

Problem Solving

CONDITIONS

Sa, 20 NOV 2021





Programming language



Programming language

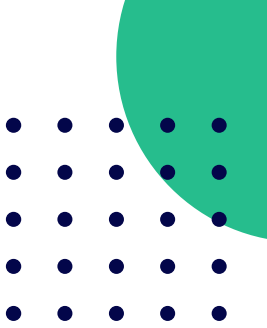
Low Level Language

- These languages give instructions to a computer in a way that is easily understood by the hardware of the computer.
- Two low-level languages are Machine Language and Assembly Language.
- Fast to run, but slow to write and understand.
- It needs assembler for translation.

High Level Language

- These languages are written in English-like language and easier for a human to understand but difficult for a computer to understand.
- There are many high-level languages e.g., C, C++, Java, COBOL, PHP, etc.
- It needs compiler or interpreter for translation.

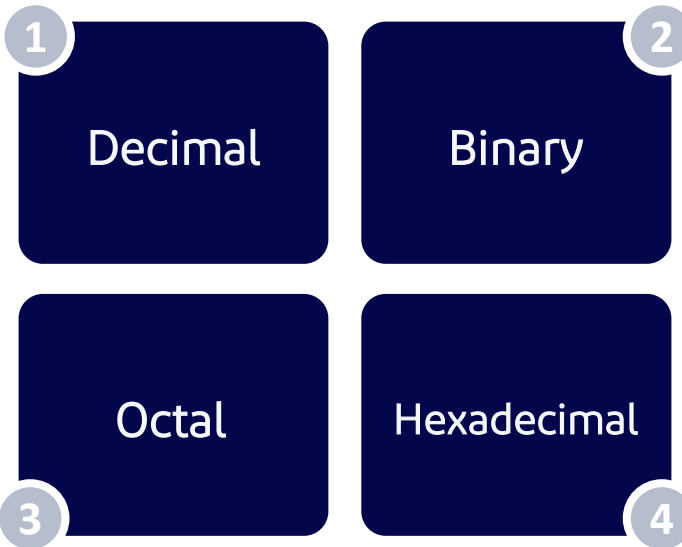




Number System

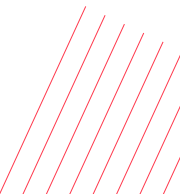
The decimal number system contains ten digits from 0 to 9 (base 10).

The octal number system has base 8 (means it has only eight digits from 0 to 7). There are only eight possible digit values to represent a number.



binary number system is used in the digital computers, for computers-since flip flops store either 0 or 1 base 2 (0, 1).

The number system has a base of 16 means there are total 16 symbols (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F) used for representing a number

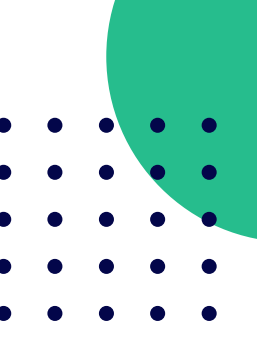



C++ Language

- C++ is used to develop games, desktop apps, operating systems, browsers, and so on because of its performance.
- C++ helps you to understand the internal architecture of a computer, how computer stores and retrieves information.
- C++ gives programmers a high level of control over system resources and memory.




+++++



Keyword in C++ Programming.

- Keywords are words that the language uses for a special purpose, such as void, int, public, etc.
- C++ programming is case sensitive, all keywords must be written in lowercase.



```
#include <iostream>

using namespace std;

int main()
{
    cout<<"Hello World!"<<endl;

    return 0;
}
```





Data Types





Data Types

- While writing program in any language, you need to use various variables to store various information.
- Variables reserved memory locations to store values.
- All variables use data-type during declaration to restrict the type of data to be stored. Therefore, we can say that data types are used to tell the variables the type of data it can store.





Data Types

- Data types in C++ is mainly divided into three types:



Primitive Data Types

These data types are built-in or predefined data types and can be used directly by the user to declare variables.

Example:

- int
- double
- char



Derived Data Types

The data-types that are derived from the primitive or built-in data types are referred to as Derived Data Types.

Example:

- Function
- Array



User-Defined Data Types

These data types are defined by user itself. Like, defining a class in C++ or a structure.





Primitive Data Types



Integer

Keyword used for integer data types is int. Integers typically requires 4 bytes of memory space.



Float

Float is used for storing single precision floating point values or decimal values. Float variables typically requires 4 byte of memory space.



Boolean

Boolean data type is used for storing boolean or logical values. A boolean variable can store either true or false. Keyword used for boolean data type is bool.



