Introduction to Programming

Prepared by: Lec Tasmiah Tamzid Anannya, CS Dept, AIUB

Computer Program

- **A computer program** is a collection of **instructions/ statements** that **performs a specific task** when executed by a computer.
- Usually written by a **computer programmer** with a **programming language** such as C, C++.
- **Computer programming** is the craft of writing useful, maintainable, and extensible source code which can be interpreted or compiled by a computing system to perform a meaningful task.

Header Files

- The **files that are specified in the include preprocessor** is called as header files.
- Header files are **precompiled** files that has some functions defined in them.
- We **can call** those functions from our program.
- Header file is given an extension **.h**
- C Source file is given an extension **.c**

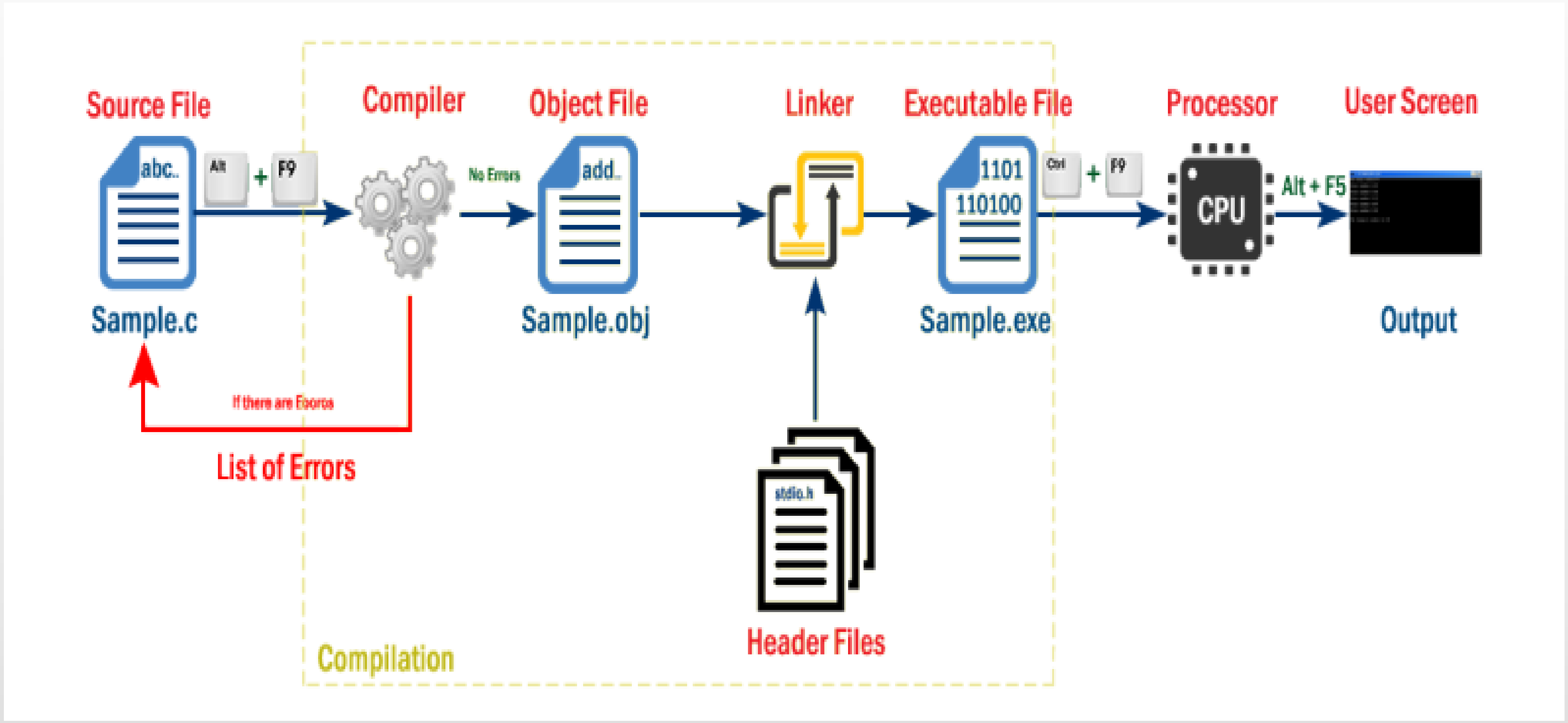
Compiling and Running a C Program

- **Type** your C program
- **Save** it with .c file extension
- **Compile**
- **Run**

Compiler

- A **Compiler** computer program which reads **source code** and process output **to assembly code or executable code** is called compiler.
- Compiler is language translator.
- A language translator is a **software** which translates the programs from a source language that are in human readable form into an equivalent program in an object language.
- The **source language** is generally a high-level programming language, and the **object language** is typically the machine language of an actual computer.

