# Fundamentals of Computer Programming

*CS-110*

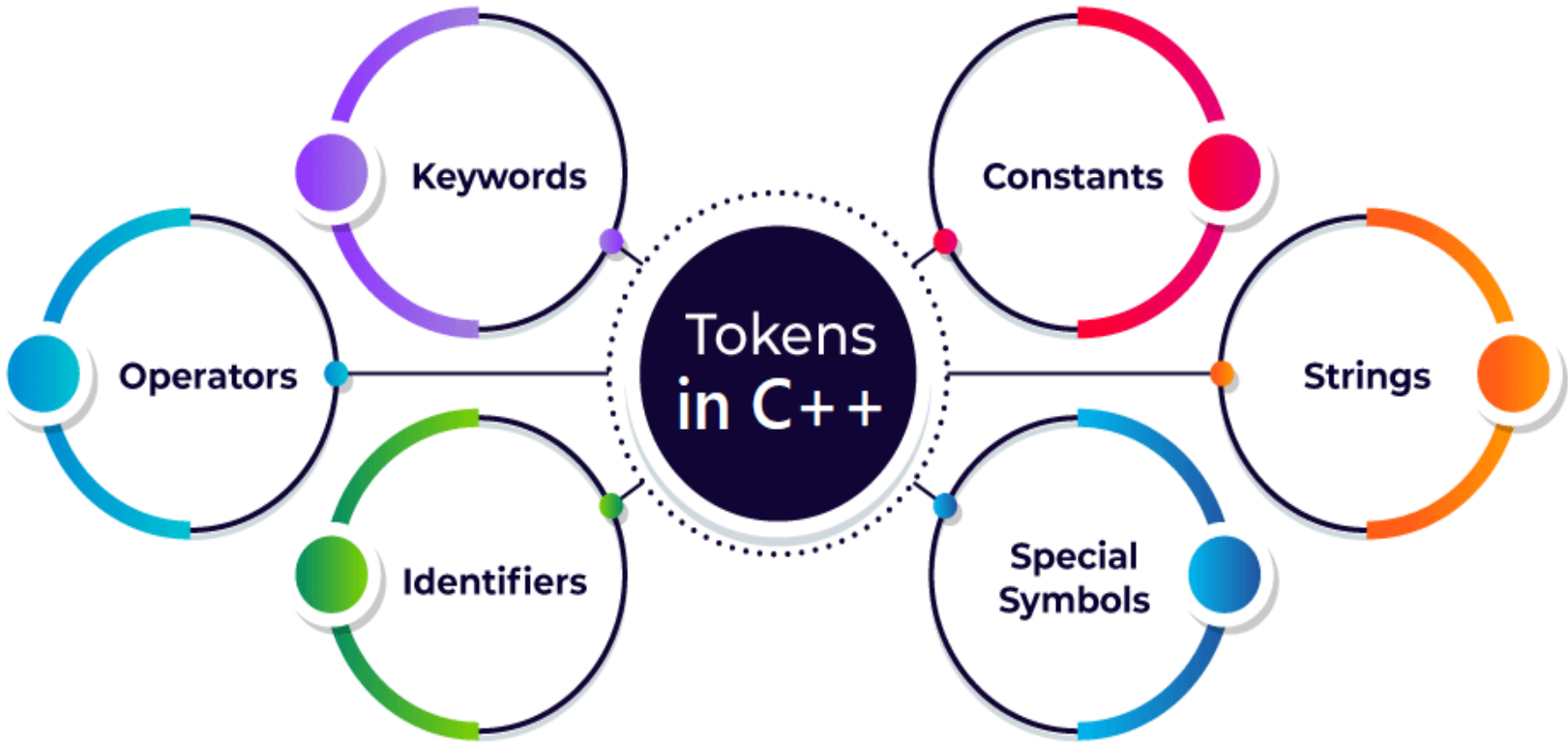
*Course Instructor: Dr. Momina Moetesum*

