For Secondary Classes

Answer Key

Modular C++ (Ver. 2.0)

1. Introduction to C++

Exercise

A. 1. a. 2. a. 3. d. 4. b. 5. a. 6. b.

B. 1. T 2. T 3. T 4. F 5. T

C. 1. curly brace 2. Run 3. pre-processor 4. semicolon 5. AT&T Bell Labs

D. 1. International Organization for Standardization

2. We need to compile a program to check whether it contains any warning or error.

3. The #include<iostream.h> statement is used to include the input/output stream header file in a C++ program.

E. 1. Four key features of the C++ language are:

i. Object-Oriented Programming: C++ focuses on objects rather than just step-by-step instructions. Objects are real-world entities that contain both data (variables) and functions (methods) that operate on the data. This makes programming more structured, reusable, and easier to manage.

Example: If we create a program for a car, we can make a Car object that contains properties (like colour, speed) and functions (like start, stop).

ii. Platform Independent: C++ is a platform-independent language, meaning that a program written in C++ can run on different operating systems with little or no modification. If a program is written and compiled on Windows, it can also run on Linux, MacOS, or other operating systems, as long as a compatible C++ compiler is available.

iii. Simple and Popular: C++ is a simple and popular programming language because it is easy to learn and widely used in various fields like gaming, software development, and robotics. Its syntax is very easy to understand.

iv. High-Level Language: C++ is a high-level programming language, which means it is easy to read and understand, just like English. Unlike low-level languages that use numbers and symbols, C++ uses simple words and rules, making it easier for beginners to learn.

2. Compiling a program helps detect warnings or errors before running the program.

Running a program shows how the program behaves after successful compilation.

IN THE LAB

Do it yourself.

2. Getting Started With C++

EXERCISE

A. 1. a. 2. b. 3. a. 4. d. 5. a. 6. d.
7. d.

B. 1. T 2. T 3. T 4. F 5. F

C. 1. Floating-Point 2. Initializing 3. strongly-typed 4. memory
5. variable

D. 1. Primitive data types are built-in or default data types. User can directly use these data types to declare variables.

2. A data type defines the type of value that a variable can store. Data types in C++ are int, char, float etc.

3. A character set refers to the letters, digits and special symbols that can be used to write programs in the C++ language.

4. data_type variable_name;

E. 1. Following are the naming conventions used to declare a variable in C++:

i. A variable name must start with a letter (a–z, A–Z) or an underscore (_).

ii. A variable name cannot start with a digit (0-9).

iii. Keywords cannot be used as variable names.

iv. A variable can only contain alpha-numeric characters and underscore (A–Z both capital as well as small) and (0–9) numbers.

2. C++ is a strongly-typed language, which means you need to declare a variable before using it. You also need to specify the data type of the variable, so that the compiler understands how the variable is interpreted. When a variable is declared, the computer reserves memory space for it.

3. Delimiters: C++ allows some special characters in its coding called delimiters or punctuators. These are: { } ( ) , ; =

Operators: Operators are special symbols in C++ that are used to perform calculations. They are used along with operands or values to get the desired result.

IN THE LAB

Do it yourself.

Periodic Assessment-1

(Based on chapters 1 & 2)

A. 1. Menu Bar: It contains different menus like File, Edit, Search, Run, Compile, Debug, Project, Options, Window and Help. These menus help user to do various tasks like create a new file, save a program, compile a program and run a program.

2. Platform Independent: C++ is a platform-independent language, meaning that a program written in C++ can run on different operating systems with little or no modification. If a program is written and compiled on Windows, it can also run on Linux, MacOS, or other operating systems, as long as a compatible C++ compiler is available.

3. main(): main() function serves as the entry point of the program. Every C++ program must have a main() function, as execution starts from this function.

4. getch(): The getch() function is used to pause the program execution until a key is pressed. This ensures that the output window remains open until the user interacts with the keyboard.

B.

```
#include<iostream.h>
#include<conio.h>
void main()
{
    cout << "Welcome to Orange Education";
    getch();
}
```

C. 1. c. 2. d. 3. b. 4. e. 5. a.

3. Input and Output in C++

Exercise

A. 1. a. 2. a. 3. d. 4. a. 5. c.

B. 1. T 2. T 3. F 4. T

C. 1. break 2. cin 3. ostream 4. comments

D. 1. C++ supports two types of comments: Single-line comment and Multiple-line comment.

2. \\ is used to represent a backslash in a string or escape sequence.