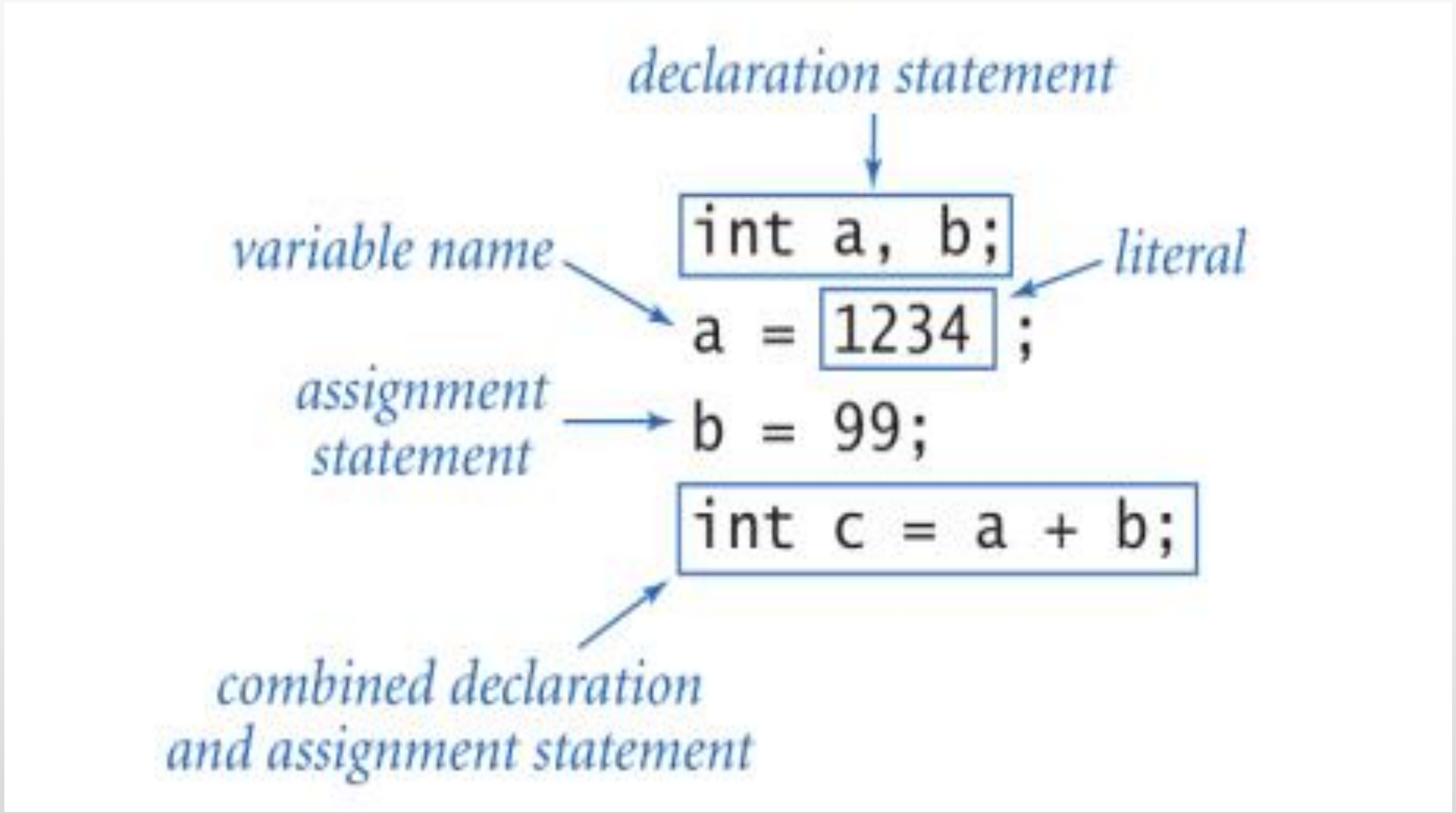
C Program Compilation and Execution



Variables

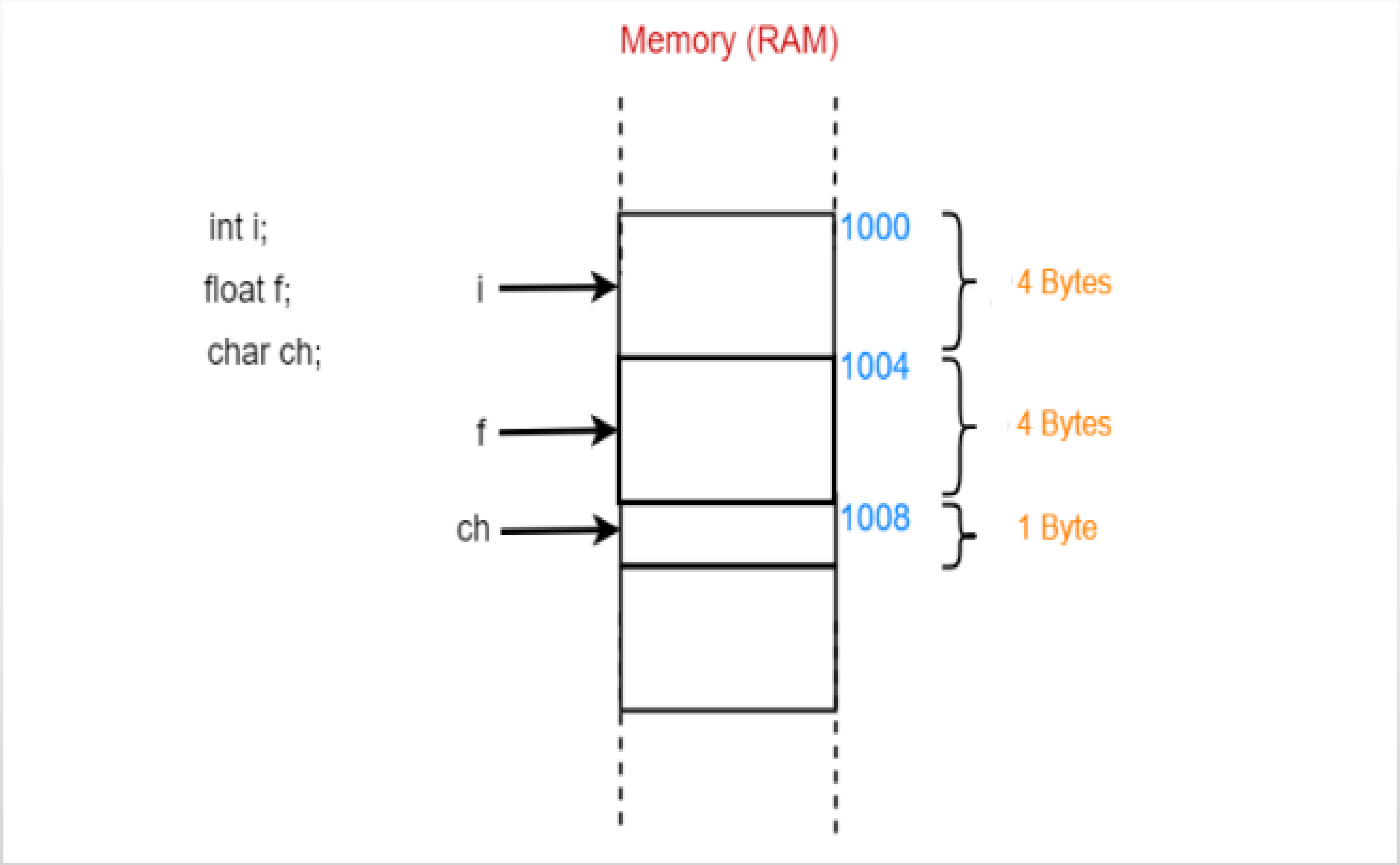
- A variable is a **named data storage location** in the computer's **Random Access Memory (RAM)** to manipulate (store, compute, retrieve etc.) the data
- A variable must be declared with it's **data type**