# Basic I/O, Variables, Constants, Escape Sequences

*Week 2-A*

```
51 template <typename T, typename I>
52 void TestIntegral(const T values[], int num_values) {
53   for (int i = 0; i < num_values; ++i) {
54     T t0 = values[i];
55     I i0 = absl::bit_cast<I>(t0);
56     T t1 = absl::bit_cast<T>(i0);
57     I i1 = absl::bit_cast<I>(t1);
58     ASSERT_EQ(0, memcmp(&t0, &t1, sizeof(T)));
59     ASSERT_EQ(i0, i1);
60   }
61 }
62
63 TEST(BitCast, Bool) {
64   static const bool bool_list[] = { false, true };
65   TestMarshall<bool>(bool_list, ABSL_ARRAYSIZE(bool_list));
66 }
67
68 TEST(BitCast, Int32) {
69   static const int32_t int_list[] =
70     { 0, 1, 100, 2147483647, -1, -100, -2147483647, -2147483647-1 };
71   TestMarshall<int32_t>(int_list, ABSL_ARRAYSIZE(int_list));
72 }
73
74 TEST(BitCast, Int64) {
75   static const int64_t int64_list[] =
76     { 0, 1, 1LL << 40, -1, -(1LL<<40) };
77   TestMarshall<int64_t>(int64_list, ABSL_ARRAYSIZE(int64_list));
```

# C++ Tokens


Tokens in C++
Keywords
Constants
Operators
Strings
Identifiers
Special Symbols

# Keywords

Keywords in C++ are the tokens that are the reserved words in programming languages that have their specific meaning and functionalities within a program.

**There are 95 keywords reserved in C++. Some of the main Keywords are:**

| | | | | | | |
|---|---|---|---|---|---|---|
| break | try | catch | char | class | const | continue |
| default | delete | auto | else | friend | for | float |
| long | new | operator | private | protected | public | return |
| short | sizeof | static | this | typedef | enum | throw |
| mutable | struct | case | register | switch | and | or |
| namespace | static_cast | goto | not | xor | bool | do |
| double | int | unsigned | void | virtual | union | while |

Safeguarding:
using namespace std;  OR
std::keyword

# Identifiers

In C++, entities like variables, functions, classes, or structs must be given unique names within the program so that they can be uniquely identified.

The unique names given to these entities are known as “identifiers”

| Valid Identifiers | Invalid Identifiers |
|---|---|
| _name | #name (Cannot start with special character except ‘_’) |
| Number89 | 2num (Cannot start with a digit) |
| first_name | first name (Cannot include space) |
| _last_name_ | string (Cannot be same as a keyword) |

# Rules for Identifiers

An identifier can only have alphabets, numbers, and the underscore _.

An identifier cannot begin with a number.

It is a preferred practice to begin an identifier with a lowercase character. For example, **name** is preferable to **Name**.

An identifier cannot be a keyword. For example, int is a keyword that is used to denote integers.

An identifier can start with an underscore. However, it's not considered a good practice.

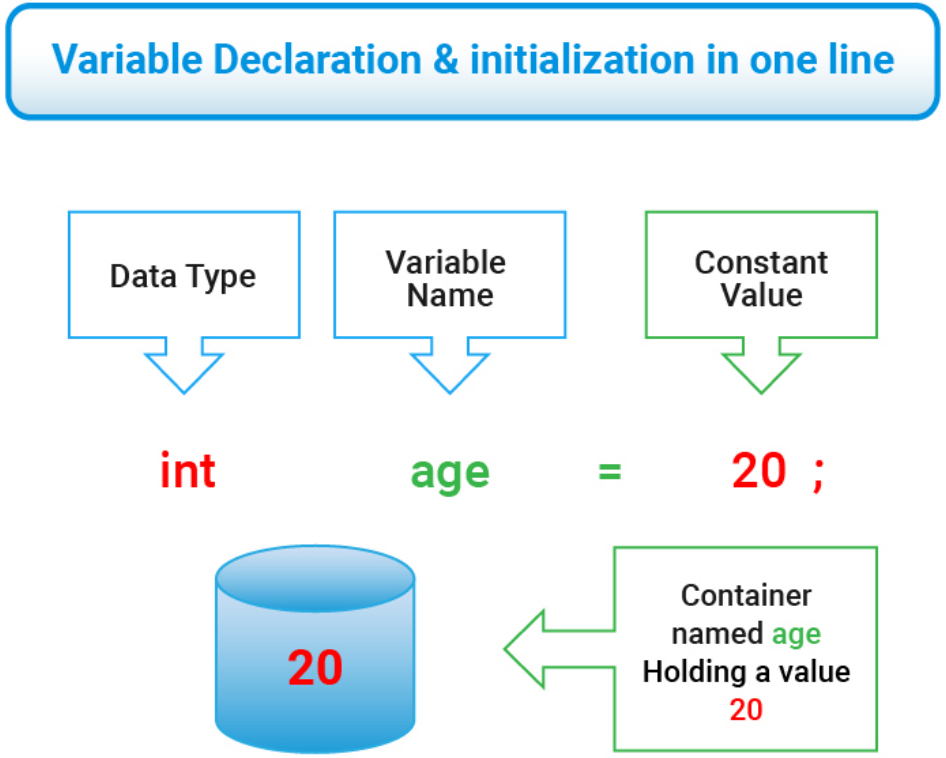
Special symbols are the token characters having specific meanings within the syntax of the programming language.