   // is used to indicate a single-line comment in C++. Everything after // on that line is treated as a comment.

3. Escape sequence characters are used to represent certain special characters that cannot be directly printed on the output screen.

E. 1. The cout object is used to display output on a computer screen (monitor). It is part of the ostream class and is also known as the Standard Output Stream. The cout object works with the stream insertion operator (<<) to show messages or values. For example:

cout<<"Output on the screen";

2. cout: The cout object is used to display output on the computer screen (monitor). It is part of the ostream class and is also known as the Standard Output Stream. The cout object works with the stream insertion operator (<<) to display messages or values.

cin: The cin object is used to take input from the user through the keyboard. It is part of the iostream library and works with the extraction operator (>>), which is two greater-than signs. The extraction operator is followed by the variable where the input value will be stored.

F. 1. Hello Touchpad

2. Enter your age: 25

25

IN THE LAB

Do it yourself.

4. Operators in C++

EXERCISE

A. 1. d. 2. c. 3. b. 4. d. 5. a.

B. 1. T 2. T 3. T 4. T 5. F

C. 1. Modulus 2. == 3. AND 4. Assignment 5. Increment

D. 1. / (Division Operator): The division operator divides the numerator by the denominator and returns the quotient.

% (Modulus Operator): The modulus operator divides the numerator by the denominator and returns the remainder of the division.

2. Operator Precedence determines the order in which operators are executed in an expression.
3. Type casting refers to the process of converting a value from one data type to another.
4. Relational operators are used to compare the value of the two operands and return True or False accordingly.
5. Binary operators are operators that work on two operands to perform operations. These operators require two values to perform the operation and return a result.

E. 1.

Operator	Name	Description
==	Equal to	It checks if the values of two operands are equal and returns True if both are equal otherwise false.
!=	Not equal to	It checks if the values of two operands are not equal and returns false if both are equal.
>	Greater than	It checks if the value of left operand is greater than the value of right operand.
<	Less than	It checks if the value of left operand is less than the value of right operand.
>=	Greater than or equal to	It checks if the value of left operand is greater than or equal to the value of right operand.
<=	Less than or equal to	It checks if the value of left operand is less than or equal to the value of right operand.

2. Prefix (++a or --a): The operator is placed before the operand.

   Postfix (a++ or a--): The operator is placed after the operand.

   The difference between prefix and postfix (++ and – –) operators is that, in case of prefix operators the value of the variable is incremented/decremented first and then the expression is evaluated. In case of postfix operators, the expression is evaluated first and then the value of the variable is incremented/decremented. Consider the preceding example:

   int a = 10, b = 0;

   b = ++a; //Prefix increment: a is incremented first, then assigned to b.

   b = a++; // Postfix increment: a is assigned to b first, then incremented.

   b = --a; // Prefix decrement: a is decremented first, then assigned to b.

   b = a--; // Postfix decrement: a is assigned to b first, then decremented.

3. Implicit type casting, also known as automatic type conversion, is performed automatically by the C++ compiler when a smaller data type is converted into a larger one. For example, a char data type can be automatically converted into an int by the compiler.

   ```
   char i = 'A';
   int num;
   num = i;
   ```

   Explicit type casting refers to manually converting one data type into another using a type cast operator. A type cast operator consists of a data type enclosed in parentheses. For example:

   int i = 10;

   float f;

   f = (float) i;

4. *= (Multiplication Assignment Operator): This operator multiplies the right operand with the left operand and assigns the result to the left operand. x*=3 is equivalent to x=x*3.

   /= (Division Assignment Operator): This operator divides the left operand by the right operand and assigns the result to the left operand. x/=3 is equivalent to x=x/3.

F. 1. 77

2. 4
3. Grade is: A
4. value of x is: 7

IN THE LAB

Do it yourself.

5. Conditional Statements

EXERCISE

A. 1. a. 2. b. 3. a. 4. c. 5. a.

B. 1. T 2. F 3. F 4. T 5. T

C. 1. if 2. true 3. false 4. else 5. break

D. 1. Conditional statements are used to test the conditions and decide the flow of a program on the basis of result of these conditions.

