University of Batna 2
Institute of safety and security
1st year LMD

Chapter 2: Basics of C++

Definition of Algorithm

- **Dictionary definition**
  - A procedure for solving a mathematical problem in a finite number of steps that frequently involves repetition of an operation
  - A step-by-step method for accomplishing a task
- **Informal description**
  - An ordered sequence of instructions that is guaranteed to solve a specific problem

Definition of Algorithm

- An algorithm is a list that looks like
  - STEP 1: Do something.
  - STEP 2: Do something.
  - STEP 3: Do something.
  - . .
  - . .
  - . .
  - STEP N: Stop. You are finished.

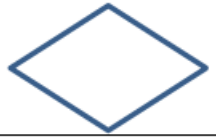
Definition of Algorithm

- We use algorithms all the time
  - **Examples ?**
    - Following directions
    - cooking a meal
    - Adding two numbers
    - Finding Greatest Common Divisor

Flowchart

- The flowchart is a diagram which visually presents the flow of data through processing systems. Algorithms are nothing but sequence of steps for solving problems. So a flow chart can be used for representing an algorithm.

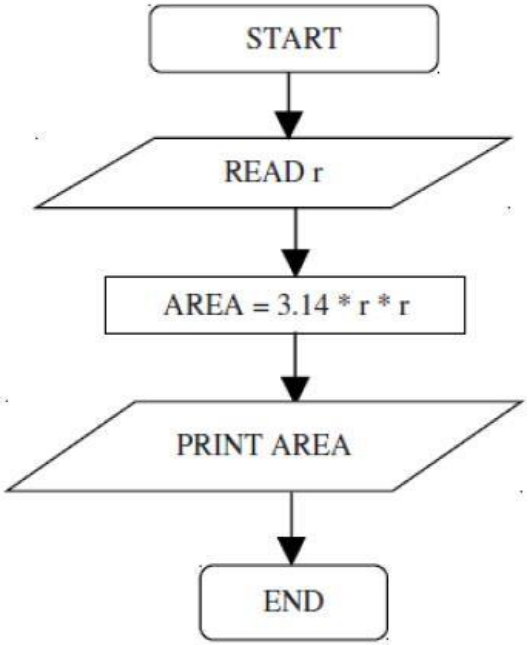
Flowchart

Symbol	Function
	starting or ending of the program
	Indicates any type of internal operation inside the Processor or Memory
	Used for any Input / Output (I/O) operation. Indicates that the computer is to obtain data or output results.
	Used to ask a question that can be answered in a binary format (Yes/No, True/False)
	Used for connection,
	Shows direction of flow.

Flowchart Symbols

Flowchart

- Example: draw a flowchart to Find the area of a circle of radius r.



C++ Program Structure

Let us look at a simple code that would print the words Hello World

```
#include <iostream>
using namespace std;
int main(){
cout << "Hello world!" << endl;
return 0;
}
```

C++ Program Structure

1. The C++ language defines several headers, which contain information that is either necessary or useful to your program. For this program, the header **<iostream.h>** is needed for output string in the screen.
2. **int main()** : is the main function where program execution begins.
3. **//** : is a single-line comment available in C++. Single-line comments begin with **//** and stop at the end of the line.
4. **cout << " : This is my first C++ program.";** causes the message "This is my first C++ program" to be displayed on the screen.
5. **<<** : it is the send operator
6. **return 0**: terminates main() function and causes it to return the value 0 to the calling process.
7. **;** : semicolon , its used as terminator for every C++ statement.

The Programming Process

- Identify the Problem - What Are You Trying To Do?
  - Requirements
  - Specification
- Design a Solution - How Is It Going To Be Done?
- Write the Program - Teaching the Computer
  - Code
  - Compile
  - Debug
- Check the Solution - Testing it Understands You

Variables and Constants

- Programs need a way to store the data they use. Variables and constants offer various ways to represent and manipulate data.
- Constants, as the name suggests, have fixed values. Variables, on the other hand, hold values which can be assigned and changed as the program executes.
- A variable/constante is designed by its name (identifier) and its type
- **Variable Declaration in C++:**

Declaration will allocate memory for specified variable with garbage value.

Syntax:

type identifiers_list ;

- **Constant Declaration in C++:**

Syntax:

const type identifier = value; ;

Variables and Constants

- **Identifiers:**
- A C++ identifier is a name used to identify a variable, function, class, …,or any other user-defined item.
- An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores, and digits (0 to 9).
- **Examples:** Here are some examples of acceptable identifiers:

  Mohd zara abc move_name
  a_123 cont1 flg_min A30m yname50
  _temp ja23b9 retVal retval
- **Examples of invalid identifiers:** 3v1, my name, True .