Variables



```
int a, b;
a = 1234 ;
b = 99;
int c = a + b;
```

Variable Declaration



Variable Assignment

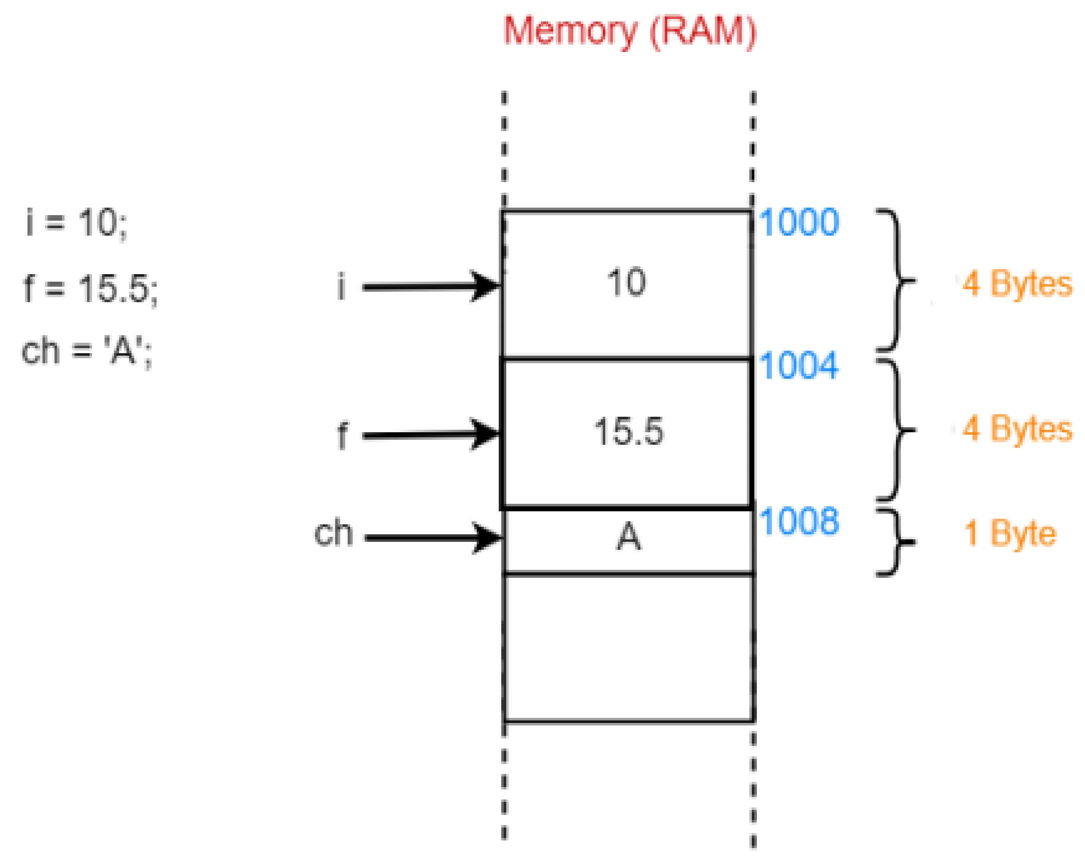
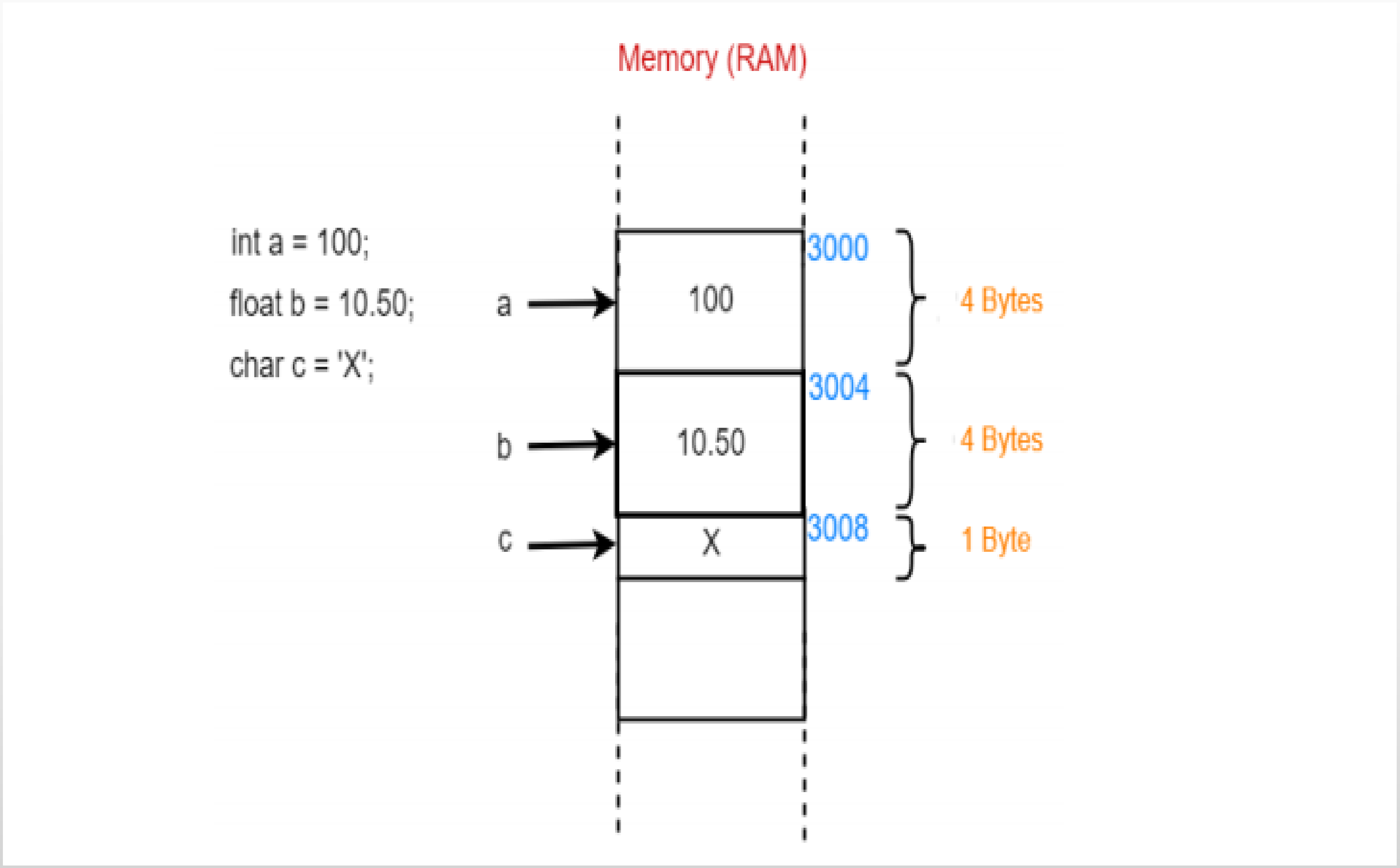