2. The if...else statement is used in programming to make decisions. If the condition is True, the program runs the if block. If it is False, the program runs the else block instead.
3. Yes, the ternary operator (? :) can be used to replace a simple if...else statement.
4. The syntax of nested statement is:

```
if(condition1)
{
    // Statements for condition1
    if(condition2)
    {
        // Statements for condition2
       // (executed only if condition1 and condition2 are both true)
    }
    else
```

```
    {
        // Statements executed if condition2 is false
    }
}
```

E. 1. The if statement is the simplest conditional statement. It selects and executes the statement(s) based on a given condition. If a particular condition or expression evaluates to True, the statements inside the if block will be executed. Otherwise, the control of execution is passed to the next statement after the if block. The syntax of the if statement is given below:

```
if(condition)
{
    statement(s);
}
```

If the value of the condition is nonzero (True), the body of the if statement gets executed.

2. The switch statement is a control structure that provides a more efficient way to replace multiple if...else...if statements when testing a single expression against multiple values. The syntax of the switch statement is:

```
switch(expression)
{
case value1:
// Statements;
    break;
case value2:
// Statements;
    break;
case value3:
// Statements;
    break;
default:
// Statements for unmatched cases;
}
```

This syntax shows how the switch statement checks for a match among multiple cases and uses break to stop execution after a match is found. If no case matches, the default case runs.

if Statement	switch Statement
Can test any type of expression: integer, float, logical, relational, etc.	Can only test integer or character expressions.
Encloses multiple statements inside curly braces { }.	Each case is written separately without curly braces.
Does not allow break (using it will cause an error).	Requires break after each case to prevent fall-through.
Example: if (n == 0) { ... } — no break used.	Example: case 1: statement; break; — break used to exit the case.

3. The default clause in a switch statement is used to define a set of statements that will execute when none of the case values match the switch expression.

4.
```
#include<iostream.h>
#include<conio.h>
void main()
{
    clrscr();
    char code;
    cin >> code;
    switch(code)
    {
        case 'A':
            cout << "Accountant";
            break;

        case 'C':
        case 'G':
            cout << "Grade IV";
            break;
        case 'F':
            cout << "Financial Advisor";
            break;
        default:
            cout << "Invalid code";
    }

    getch();
}
```

Note: Combined case 'C': and case 'G': for logical OR using fall-through.

5.
```
#include<iostream.h>
#include<conio.h>
void main()
```

```
{
    clrscr();
    char draw;
    cout << "Enter your choice (R/C/E/P): ";
    cin >> draw;
    switch (draw) {
        case 'R':
            cout << "Draw rectangle";
            break;

        case 'C':
            cout << "Draw circle";
            break;

        case 'E':
            cout << "Draw ellipse";
            break;

        case 'P':
            cout << "Draw polygon";
            break;
        default:
            cout << "Wrong choice";
    }
    getch();
}
```

6.
```
#include<iostream.h>
#include<conio.h>
void main()
{
    clrscr();
    int time = 8;

    if (time < 12)
    {
        cout << "Good morning.";
    }
    else if (time > 20)
    {
        cout << "Good night.";
    }
    else
    {
        cout << "Good evening.";
```

```
    }

    getch();
}
```

Periodic Assessment-2

(Based on chapters 3 to 5)

A.
1. It checks if the values of two operands are not equal and returns false if both are equal. Example: (x=8 and y=6) x!=y
2. It returns true, if both operands are true. Example: (x<5)&&(x<10)
3. It takes modulus of two operands and assigns the result to left operand. Example: x%=3
4. The operator is placed before the operand. Example: b = ++a.

B. Good morning.

C.

```
#include<iostream.h>
#include<conio.h>
void main()
{
    clrscr();
    int A = 25;
    int B = 10;
    int result;
    result = A+3*(B-4)-10/2;
    cout << "The result of the expression is: " << result;
    getch();
}
```

Test Sheet-1

(Based on chapters 1 to 5)

Section A

A.	1.	a.	2.	d.	3.	d.	4.	a.	5.	a.	6. a.
	7.	d.	8.	a.	9.	a.	10.	b.			
B.	1.	semicolon	2.	memory	3.	==	4.	ostream	5.	curly brace	6. break
C.	1.	T	2.	F	3.	T	4.	T	5.	F	6. T

Section B

A. 1. The #include<iostream.h> statement is used to include the input/output stream header file in a C++ program.

2. data_type variable_name;

3. / (Division Operator): The division operator divides the numerator by the denominator and returns the quotient.