- **Semicolon (;):** It is used to terminate the statement.
- **Square brackets []:** They are used to store array elements.
- **Curly Braces {}:** They are used to define blocks of code.
- **Scope resolution (::):** Scope resolution operator is used to access members of namespaces, classes, etc.
- **Dot (.):** Dot operator also called member access operator used to access class and struct members.
- **Assignment operator '=':** This operator is used to assign values to variables.
- **Double-quote ("):** It is used to enclose string literals.
- **Single-quote ('):** It is used to enclose character literals.

# Variables

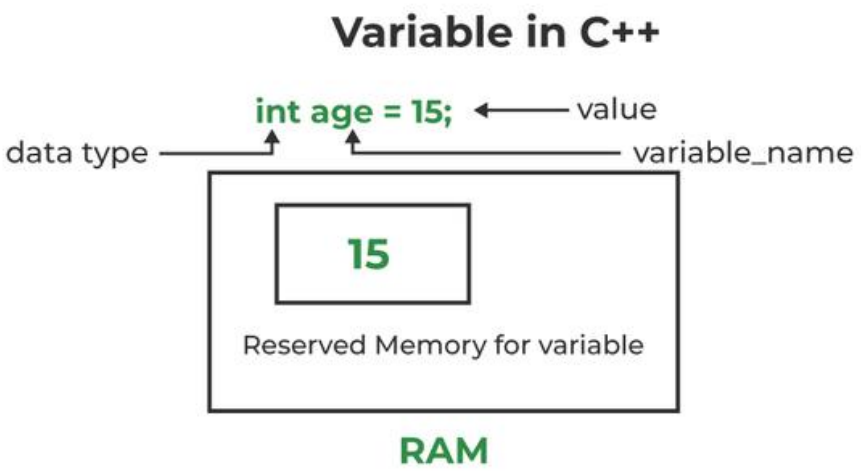
```
int age = 14;    // age is 14
age = 17;        // age is 17
```

- Variables in C++ is a name given to a memory location. It is the basic unit of storage in a program.
- The value stored in a variable can be changed during program execution.
- A variable is only a name given to a memory location, all the operations done on the variable effects that memory location.
- In C++, all the variables must be declared before use.