Character

Character data type is used for storing characters. Keyword used for character data type is char. Characters typically requires 1 byte of memory space



Double

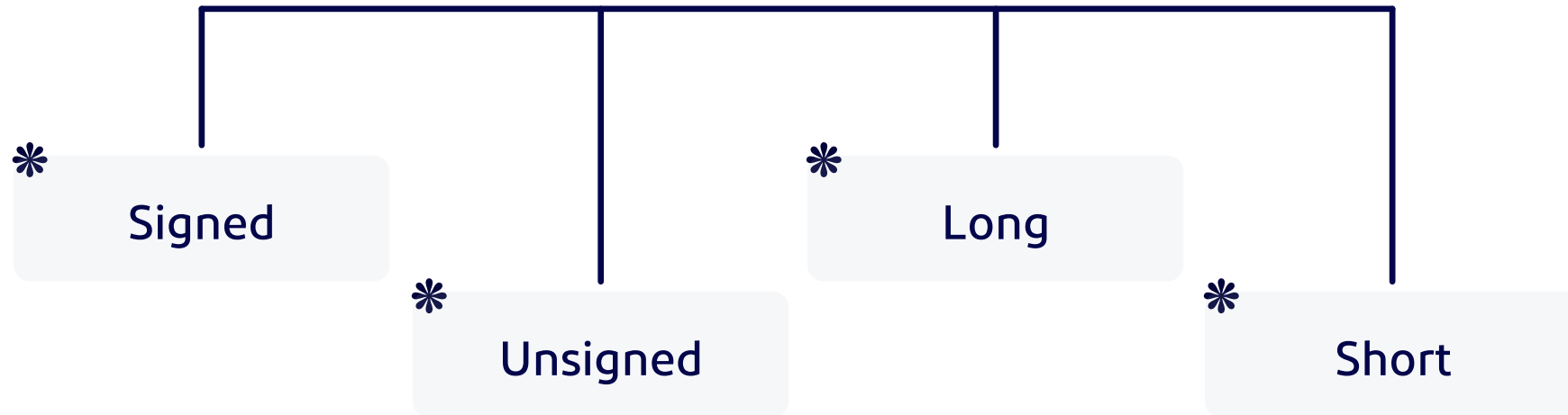
Same as Float. Double variables typically requires 8 byte of memory space.





Data type Modifiers

Data type modifiers are used with the built-in data types to modify the length of data that a particular data type can hold.





Data type Size



Type	Size (in bytes)	Range
short int	2	-32,768 to 32,767
unsigned short int	2	0 to 65,535
int	4	-2,147,483,648 to 2,147,483,647
unsigned int	4	0 to 4,294,967,295
long long int	8	$-(2^{63})$ to $(2^{63})-1$

- The character data type consists of ASCII characters and each character is given a specific value.

Type	Size (in bytes)	Range
char	1	-127 to 127 or 0 to 255
float	4	
double	8	
long double	12	

- Difference between float and double:
 - Size of float(Single precision float data type) is 4 bytes
 - double(Double precision float data type) is 8 bytes
 - Floating point variables has a precision of 6 digits whereas the precision of double is 14 digits.

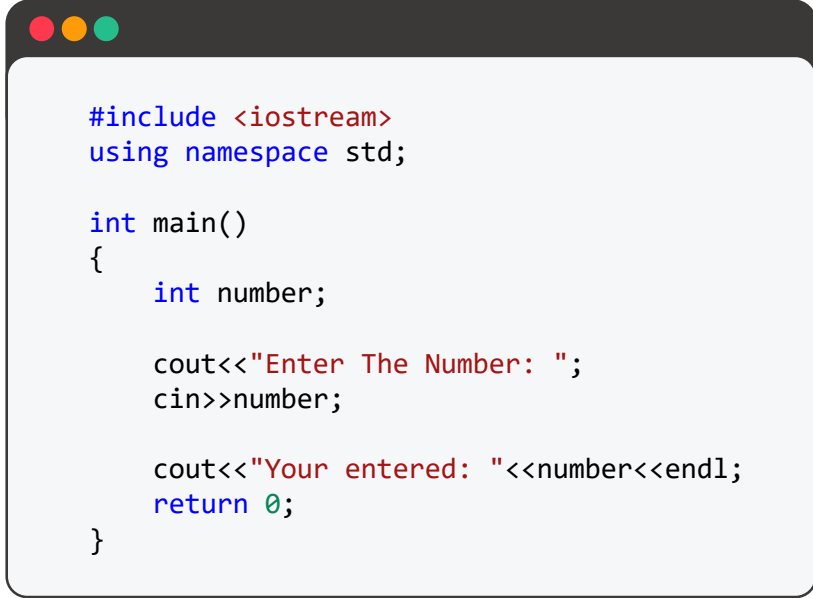




Input and Output

C++ comes with libraries that provide us with many ways for performing input and output.

