# **Lab Manual 02**

## **Tokens & Escape Sequences**

**Lab Objectives:**

At the end of this lab students will know about

* **Tokens**
* **Types of escape sequences**
* **Uses of escape sequences**
* **How to use in C++ programs**
* **What is a data type?**
* **Different types of data types used in C++**
* **How to use each data type**

**Tokens**

Tokens are the minimal chunk of program that have meaning to the compiler –the smallest meaningful symbols in the language. Our code displays all 6 kinds of tokens, though the usual use of operators is not present here:

|  |  |  |  |
| --- | --- | --- | --- |
| Token type | Description/Purpose | Examples |  |
| Keywords | Words with special meaning to the compiler | int, double,for, if |
| Identifiers | Names of sources in which input or any data is stored | cout, x1, cin |
| Literals | Basic constant values | 24.3, “Hello world” |
| Operators | To perform operations like logical, arithematic etc | &, ||, +, - |
| Punctuation/Separators | Defining the structure of program | {,},(, ; |
| Whitespace | Spaces of various types use for formatting | Newline, tab, backslash |

**Escape sequences:**

– a symbol used to represent a special character in a text literal. The \n indicates a newline character. It is an example of an escape sequence. Here are all the C++ escape sequences which you can include in strings:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | | |  |  |  | |  |  |  |  |  |  |
|  | Escape Sequence | | | |  |  | Represented Character | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

\a System bell (beep sound)

\b Backspace

\f Form feed (page break)

\n Newline (line break)

|  |  |
| --- | --- |
| \r | “Carriage return” (returns cursor to start of line) |

\t Tab

\\ Backslash

\’ Single quote character

\" Double quote character

**Data Types**

Every expression has a type – a formal description of what kind of data its value is. For instance, 0 is an integer, 3.142 is a floating-point (decimal) number, and "Hello, world!\n" is a string value(a sequence of characters). Data of different types take a different amounts of memory to store. Here are the built-in data types we will use most often:

|  |  |  |  |
| --- | --- | --- | --- |
| **Type Names** | **Description** | **Size** | **Range** |
| char | Single text character or small integer. Indicated with single quotes (’a’, ’3’). | 1 byte | signed: -128 to 127 unsigned: 0 to 255 |
| int | Larger integer. | 4 byte | signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295 |
| boolean | Boolean (true/false). Indicated with the keywords true and false. | 1 byte | Just true (1)or false (0). |
| double | “Doubly” precise floating point number. | 8 byte | +/-1.7e +/-308 ( 15 digits) |

**For Example:**

#include<iostream>

#include<conio.h>

using namespace std;

main()

{

int x=3;

y=3\*x;

cout<< “value of x =”<< x<< “value of y=” << y;

getch();

}

**Question#01**

**Print Diamond using escape sequence “\n” and “\t”? In single cout statement?**

**Output should be like**

**\***

**\* \* \***

**\* \* \* \* \***

**\* \* \***

**\***

**Question #02**

**Print your Result Card using required escape sequences? Print all other information as it is but get input from user in obtained marks and calculate the percentage of entered marks?**

**Output:**

**Result Card**

**Name: ABC Reg. #: CIIT/---------**

**Session: Fall-16 Semester: 2**

**Subjects Total Marks Obtained Marks**

**Digital Logic Design 100 \_\_\_user input\_\_\_**

**Electronics-I 100 \_\_\_user input\_\_\_**

**English 100 \_\_\_user input\_\_\_**

**Percentage= %**

**Question#03**

**Develop a C++ program that declare almost 6 types of identifiers and display their sizes in bytes on the screen.**

**Question #04**

**Develop a C++ program that prints the table of 2 without using loop?**

**Question #5:**

**write a program to swap the value of a and b**

**let a=10 and b = 20**

**your output should be**

**a = 20 and b = 10**