# Variables

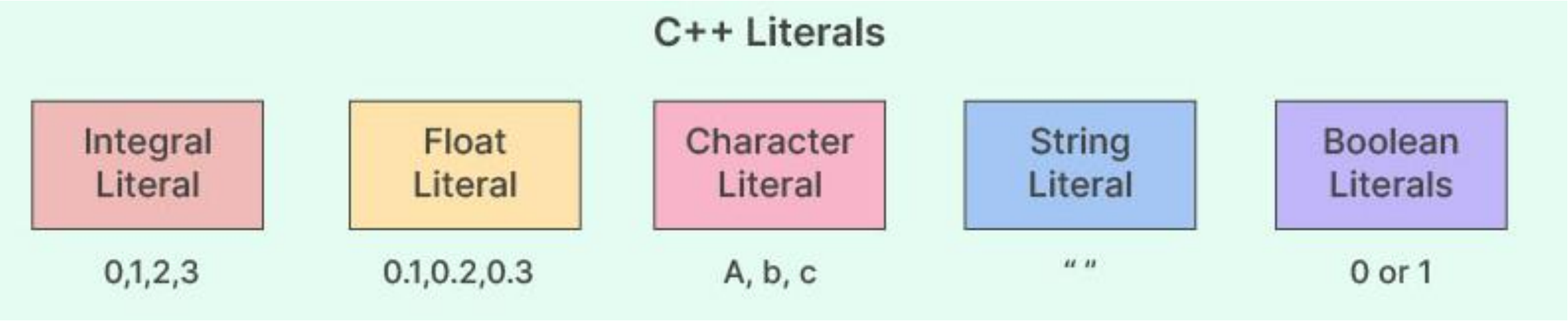
- Variable Declaration
- Variable Initialization
- Variable Definition




```cpp
/* C++ program to show difference between
 definition and declaration of a variable */
#include <iostream>
using namespace std;
int main()
{
    int a;  // this is declaration of variable a
    a = 10; // this is initialisation of a
    int b = 20; // this is definition = declaration + initialisation
    char a123 = 'a'; // declaration and definition of variable 'a123'
    // This is also both declaration and definition
    // as 'c' is allocated memory and assigned some garbage value.
    float c;
    int _c, _d45, e;  // multiple declarations and definitions
    int num1=34, num2=55, num3=68; // multiple definitions
    // Let us print a variable
    cout << a123 << endl;
    return 0;
}
```