Variables and Constants

- **Data types in C++:**

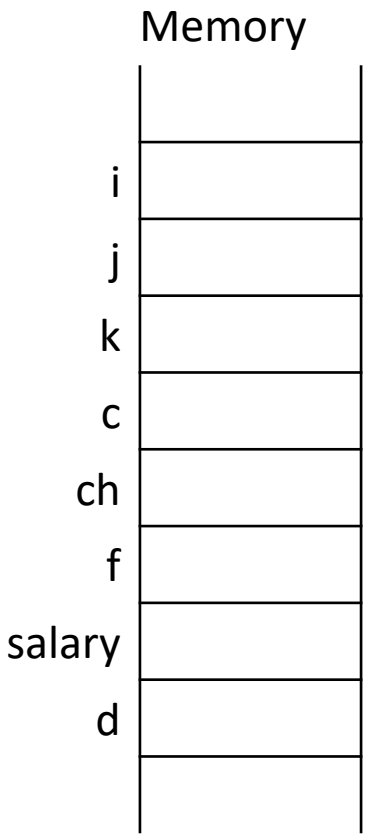
Every variable and constant has an associated type which defines the set of values that can be legally stored in it.

Type	Meaning	Minimum Size
char	character	8 bits
short	short integer	16 bits
int	integer	16 bits
long	long integer	32 bits
float	single-precision floating-point	6 significant digits
double	double-precision floating-point	10 significant digits

Variables and Constants

- **Example:**

```
int i, j, k;
char c, ch;
float f, salary;
double d;
```



Variables and Constants

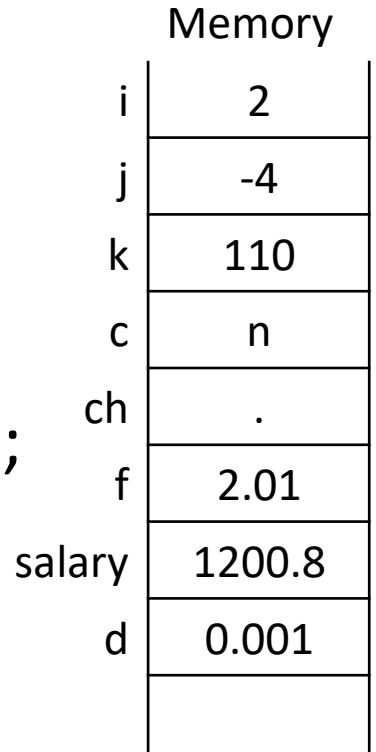
- **Initialization of Variable**
- Initialization means assigning value to declared variable. Every value will overwrite the previous value.
- **Example:**

  int i=2, j=-4, k=110;

  char c='n', ch='.';

  float f=2.01, salary=1200.8;

  double d=0.001;

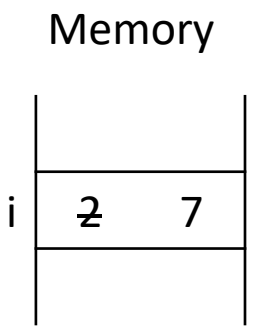


Variables and Constants

- **Variable assignment**
- The assignment statement indicates that the value given by the expression on the right hand side of the assignment operator (symbol =) must be stored in the variable named on the left hand side.
- The assignment operator should be read as ``becomes equal to'' and means that the variable on the left hand side has its value changed to the value of the expression on the right hand side. Every value will overwrite the previous value.

- **Example:**

```
int i=2;
i=7;
```



C++ Operators

- An operator is a symbol that tells the compiler to perform specific mathematical or logical calculations on operands(variables).

