Introduction to C++

C++ is a high-level, object-oriented programming language used for developing applications, games, and system software.

Peprocessor directives :-Preprocessor directives in C++ begin with '#' and are used to give instructions to the compiler before compiling the code, such as including header files or defining constants.

Key words in C++:- are special words that have predefined meanings and are reserved for specific purposes. Here are some examples of keywords in C++: "int" is a keyword used to declare integer variables.

For example

1. int x = 5; "float" is a keyword used to declare floating-point variables.
2. float y = 3.14; "if" is a keyword used for conditional statements.
3. if (x > y) { ... } "while" is a keyword used for loops.
4. while (x < 10) { ... } "class" is a keyword used to define a class.
5. class MyClass { ... }; "namespace" is a keyword used to define a namespace.namespace MyNamespace { ... };

C++Identifiers:- In C++, an identifier is a name given to a variable, function, class, or other user-defined entity. An identifier can consist of letters, digits, and underscores, and must begin with a letter or underscore. Identifiers are case sensitive and should be chosen to be descriptive and meaningful. Examples of valid identifiers in C++ include "myVar", "some_function", and "MyClass".

Examples

Variables: int myAge, double mySalary, bool isFinishedFunctions: void printMessage(), int calculateSum(int a, int b), double calculateArea(double radius)Classes: class Car, class Person, class StudentConstants: const int MAX_SIZE = 100, const double PI = 3.14159

C++ comments :- In C++, comments are used to add explanatory text that is ignored by the compiler.

There are two types of comments in C++:

1. Single-line comments: Begin with two forward slashes (//) and continue to the end of the line. Example: // This is a single-line comment.
2. Multi-line comments: Begin with a forward slash followed by an asterisk (/) and end with an asterisk followed by a forward slash (/).
   Example: /*This is a multi-line comment. It can span multiple lines.*/ C++

VariablesIn C++:- are used to store data in memory. Before using a variable, it must be declared with a data type and an identifier. Here are some examples of how to declare and create variables in C++:int age; // variable declarationage = 30;

// variable initializationint x, y, z;x =9;

Assigning values to variables Basic data types and their rangesC++ supports several basic data types, which are used to represent different kinds of values in memory. Here is a list of the basic data types in C++: char: 1 byte, range from -128 to 127 or 0 to 255 (if unsigned) int: 4 bytes, range from -2,147,483,648 to 2,147,483,647 short: 2 bytes, range from -32,768 to 32,767 long: 4 bytes, range from -2,147,483,648 to 2,147,483,647 long long: 8 bytes, range from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807float: 4 bytes, range from approximately 1.2E-38 to 3.4E+38 with 6 decimal places of precision double: 8 bytes, range from approximately 2.3E-308 to 1.7E+308 with 15 decimal places of precision bool: 1 byte, can store true (1) or false (0) Note that the actual range and size of each data type may depend on the compiler and operating system being used. Additionally, there are other data types in C++ that are less commonly used or are specific to certain applications or libraries.

C++ operators:-are symbols that are used to perform various operations on operands such as variables, constants, and expressions. C++ supports a wide range of operators, including: Arithmetic operators: Used to perform arithmetic operations on operands.

Examples: + (addition), - (subtraction), * (multiplication), / (division), and % (modulus).
Assignment operators: Used to assign a value to a variable.

Examples: = (simple assignment), += (add and assign), -= (subtract and assign), *= (multiply and assign), /= (divide and assign), and %= (modulus and assign).

Comparison operators: Used to compare two values and return a Boolean result.

Examples: == (equality), != (inequality), < (less than), > (greater than), <= (less than or equal to), and >= (greater than or equal to).

Logical operators: Used to combine Boolean expressions and return a Boolean result. Examples: && (logical AND), || (logical OR), and ! (logical NOT). Bitwise operators: Used to perform bitwise operations on binary values.

Examples: & (bitwise AND), | (bitwise OR), ^ (bitwise XOR), ~ (bitwise NOT), << (left shift), and >> (right shift).

Ternary operator: Used to provide a shortcut for an if-else statement.

Example: (condition) ? value_if_true : value_if_false.

Increment and decrement operators:-are used to increment or decrement the value of a variable by 1 in C++.The increment operator (++) adds 1 to the value of a variable, while the decrement operator (--) subtracts 1 from the value of a variable. These operators can be used with both integer and floating-point variables.

Here are some examples of how to use these operators in C++:- int num = 5;num++; // Increment the value of num by 1cout << "num after increment: " << num << endl; int count = 10;count--; // Decrement the value of count by 1cout << "count after decrement: " << count << endl; float value = 3.14;value++; // Increment the value of value by 1cout << "value after increment: " << value << endl; //num after increment: 6//count after decrement: 9//value after increment:

Simple data type conversion: Type conversion in C++ refers to the process of converting one data type to another. C++ provides two types of type conversion: implicit type conversion (also called type coercion) and explicit type conversion (also called type casting). Implicit type conversion is performed automatically by the compiler when necessary, without the need for any special syntax.