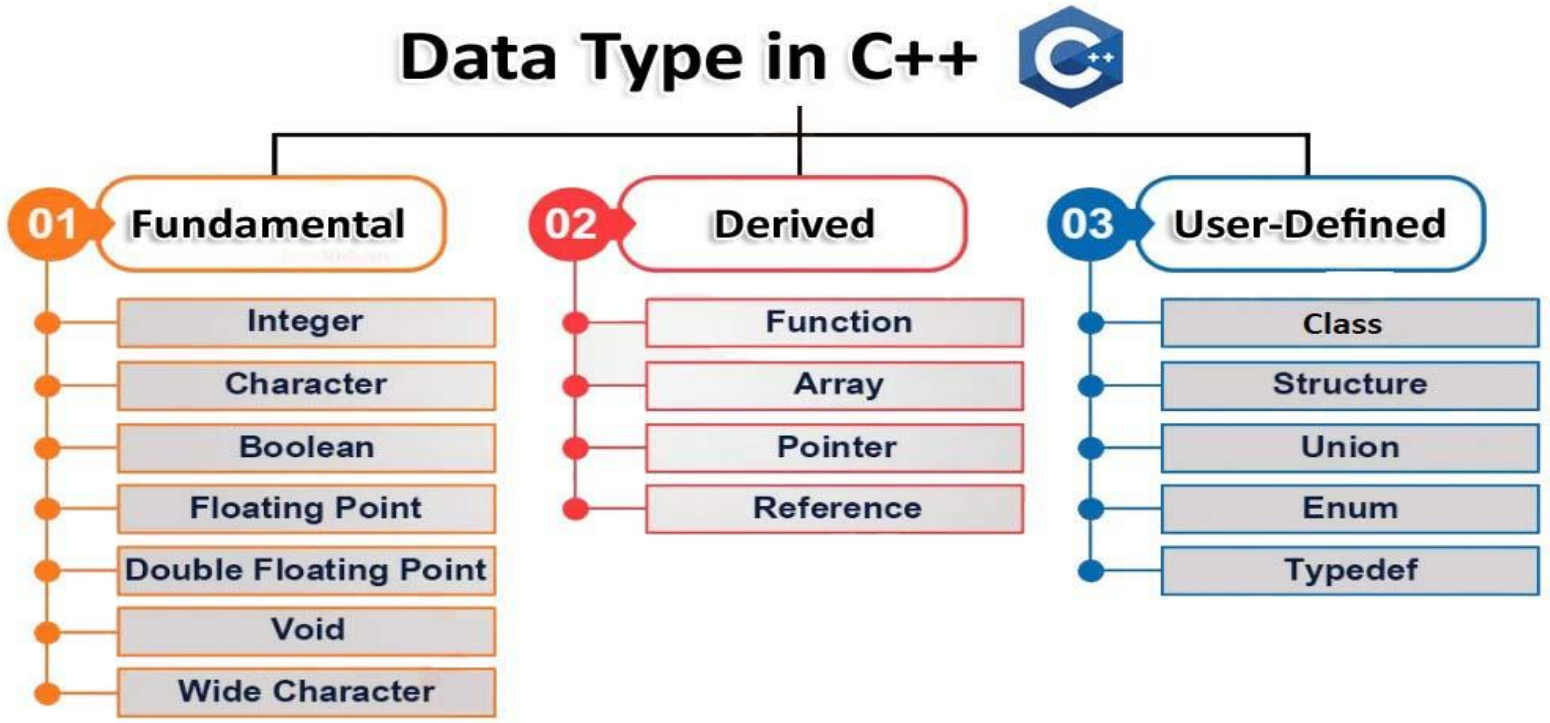
# Literals

- Literals represent fixed values that cannot be modified.
- Literals contain memory but they do not have references as variables.



# Data Types

C++ supports a wide variety of data types and the programmer can select the data type appropriate to the needs of the application. Data types specify the size and types of values to be stored.




int - stores integers (whole numbers), without decimals, such as 123 or -123

double - stores floating point numbers, with decimals, such as 19.99 or -19.99

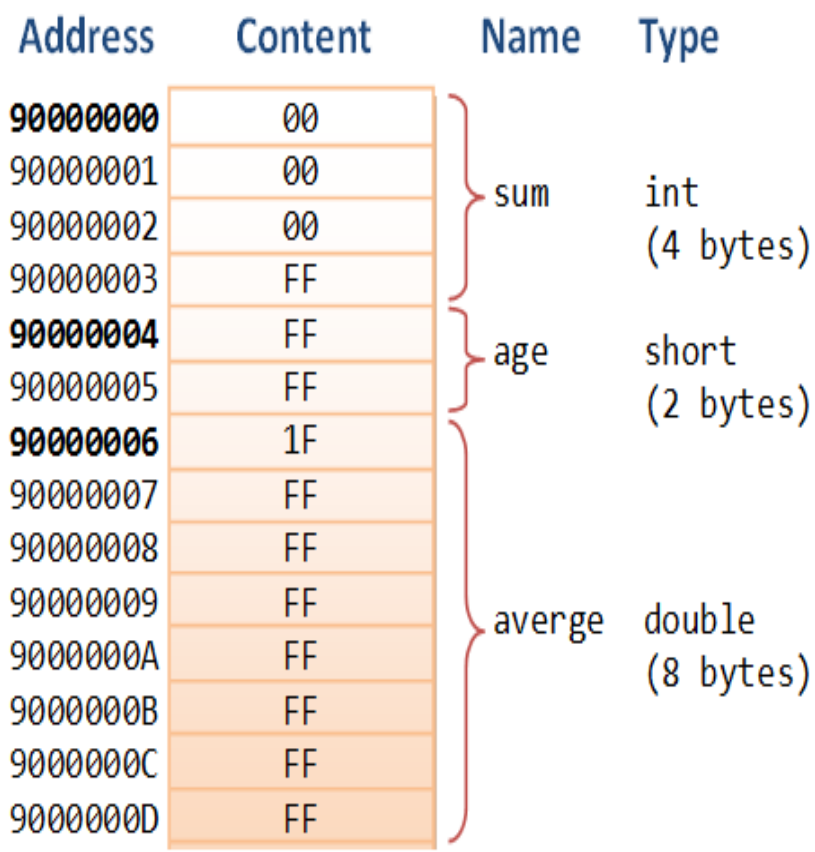
char - stores single characters, such as 'a' or 'B'. Char values are surrounded by single quotes

string - stores text, such as "Hello World". String values are surrounded by double quotes

bool - stores values with two states: true or false

# Data Types and Memory Allocation

| Data Type | Size (In Bytes) | Range |
|---|---|---|
| short int | 2 | -32,768 to 32,767 |
| unsigned short int | 2 | 0 to 65,535 |
| unsigned int | 4 | 0 to 4,294,967,295 |
| int | 4 | -2,147,483,648 to 2,147,483,647 |
| long int | 4 | -2,147,483,648 to 2,147,483,647 |
| unsigned long int | 4 | 0 to 4,294,967,295 |
| long long int | 8 | -(2^63) to (2^63)-1 |
| unsigned long long int | 8 | 0 to 18,446,744,073,709,551,615 |
| signed char | 1 | -128 to 127 |
| unsigned char | 1 | 0 to 255 |
| float | 4 | |
| double | 8 | |
| long double | 12 | |
| wchar_t | 2 to 4 | 1 wide character |

