorthy

Associate Professor
School of Computing and Data Science

ram.krish@saiuniversity.edu.in

Operators

sizeof()

Operators → sizeof()

sizeof( )

An unary operator used to find the storage size of a data type.

```c
#include <stdio.h>

int main(void)
{
  char alpha = 'a';
  int value1 = 10;
  float value2 = 22.5;

  /* Apply sizeof() operator to get storage size */

  printf("sizeof(char): %ld bytes\n", sizeof(char));
  printf("sizeof(alpha): %ld bytes\n", sizeof(alpha));
  printf("\n");
  printf("sizeof(int): %ld bytes\n", sizeof(int));
  printf("sizeof(value1): %ld bytes\n", sizeof(value1));
  printf("\n");
  printf("sizeof(float): %ld bytes\n", sizeof(float));
  printf("sizeof(value2): %ld bytes\n", sizeof(value2));
  printf("\n");

  return 0;
}
```

```
sizeof(char): 1 bytes
sizeof(alpha): 1 bytes

sizeof(int): 4 bytes
sizeof(value1): 4 bytes

sizeof(float): 4 bytes
sizeof(value2): 4 bytes
```

Using `sizeof` operator without `()`

Operators → `sizeof()`

sizeof( )

Using `sizeof` operator without `()`

```c
#include <stdio.h>

int main(void)
{
  int x = 12;
  double d = 12.5;

  /* sizeof() */
  /* It is mandatory to enclose the datatypes in parentheses */
  printf("sizeof(int): %ld bytes\n", sizeof (int));

  /* The variable name can be used without parentheses */
  printf("sizeof x : %ld bytes\n", sizeof x);


  printf("sizeof(double): %ld bytes\n", sizeof (double));
  printf("sizeof d: %ld bytes\n", sizeof d);

  return 0;
}
```

Statement - Expression - Operator

Building Blocks

Statements; Expressions; Operator

Statement is a complete instruction which carry out a task.

- A group of two or more statements is called ***block*** or ***compound statement***.
- They can be grouped under { }.

Expression is anything that evaluates to a numeric value.

Operator is a symbol which instructs to perform some operation on one or more operands

Operands are something which operator acts on.

Types of operators:

- Assignment operator
- Mathematical operator
- Relational operator
- Logical operator

Assignment Operator

Assignment Operator

Assignment Operator

```
variable = expression;
```

Example:

```
x = 2;

y = x = a + 10; // Value of the expression a + 10 is assigned to both x and y

x = 6 + (y = 4 + 5); // y will be stored 9 and x will contain 15
```

Note:

```
Do not confuse = with == which is equality checking.

$20 million bug at Sun Microsystems
    x == 2;
```

Note:

We can represents comments in two ways

- /* Can also be used for

  multi-line comments */
- // Single line comments

Mathematical Operator
Unary

Mathematical Operator

Mathematical Operator - Unary

Unary operators acts on single operand which should be variables.

Operator	*Symbol*	*Action*	*Examples*
Increment	++	Increments the operand by one	++x, x++
Decrement	--	Decrements the operand by one	--x, x--

++x or --x (**prefix** mode): increments/decrements the operand **before** it is used.

x++ or x-- (**postfix** mode): increments/decrements the operand **after** it is used.

Mathematical Operator

Mathematical Operator - Unary

```c
#include <stdio.h>

int main(void)
{
  int x = 10;
  int y = 20;

  int a = 0;
  int b = 0;

  printf("x = %d, a = %d \n", x, a);

  /* First the value is assigned to a, then x gets incremented */
  a = x++;

  printf("Postfix: x = %d, a = %d \n", x, a);

  printf("y = %d, b = %d \n", y, b);

  /* First, y gets incremented, then y gets assigned to b.*/
  b = ++y;

  printf("Prefix: y = %d, b = %d \n", y, b);

  return 0;
}
```

```
x = 10, a = 0
Postfix: x = 11, a = 10
```

```
y = 20, b = 0
Prefix: y = 21, b = 21
```

Mathematical Operator
Binary

Mathematical Operator

Mathematical Operator - Binary

Binary operators acts on **two** operand.

Operator	*Symbol*	*Action*	*Example*
Addition	+	Adds two operands	x + y
Subtraction	-	Subtracts the second operand from the first operand	x - y
Multiplication	*	Multiplies two operands	x * y
Division	/	Divides the first operand by the second operand	x / y
Modulus	%	Gives the remainder when the first operand is divided by the second operand	x % y

Mathematical Operator

Exercise (live demo):

- Read two numbers from user and store it in integer variables x and y.
- Store both unary and binary operation results in a variable.
- Print the results of various operations.

Operator Precedence

Mathematical Operator

Operator precedences and parentheses

When an expression contains more than one operator, in which order the operations are performed?

Operators	*Relative Precedence*
++ --	1
* / %	2
+ -	3

Operations are performed in the following order:

- Unary increment and decrement
- Multiplication, division and modulus
- Addition and Subtraction

Note:

For operators with same precedence

- Performed left-to-right order in expression.

Parentheses can be used to modify the evaluation order.

Mathematical Operator

Operator precedences and parentheses

```
x = 4 + 5 * 3; // 19

x = (4 + 5) * 3; // 27

y = 12 % 5 * 2; // 4

x = 25 - (2 * (10 + (8 / 2))); // -3
```

Relational Operator

Relational Operators

Relational operators

Compares two expressions and evaluates to either true (1) or false (0).

Operator	*Symbol*	*Question Asked*	*Example*
Equal	`==`	Is operand 1 equal to operand 2?	`x == y`
Greater than	`>`	Is operand 1 greater than operand 2?	`x > y`
Less than	`<`	Is operand 1 less than operand 2?	`x < y`
Greater than or equal to	`>=`	Is operand 1 greater than or equal to operand 2?	`x >= y`
Less than or equal to	`<=`	Is operand 1 less than or equal to operand 2?	`x <= y`
Not equal	`!=`	Is operand 1 not equal to operand 2?	`x != y`

Conditional Operators

Conditional Operators

if - else clause

Three different forms of **if-else** clause

Form 1:

```
if(expression)
{
    statement;
}
```

Form 2:

```
if(expression)
{
    statement1;
}
else
{
    statement2;
}
```

Form 3:

```
if(expression1)
{
    statement1;
}
else if(expression2)
{
    statement2;
}
else
{
    statement3;
}
```

Conditional Operators

Exercise (live demo):

- Read three numbers from user and store it in integer variables x, y, z.
- Print the results of various relational operations.