Figure 5: Variable Assignment

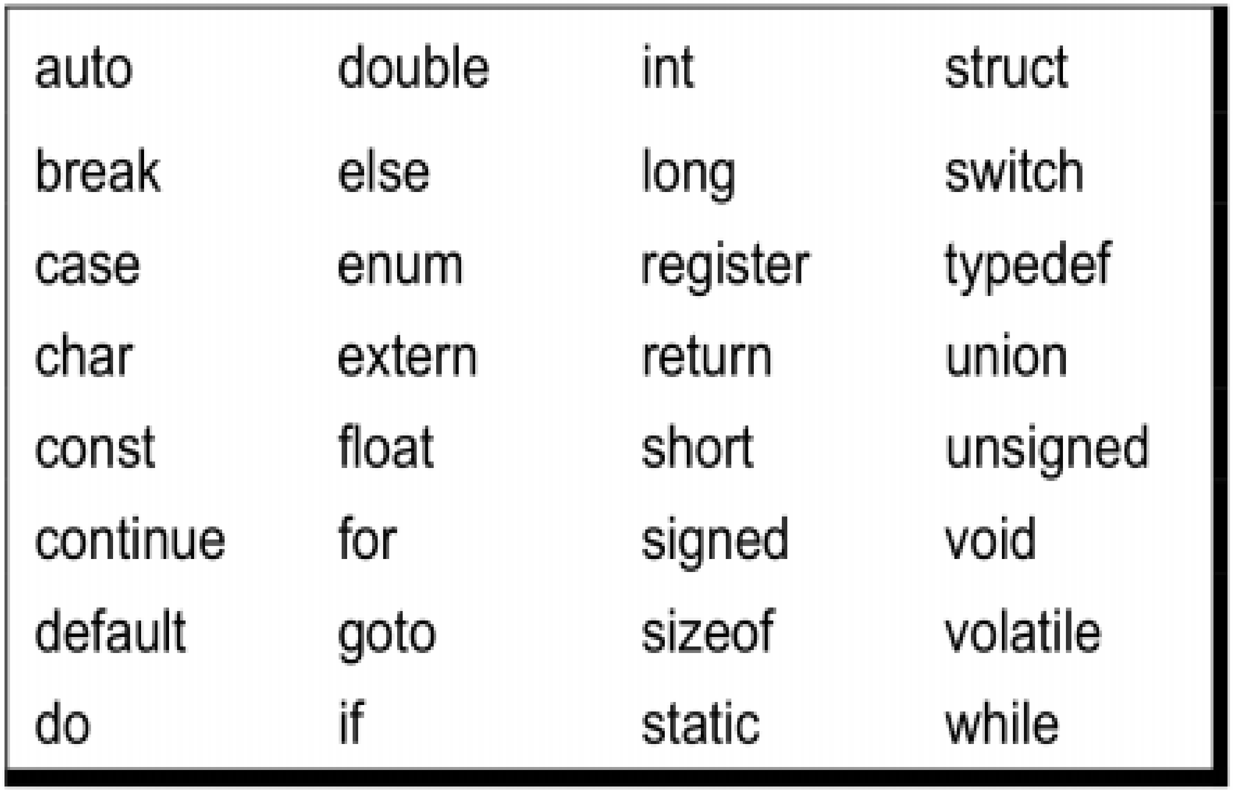
Variable Declaration and Initialization



Variable Naming Rules in C

- Variable name can contain **letters** (a to z and A to Z), **digits** (0 to 9), and the **underscore** character ( ).
- The **first character of the name must be a letter.** The underscore is also a legal first character, but not recommended.
- A digit (0 to 9) **cannot** be used as the first character.
- **C is case-sensitive**, thus, the names count and Count refer to two different variables.
- **C keywords can not** be used as variable names

C Keywords



auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Data Types in C

- int
- char
- float
- double
- void

- The sizes and ranges of data types are compiler (bit) dependent!

Statement

- A **statement** is a complete instruction given to the computer to do some task.
- Usually written line by line, ended by a semicolon (;)
- for example,
  - int x=2;
  - x=2+4;
  - printf("This is my program.");
- A statement can be a null statement also
- A computer program is made up of series of statements

NULL Statement

- A null statement is allowed in C however it does not do anything!
- for example,

//It’s a null statement

or,

/*Again, it’s a null statement.