- **Types of operators available in C++**

  - Arithmetic / Mathematical operator
  - Assignment operator
  - Increment Decrement operator
  - Relational operator
  - Logical operator
  - Unary operator

C++ Operators

- **Arithmetic Operator:**
- There are following arithmetic operators supported by C++ language: Assume variable A holds 10 and variable B holds 20, then:

Operator	Description	Example
+	Adds two operands	A + B will give 30
-	Subtracts second operand from the first	A - B will give -10
*	Multiplies both operands	A * B will give 200
/	Divides numerator by de-numerator	B / A will give 2
%	Modulus Operator and remainder of after an integer division	B % A will give 0

C++ Operators

- **Increment Decrement operator**
- Increment Decrement operators increase or decrease the operand by one value.
- Example: Assume A=10, find the output result for the following expressions

++	Increment operator, increases integer value by one	A++ will give 11
--	Decrement operator, decreases integer value by one	A-- will give 9

C++ Operators

- **Assignment operator**
- Assignment operator is used to copy value from right to left variable.
- Suppose we have:

=	Equal sign Copy value from right to left.	X = Y, now both X and Y have 2
+=	**Plus Equal operator** to increase the left operand by right operand.	X+=5 → X=X+5 will give X= 10
-=	**Minus Equaloperator** will return the subtraction of right operand from left operand.	Y-=1 → Y= Y-1 will give Y=1
*=	**Multiply Equal operator** will return the product ofright operand and left operand.	X *= Y → X = X * Y,X = 10
/=	**Division Equal operator** will divide right operand by left operand and return the quotient.	X /= Y → X = X / Y,X = 2.5
%=	Modulus Equal to operator will divide right operandby left operand and return the mod (Remainder).	X %= Y is similar to X = X % Y, now X is 1

C++ Operators

- **Logical Operator:**
- Logical operators are used in situation when we have more then one condition in a single if statement.
- Suppose we have, int X = 5, Y = 2;

Operator	Name	Description	Example
&&	AND	Return true if all conditions are true, return false if any of the condition is false.	if(X > Y && Y < X) will return true
||	OR	Return false if all conditions are false, return true if any of the condition is true.	if(X > Y || X < Y) will return true
!	NOT	Return true if condition if false, return false if condition is true.	if(!(X>y)) will return false

Basic Input/Output

- **Standard output (cout)**
- The **cout** instruction allows you to display one or more objects on the screen.

  Syntax:

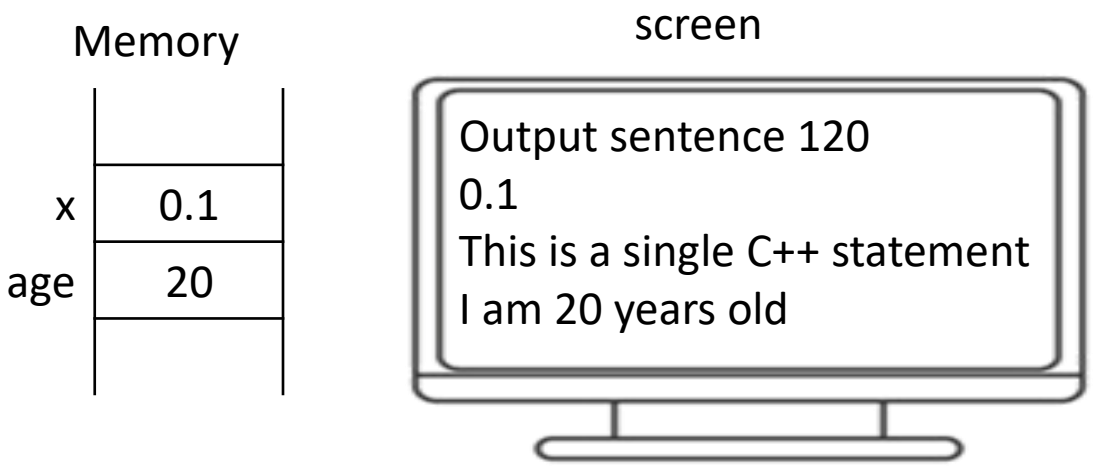
  cout<< objects_list;

- **objects_list** is either one object or several ones separated by the separator <<.
- An object can be a value, a constant, an arithmetic expression, a variable or a character string constant.
- To insert a line break, a new-line character shall be inserted at the exact position the line should be broken. In C++, a new-line character can be specified as \n or using endl manipulator

Basic Input/Output

- **Standard output (cout)**
- **Example:**

```
float x=0.1; int age = 20;
cout << "Output sentence";
cout << 120 <<endl;
cout << x <<endl;
cout << "This " << " is a " << "single C++ statement" <<endl;
cout << "I am " << age << " years old " <<endl;
```



Basic Input/Output

- **Standard input (cin)**
- In most program environments, the standard input by default is the keyboard, and the C++ stream object defined to access it is **cin**.
- For formatted input operations, cin is used together with the extraction operator, which is written as >> (i.e., two "greater than" signs). This operator is then followed by the variable where the extracted data is stored.
- **Example:**

```
int age;
cin >> age;
```