- I/O Library Header Files:
 - `<iostream>`
 - `iostream` stands for standard input-output stream. This header file contains definitions of objects like `cin`, `cout`.
 - `<iomanip>`
 - This file declares services useful for performing formatted I/O with so-called parameterized stream manipulators, such as `setw` and `setprecision`.
 - `<fstream>`
 - This file declares services for user-controlled file processing.



```
#include <iostream>
using namespace std;

int main()
{
    int number;

    cout<<"Enter The Number: ";
    cin>>number;

    cout<<"Your entered: "<<number<<endl;
    return 0;
}
```



Operators



Escape Character

Escape sequences are used to represent certain special characters within string literals and character literals. The following escape sequences are available:

Escape Character	Description
\'	single quote
\"	double quote
\?	question mark
\\	backslash
\a	audible bell

Escape Character	Description
\b	single quote
\n	new line
\r	carriage return
\t	horizontal tab
\v	vertical tab





Arithmetic Operators

Arithmetic operators are used to perform common mathematical operations.

Operator	Name	Description	Example
+	Addition	Adds together two values.	$x + y$
-	Subtraction	Subtracts one value from another.	$x - y$
*	Multiplication	Multiplies two values.	$x * y$
/	Division	Divides one value by another.	x / y
%	Modulus	Returns the division remainder.	$x \% y$
++	Increment	Increases the value of a variable by 1.	<code>++x</code> or <code>x++</code>
--	Decrement	Decreases the value of a variable by 1.	<code>--x</code> or <code>x--</code>





Comparison Operators

Comparison operators are used to compare two values.

- The return value of a comparison is either true (1) or false (0).

Operator	Name	Example
<code>==</code>	Equal to	<code>x == y</code>
<code>!=</code>	Not equal	<code>x != y</code>
<code>></code>	Greater than	<code>x > y</code>
<code><</code>	Less than	<code>x < y</code>
<code>>=</code>	Greater than or	<code>x >= y</code>
<code><=</code>	Less than or	<code>x <= y</code>





Logical Operators

Logical operators are used to determine the logic between variables or values:

Operator	Name	Description	Example
&&	Addition	Returns true if both statements are true.	<code>x < 5 && x > 1</code>
	Subtraction	Returns true if one of the statements is true.	<code>x < 5 x < 4</code>
!	Multiplication	Reverse the result, returns false if the result is true.	<code>!(x < 5 && x < 10)</code>





Assignment Operators

Assignment operators are used to assign values to variables

Operator	Example	Same As
=	<code>x = 3</code>	<code>x = 3</code>
+=	<code>x += 3</code>	<code>x = x + 3</code>
-=	<code>x -= 3</code>	<code>x = x - 3</code>
*=	<code>x *= 3</code>	<code>x = x * 3</code>
/=	<code>x /= 3</code>	<code>x = x / 3</code>
%=	<code>x %= 3</code>	<code>x = x % 3</code>





Examples:

What is the output?

```
#include <iostream>
using namespace std;

int main()
{
    int x = 2, y = 3, result;

    result = x+ ++y;
    cout<<result<<"\nx = "<<x<<" : y = "<<y<<endl;
    return 0;
}
```

```
#include <iostream>
using namespace std;

int main()
{
    int x = 2, y = 3, result;

    result = x++ +y;
    cout<<result<<"\nx = "<<x<<" : y = "<<y<<endl;
    return 0;
}
```





Examples:

What is the output?

```
#include <iostream>
using namespace std;

int main()
{
    int x = 2, y = 3, z = 12, a = 7, b = 6, result;

    result = x + y * (z / b) - a;
    cout<<result<<endl;
    return 0;
}
```

Try writing a code to swap two numbers.