   % (Modulus Operator): The modulus operator divides the numerator by the denominator and returns the remainder of the division.

4. \\ is used to represent a backslash in a string or escape sequence.

   // is used to indicate a single-line comment in C++. Everything after // on that line is treated as a comment.

5. The if...else statement is used in programming to make decisions. If the condition is True, the program runs the if block. If it is False, the program runs the else block instead.

6. Yes, the ternary operator (? :) can be used to replace a simple if...else statement.

B. 1. Following are the naming conventions used to declare a variable in C++:

   i. A variable name must start with a letter (a–z, A–Z) or an underscore (_).

   ii. A variable name cannot start with a digit (0-9).

   iii. Keywords cannot be used as variable names.

   iv. A variable can only contain alpha-numeric characters and underscore (A–Z both capital as well as small) and (0–9) numbers.

2. *= (Multiplication Assignment Operator): This operator multiplies the right operand with the left operand and assigns the result to the left operand. x*=3 is equivalent to x=x*3.

   /= (Division Assignment Operator): This operator divides the left operand by the right operand and assigns the result to the left operand. x/=3 is equivalent to x=x/3.

3. Compiling a program helps detect warnings or errors before running the program.

   Running a program shows how the program behaves after successful compilation.

4. cout: The cout object is used to display output on the computer screen (monitor). It is part of the ostream class and is also known as the Standard Output Stream. The cout object works with the stream insertion operator (<<) to display messages or values.

   cin: The cin object is used to take input from the user through the keyboard. It is part of the iostream library and works with the extraction operator (>>), which is two greater-than signs. The extraction operator is followed by the variable where the input value will be stored.

5. ```
   #include <iostream.h>
   #include <conio.h>
   ```
   ```

```
void main()
{
    clrscr();
    int num;
    cout << "Enter a number (1 to 7): ";
    cin >> num;

    switch(num)
    {
        case 1:
            cout << "Sunday";
            break;
        case 2:
            cout << "Monday";
            break;
        case 3:
            cout << "Tuesday";
            break;
        case 4:
            cout << "Wednesday";
            break;
        case 5:
            cout << "Thursday";
            break;
        case 6:
            cout << "Friday";
            break;
        case 7:
            cout << "Saturday";
            break;
        default:
            cout << "Invalid input! Please enter a number between 1
            and 7.";
    }
    getch();
}
```

6. The default clause in a switch statement is used to define a set of statements that will execute when none of the case values match the switch expression.

6. Loops

Exercise

A. 1. d. 2. b. 3. a. 4. a. 5. c.

B. 1. F 2. F 3. F 4. T 5. T

C. 1. three 2. do-while 3. Iteration 4. nested 5. infinite

D. 1. Loops help us avoid writing the same code again and again, making programs shorter and more efficient.

2. C++ offers four jump statements — break, continue, return and goto.

3. C++ allows you to declare and initialise a variable inside the for loop.

   Syntax: for (int i = 0; i < 10; i++)

4. This statement is used to change the normal flow of program execution by transferring control to a specific location in the program.

E. 1.

break	continue
Exits the loop immediately	Skips the current iteration of the loop
Transfers control to the statement after the loop	Transfers control to the next iteration of the loop
Terminates the loop when a specific condition is met	Does not terminate the loop, just skips the current iteration
Only exits the innermost loop if used in a nested loop	Skips the current iteration in the innermost loop
Used when you want to exit the loop early	Used when you want to skip an iteration without exiting the loop

2.

while	do-while
The while loop is a pre-test (or top-test) loop, meaning the condition is checked first before entering the loop body.	The do-while loop is a post-test (or bottom-test) loop, meaning the loop body executes first, and then the condition is checked.
In a while loop, the minimum number of iterations is zero because the condition is tested before executing the loop body.	In a do-while loop, the minimum number of iterations is one, as the loop body executes at least once, even if the condition is false.

3. Yes, a while loop can contain another while loop, also known as a nested while loop.

   Syntax of a Nested While Loop:

```
while (condition1)
```

```
{
    // Outer loop body
    while (condition2)
    {
        // Inner loop body
    }
    // Outer loop continues...
}
```

F. 1. 1
2
3
4
6
7
8

2. 124

3. 0 1 3 6 10 15 21 28 36 45 55

4. * * *
* * *
* * *
* * *
* * *

IN THE LAB

Do it yourself.