This statement will not execute.*/

Input & Output

- Input
  – scanf(¨ %d¨ , &a); /* for taking an integer input */
- Output
  – printf(¨ %d¨ , a); /* for printing out an integer */

Sample Program

```
#include<stdio.h>
int main()
{
   int num;
   num=100;
   printf(“The value of the variable is %d”, num);
   return 0;
}
```

Sample Program

```
#include<stdio.h>
int main()
{
   int i=10;
   float f=12.2;
   char c=‘a’;
   printf(“i is %d”, i);
   printf(“f is %f”, f);
   printf(“c is %c”, c);
   return 0;
}
```

Practice

- Write a program that declares one integer value and give this variable the value 2000 and then using **printf()** statement, display the value on the screen like this:

  The value of the variable in this program is 2000

- Write a program that inputs two double numbers and then display both the numbers.

```
#include<stdio.h>
int main()
{
   char c='0';
   printf("The output: %c", c);
   return 0;
}
```

```
#include<stdio.h>
int main()
{
   char c='0';
   printf("The output: %d", c);
   return 0;
}
```

```
#include<stdio.h>
int main()
{
    char c='0';
    printf("The output: %c", c);
    return 0;
}
```

The output: 0

```
#include<stdio.h>
int main()
{
    char c='0';
    printf("The output: %d", c);
    return 0;
}
```

```c
#include<stdio.h>
int main()
{
   char c=‘0’;
   printf(“The output: %c”, c);
   return 0;
}
```

The output: 0

```c
#include<stdio.h>
int main()
{
   char c=‘0’;
   printf(“The output: %d”, c);
   return 0;
}
```

The output: 48

Operators

- An operator is a **symbol** which **operates** on a value or a variable.
- Symbols that are used to perform different types of mathematical and logical operations are called operators
- Types of operators in C language are:
  – Arithmetic operators (**+,-,*,/,%,++,--**)
  – Relational operators (**<,>,<=,>=,==,!=**)
  – Logical operators (**&&,||,!**)
  – Bitwise operators (**&,|**)
  – Assignment operators (**=**)
  – Conditional / Ternary operator (**? :**)
  – Special operators (**&,***)

Arithmetic Operators

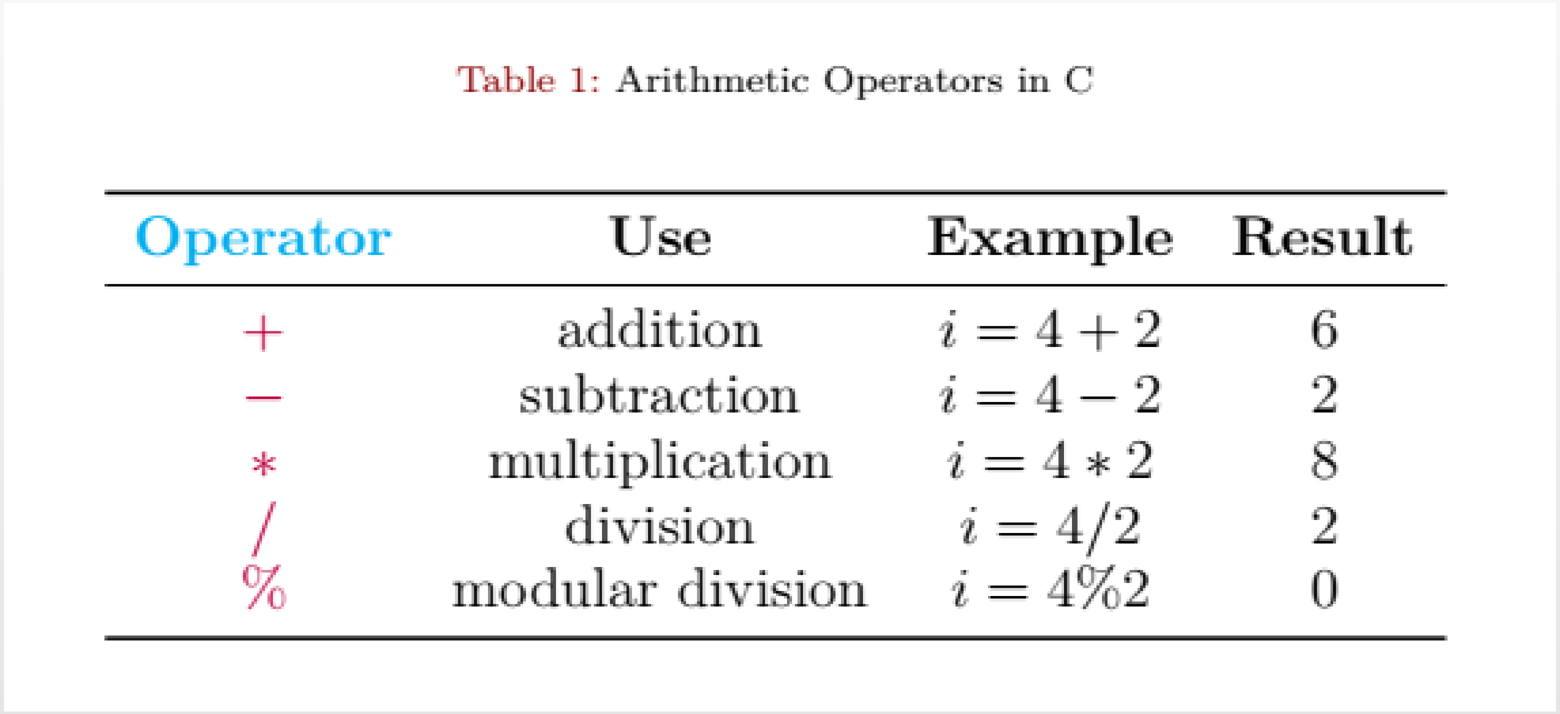


Table 1: Arithmetic Operators in C

Operator	Use	Example	Result
+	addition	$i = 4 + 2$	6
−	subtraction	$i = 4 - 2$	2
*	multiplication	$i = 4 * 2$	8
/	division	$i = 4/2$	2
%	modular division	$i = 4\%2$	0

% operator produces the remainder of an integer division. The % may be used with Integer types only.