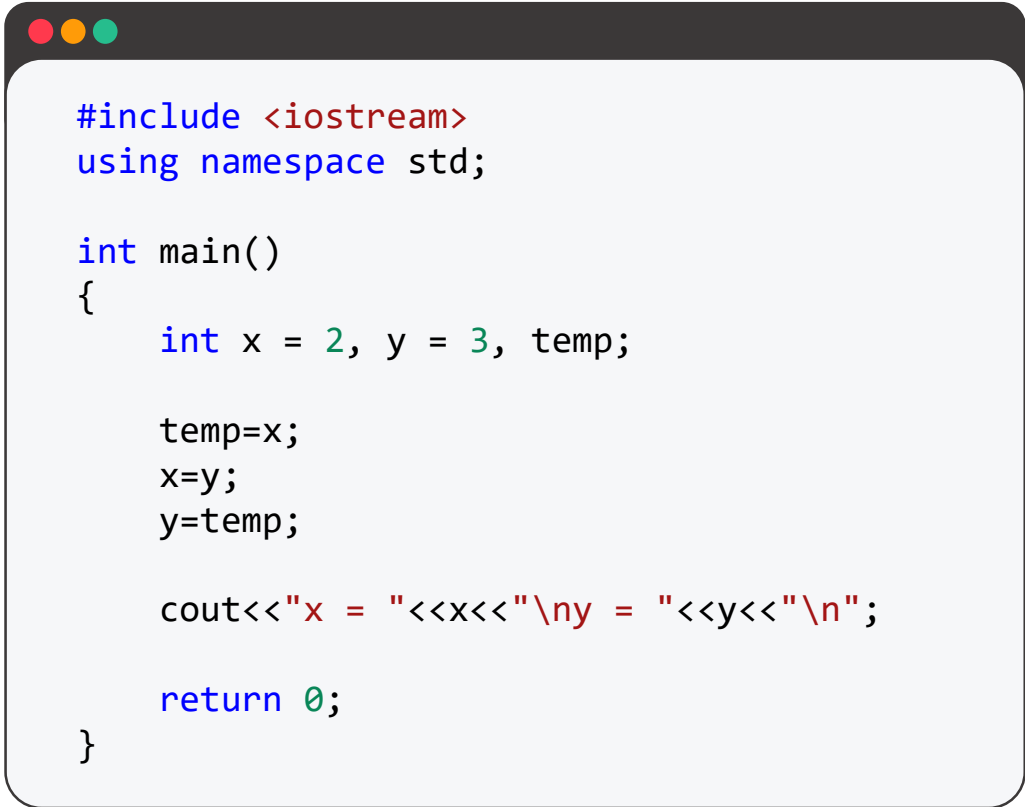


Examples: (Answer)



Swap two numbers:

- The contents of the first variable is copied into the temp variable. Then, the contents of second variable is copied to the first variable.
- Finally, the contents of the temp variable is copied back to the second variable which completes the swapping process.



```
#include <iostream>
using namespace std;

int main()
{
    int x = 2, y = 3, temp;

    temp=x;
    x=y;
    y=temp;

    cout<<"x = "<<x<<"\ny = "<<y<<"\n";

    return 0;
}
```