For example, when you assign a value of a smaller data type to a variable of a larger data type, the value is automatically converted to the larger type.

Here is an example of implicit type conversion: int num1 = 5;double num2 = num1; // Implicitly convert int to double Explicit type conversion, on the other hand, requires the use of a special syntax to convert one data type to another. This can be done using the C++ cast operator or the functional notation.

Here are some examples of explicit type conversion: double num1 = 3.14;int num2 = static_cast<int>(num1); // Cast double to int using static cast int num3 = 10;double num4 = (double) num3; // Cast int to double using functional notation.

Exercise 1

```
// This program calculates the product of three integers

 #include <iostream>

using namespace std;

int main()

{

int x, y, z, result;    // declare variable

cout << "Please enter three integers: "; //prompt the usercin >> x >> y >> z; //read data from the keyboard

result = x * y * z;

cout << "The product is " << result << endl;

 return 0;

}
```

Exercise 2

```
/*Write a program that accepts two integers and display the sum, difference, product and division
of the two numbers. The program should also state the greater and smaller number. *

/#include <iostream>

using namespace std;

int main()

{

int num1, num2, sum, difference, product;

float division;

cout << "Enter two integers: ";

cin >> num1 >> num2;

 sum = num1 + num2;

difference = num1 - num2;

product = num1 * num2;

division = (float) num1 / num2;    // Convert num1 to float to avoid integer division

cout << "Sum: " << sum << endl;

cout << "Difference: " << difference << endl;

cout << "Product: " << product << endl;

cout << "Division: " << division << endl;

 if (num1 > num2) {cout << num1 << " is greater than " << num2 << endl;

}

else if (num2 > num1) {cout << num2 << " is greater than " << num1 << endl;

}
```

```
 else
{
cout << "The numbers are equal" << endl;
}
 return 0;
}
```

Exercise 3

```
/* Write a program that calculate and display the circumference of a circle. (C = 2∏r )*/
#include <iostream>
using namespace std;
 int main()
{
const float PI = 3.14159; // Declare and initialize the constant PI
float radius, circumference;
 cout << "Enter the radius of the circle: ";
cin >> radius; circumference = 2 * PI * radius; // Compute the circumference
cout << "The circumference of the circle is " << circumference << endl;
 return 0;
}
```

```
Exercise 4 /*4.

Write a program to solve a quadratic equation.

 Hint: y = x2 + bx

Note:include math.h to use square root function(sqrt(double)) */

 #include <iostream>

#include <math.h>

using namespace std;

int main ()

{

double discriminanate,root1, root2;

cout<<"enter cooficients a, b and c:";

cin>>a>>b>>c;

// calculate the discriminant

 Discriminate=b*b-4*ac;

//find the roots the the quadratic equation

If(discriminate>0)

{
```

Chapter Three

Exersice 1

```
//1

#include <iostream>

Using namespace std;

Int main()

{

Int sum =0;

For (int i=0;i<100;i++)

{

Sum=sum+i;

}

Cout<<”the sum of numbers from 1 to 100 is :sum<<endl;

Return 0;

}
```

#2

```
#include <iostream>

Using namespase std;

Int main()

{

For (Int i=0;

i<=100;

i++)

{

If(i%2==0&&i%3==0&&i%5==0)

{

Cout<<i<<"***"<<endl;

}

Return 0;

}
```

#3

```
#include <iostream>
Using namespase std;
Int main()
{
Int num,fact=1;
//using for loop
Cout<<" enter a number to find its factorial using for loop:";
Cin>>num;
For( int i-=1;1<=num;i++)
{
Num<<fact*=i;
"1
Cout<<"factorial of <<"1
//Using for  loop is
<<fact<<endl;
// using while loop
Fact=1;
11
Cout<<"enter the number to find its factorial using while loop:";
Cin>>num;
Intj=1;
While(j<=num)
{
Fact*=j;
J++;
}
“1
Cout<<”factorial of <<0 num <<using while loop is:<<fact<<endl;
//using do while loop fact =1;
Cout<<”enter a number to find its factorial using do while loop:”;
Num
Cin>> num;
Int k=1;
Do{
Fact*=k;
K++;
}
While (k<=num);
Cout<<”factorial of “num<<using do while loop is :’’<< fact <<endl;
Return0;
}
```

#4

```
#include <iostream>

Using namespase std;

Int main()

{

Int count=1,sum=0;

Float average ;

While (count<=10)

{

Sum+=count,count++;}

Average sum /10.0;

Cout<<” the average is<<average<<enld;

Return0;

}
```