Arithmetic Operators (Contd.)

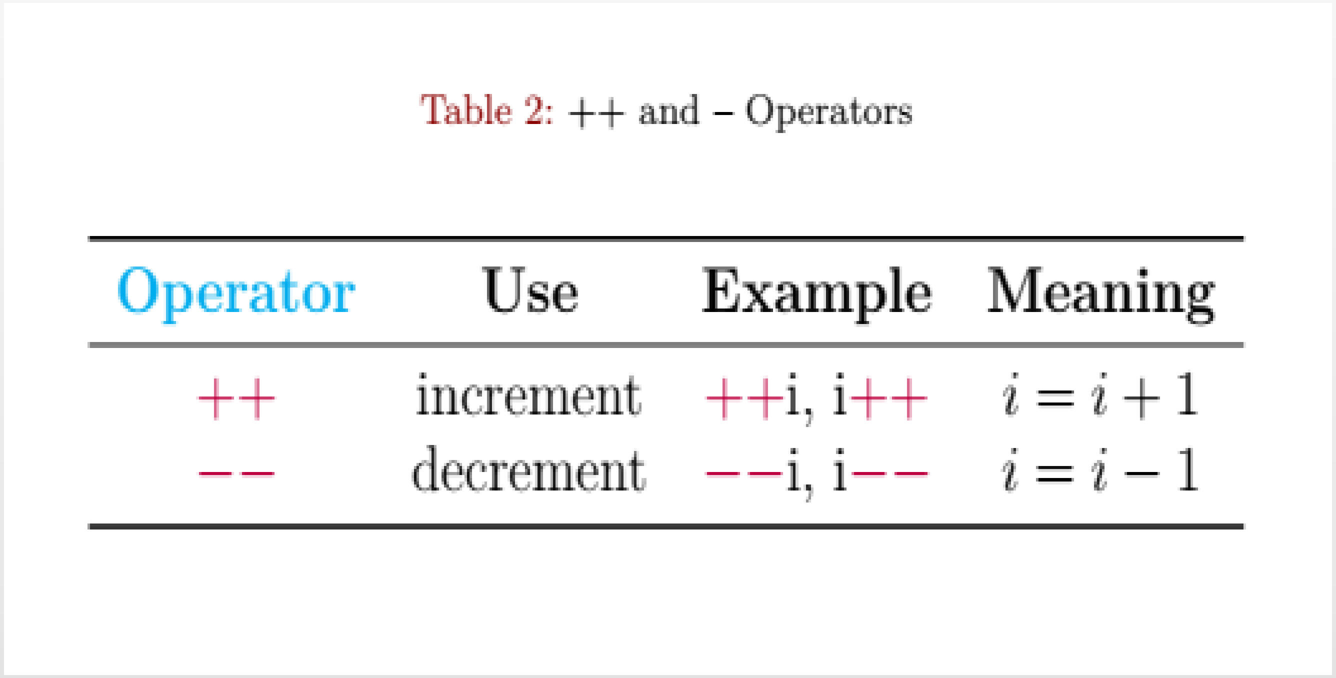


Table 2: ++ and – Operators

Operator	Use	Example	Meaning
++	increment	++i, i++	$i = i + 1$
−−	decrement	−−i, i−−	$i = i - 1$