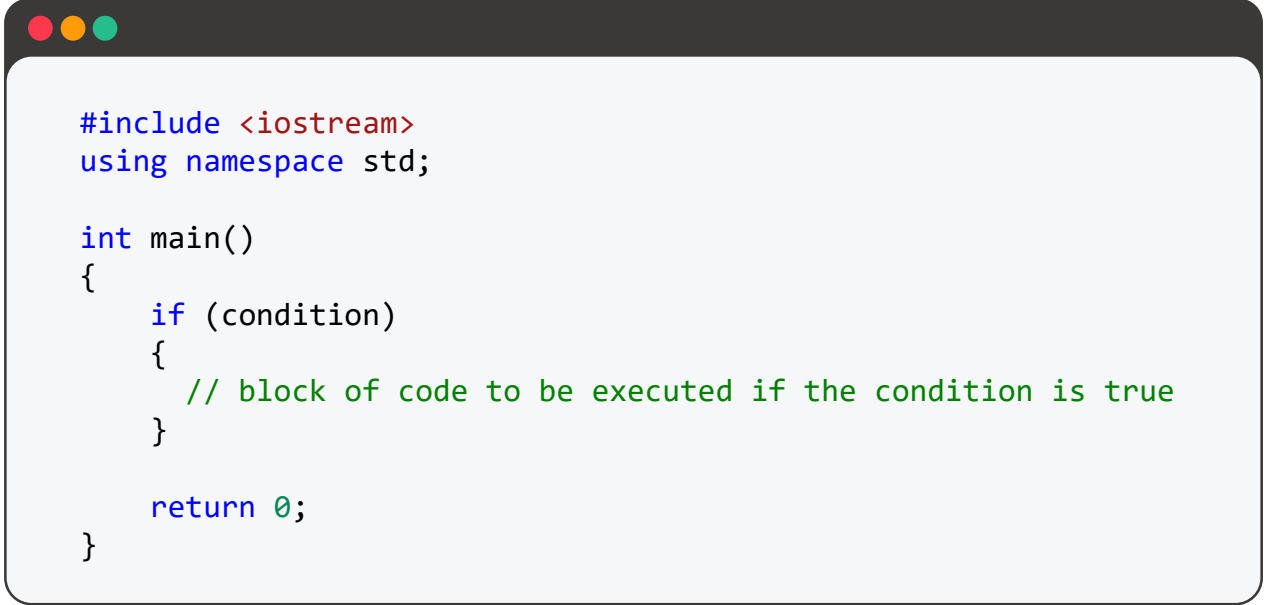


Conditions



If Statement:

If statement is the most simple decision making statement. It is used to decide whether a certain statement or block of statements will be executed or not if a certain condition is true then a block of statement is executed otherwise not.

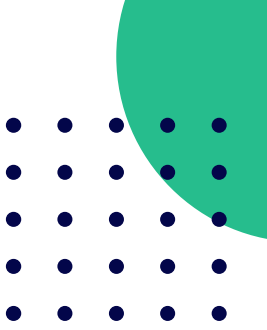
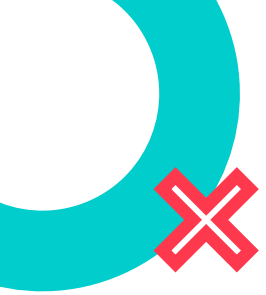


```
#include <iostream>
using namespace std;

int main()
{
    if (condition)
    {
        // block of code to be executed if the condition is true
    }

    return 0;
}
```





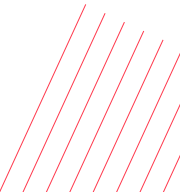
Else Statement:

Use the else statement to specify a block of code to be executed if the condition is false.

```
#include <iostream>
using namespace std;

int main()
{
    if (condition)
    {
        // block of code to be executed if the condition is true
    }
    else
    {
        // block of code to be executed if the condition is false
    }

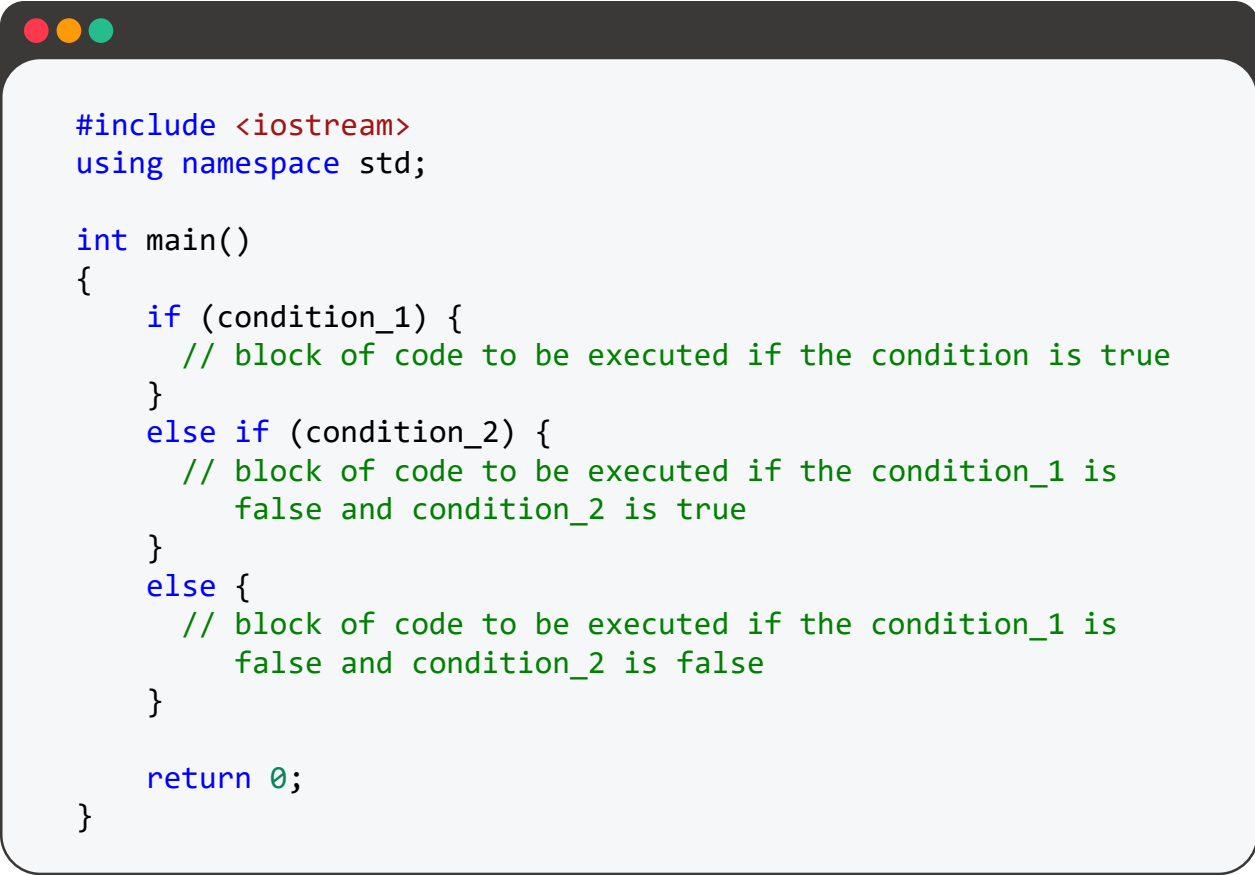
    return 0;
}
```