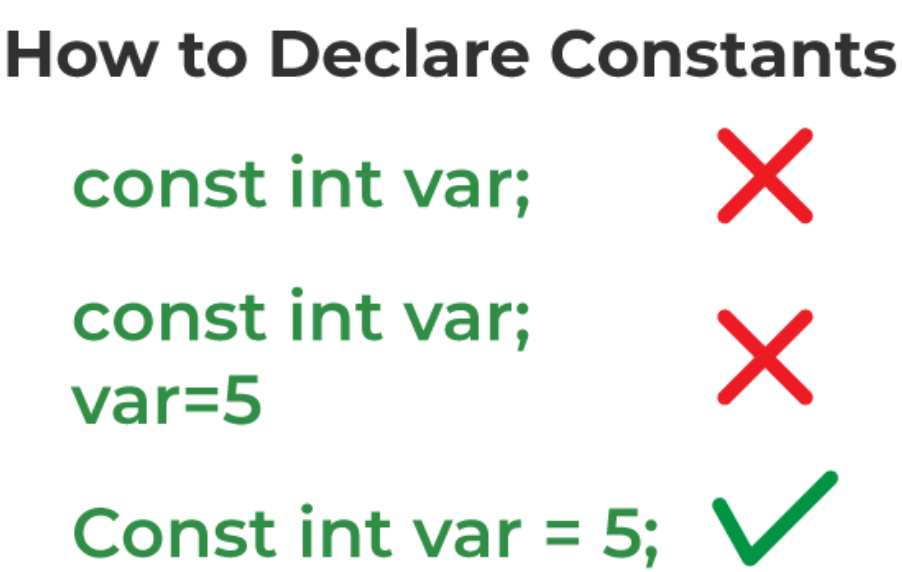



C and C++ language provide the **sizeof()** operator to check the size of the data types

# Constants

- Constants are variables whose values cannot be modified once they are declared in the program.
- In C and C++ language, the **const** keyword is used to define the constants.

```cpp
// C++ program to illustrate constant variable definition
#include <stdio.h>
int main()
{
    // defining integer constant using const keyword
    const int int_const = 25;
    // defining character constant using const keyword
    const char char_const = 'A';
    // defining float constant using const keyword
    const float float_const = 15.66;
    cout<<"Printing value of Integer Constant="<<int_const;
    cout<<"Printing value of Integer Constant="<<char_const;
    cout<<"Printing value of Integer Constant="<<float_const;
    return 0;
}
```




Read About Constants in C++:https://www.geeksforgeeks.org/different-ways-declare-variable-constant-c-c/?ref=lbp

# Basic User Input/Output

- **#include <iostream>** is a header file library that lets us work with input and output objects
- **cout** is pronounced "see-out". Used for output, and uses the **insertion** operator (<<)
- **cin** is pronounced "see-in". Used for input, and uses the **extraction** operator (>>)

```
int x, y;
int sum;
cout << "Type a number: ";
cin >> x;
cout << "Type another number: ";
cin >> y;
sum = x + y;
cout << "Sum is: " << sum;
```

# Escape Sequences

- Sometimes, it is necessary to use characters that cannot be typed or has special meaning in C++ programming. For example, newline (enter), tab, question mark, etc.
- In order to use these characters, escape sequences are used.
- Try the following:
- https://ciphertrick.com/c-beginners-introduction-to-escape-sequences/
- https://programmingdigest.com/escape-sequence-in-c-with-examples/

| Escape Sequence | Meaning |
|---|---|
| \n | New Line |
| \t | Horizontal Tab |
| \b | BackSpace |
| \r | Carriage Return |
| \a | Audible bell |
| \' | Printing single quotation |
| \" | printing double quotation |
| \? | Question Mark Sequence |
| \\ | Back Slash |
| \f | Form Feed |
| \v | Vertical Tab |
| \0 | Null Value |
| \nnn | Print octal value |
| \xhh | Print Hexadecimal value |

# Acknowledgment

- Content of these slides are taken from:
  - https://www.geeksforgeeks.org/
  - https://www.tutorialspoint.com/
  - https://www.programiz.com/