Assignment Operator

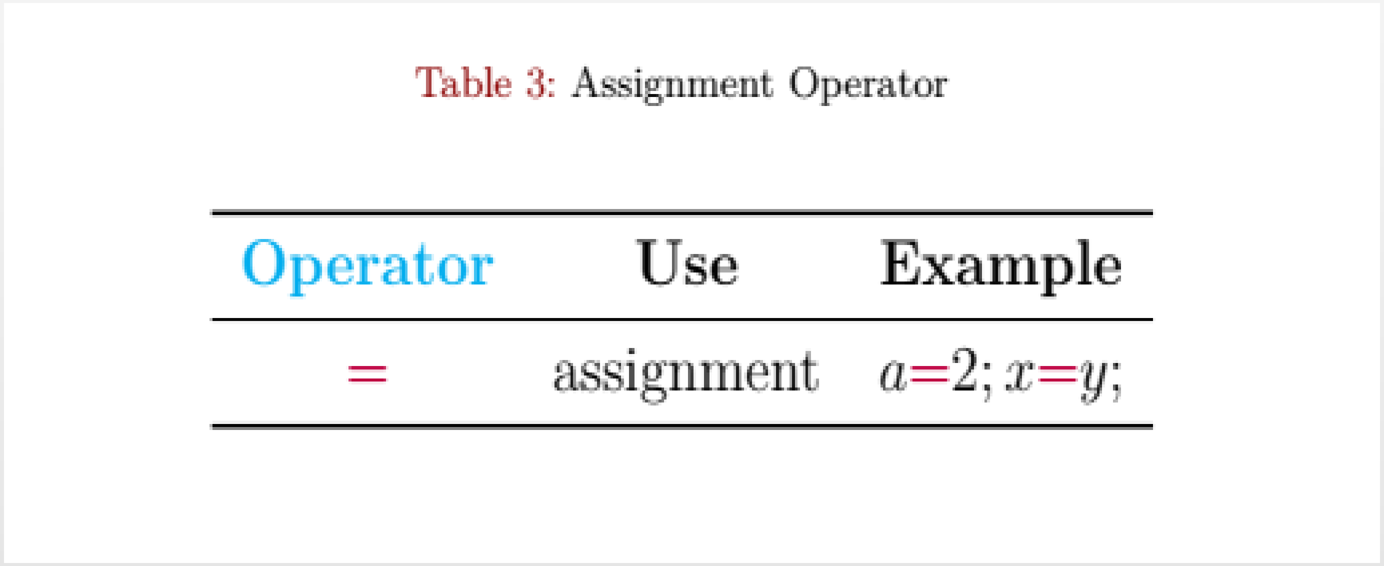
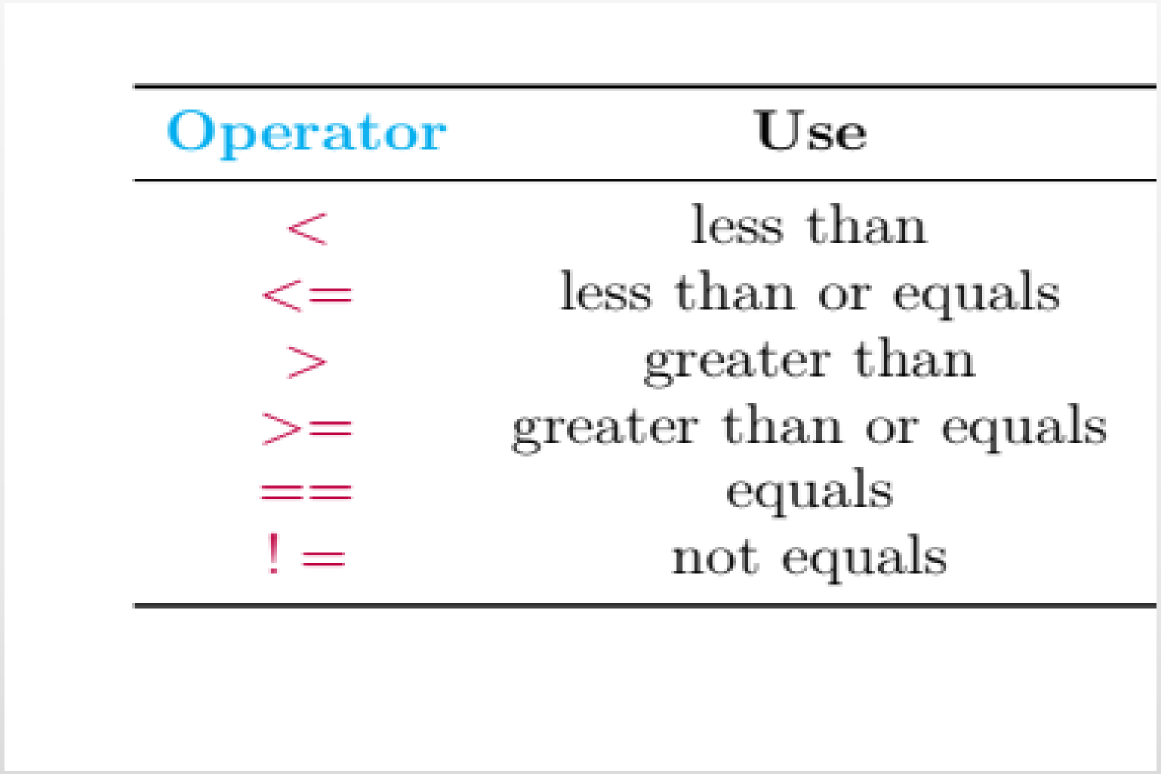


Table 3: Assignment Operator

Operator	Use	Example
=	assignment	a=2; x=y;

Relational Operators



Operator	Use
<	less than
<=	less than or equals
>	greater than
>=	greater than or equals
==	equals
! =	not equals

Example

```
#include<stdio.h>


int main()
{
   printf("%d ", 5/2);
   printf("%d ", 5%2);
   printf("%d ", 4/2);
   printf("%d ", 4%2);


}
```

Practice

- Write a program that will ask the user to give two inputs which will represent the width and height of a rectangle. Then compute the area of the rectangle and display the result.

- Area of a rectangle= height x width