Else if Statement:

Use the else if statement to specify a new condition if the first condition is false.



```
#include <iostream>
using namespace std;

int main()
{
    if (condition_1) {
        // block of code to be executed if the condition is true
    }
    else if (condition_2) {
        // block of code to be executed if the condition_1 is
        // false and condition_2 is true
    }
    else {
        // block of code to be executed if the condition_1 is
        // false and condition_2 is false
    }

    return 0;
}
```





Examples:

What is the output?

```
#include <iostream>
using namespace std;

int main()
{
    int time = 22;

    if (time < 10) {
        cout << "Good morning.";
    }
    else if (time < 20) {
        cout << "Good day.";
    }
    else {
        cout << "Good evening.";
    }

    return 0;
}
```

+++++



Examples:

write a program to read a number and print “Hello World” if the number is even and less than 20, and print “Hello People” if number odd and less than 40, otherwise print “Without Hello”.

- Input Sample:

14

- Output Sample:

Hello World





Examples:

Answer

```
#include <iostream>
using namespace std;

int main()
{
    int number;
    cin>>number;

    if(number < 20 && number % 2 == 0)
        cout<<"Hello World"<<endl;
    else if(number < 40 && number % 2 != 0)
        cout<<"Hello People"<<endl;
    else
        cout<<"Without Hello"<<endl;

    return 0;
}
```

+++++

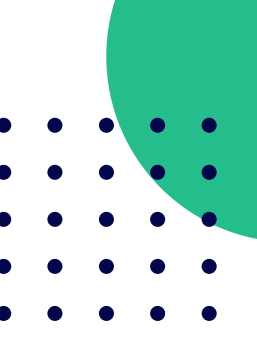


Switch Statements:

Use the switch statement to select one of many code blocks to be executed.

how it works:

- The switch expression is evaluated once.
- The value of the expression is compared with the values of each case.
- If there is a match, the associated block of code is executed.
- The break will stop the execution of more code and case testing inside the block.
- The default keyword specifies some code to run if there is no case match.



```
#include <iostream>
using namespace std;

int main()
{
    switch(expression) {
        case x:
            // code block
            break;
        case y:
            // code block
            break;
        default:
            // code block
    }

    return 0;
}
```





Examples:

What is the output?



```
#include <iostream>
using namespace std;

int main()
{
    int day = 4;

    switch (day) {
        case 6:
            cout << "Today is Saturday";
            break;
        case 7:
            cout << "Today is Sunday";
            break;
        default:
            cout << "Looking forward to the Weekend";
    }

    return 0;
}
```





References

Videos

■ DataTypes 1.

- <https://www.youtube.com/watch?v=RRouhxZJKak>

■ DataTypes 2.

- <https://www.youtube.com/watch?v=7OynnsDBoHk>

■ Conditation.

- <https://www.youtube.com/watch?v=xmjB7u7mHWE>

■ Logical Operators.

- <https://www.youtube.com/watch?v=qmHE9QISuVQ>

■ Division & Modulus.

- https://www.youtube.com/watch?v=jjVaDL_dePk





THANKS!

Any Questions?!