7. OOP Concepts

EXERCISE

A. 1. d. 2. a. 3. a. 4. b 5. d. 6. b.

B. 1. T 2. T 3. F 4. T

C. 1. Encapsulation 2. Derived class 3. Properties 4. Methods

D. 1. Object-Oriented Programming (OOP) is a programming paradigm that focuses on objects rather than routines or functions.

2. Encapsulation refers to a process of binding data and function together into a single unit , similar to a capsule.
3. Inheritance is one of the most important features of object-oriented programming that allows a class to acquire or inherits all the properties and behaviours of its parent class.
4. The process of defining multiple functions with the same name but with different numbers, types, or sequences of parameters.

E. 1. Function Overloading: The process of defining multiple functions with the same name but with different numbers, types, or sequences of parameters.

Function Overriding: The process of redefining a function with the same name, return type, and argument list in both the subclass and the superclass.

2. A class can be defined as a user defined blueprint or prototype used to create objects. It contains objects that have similar properties and behaviour. For example, an animal class has different types of animals.

An object is a real-world entity like car, dog, pencil and computer. All real-world entities have properties and behaviour. Properties represent the physical appearance and qualities of an object. On the other hand, behaviour represents the functions or actions that an object can perform. For example, a dog has some properties like colour, breed, hairs, ear, etc. and behaviour like barking, eating, sleeping, etc.

3. The class which inherits the members of another class is called derived class or sub class. On the other hand, the class whose members are inherited is called base class or super class.

IN THE LAB

Do it yourself.

Periodic Assessment-3

(Based on chapters 6 & 7)

A. 1 2 3 4 5 6 7 8 9 10

B.
```
#include<iostream.h>
#include<conio.h>
void main()
{
    clrscr();
    int i = 1;
    while(i <= 5)
   {
```

```
            int j = 1;
            while(j <= i)
            {
                cout << "* ";
                j++;
            }
            cout << endl;  // Move to the next line after each row
            i++;
        }
        getch();
}
```

C. 1. The term polymorphism is derived from two Greek words: "poly" meaning many and "morph" meaning forms. In Object-Oriented Programming (OOP), polymorphism is a feature that allows an object to behave differently in different situations.

2. An object is a real-world entity like car, dog, pencil and computer. All real-world entities have properties and behaviour. Properties represent the physical appearance and qualities of an object. On the other hand, behaviour represents the functions or actions that an object can perform.

3. Encapsulation refers to a process of binding data and function together into a single unit, similar to a capsule. It is used to restrict access to private data members from outside the class, ensuring data security and integrity.

8. Functions in C++

Exercise

A. 1. a. 2. a. 3. b. 4. a. 5. a. 6. a.

B. 1. T 2. T 3. T 4. F 5. F

C. 1. scope resolution 2. reference 3. arguments
4. definition 5. functions

D. 1. A function is a block of organised and reusable code used to perform a particular task.

2. Declaring a function is called function prototyping.

3. In a function definition, void is used to specify that the function does not return any value.

4. The return statement is a jump statement used to exit from a function and pass control back to the calling function.

5. The variables that are declared outside of all the functions are called global variables.

E. 1. Declaring a function: Before using a function, you need to declare it. Declaring a function is called function prototyping. A function prototype tells the compiler about the data type and parameters of the function. The syntax to declare a function in C++ is:

```
return_type function_name([parameter 1, parameter 2, …]);
```

Where, the return_type specifies the data type of the value returned by the function. If no value is returned, the function should be declared with void. The function_name is the assigned name used to define and call the function later. Parameters are placed inside parentheses () and act as placeholders for input values. The declaration must end with a semicolon (;).

int display(int);

In the above example, the display function is declared with int data type and and an int parameter.

Defining a function: After declaring a function, you need to define it. Defining a function means to give the body of the function. The definition of a function contains the functionality of the function in within curly braces{ }. Let us consider the preceding example of display function:

```
int display(int age)
{
    cout<<"Your age is: " << age <<" years.";
    return 0;
}
```

2. Local Variables: The variables which are declared in the body of a function or block are called local variables for that function. You can use these variables only inside the function in which they are declared. If you try to use the local variables outside their scope, C++ compiler will show an error message.

   Global Variables: The variables that are declared outside of all the functions are called global variables. Global variables can be used by any function in the program. Generally, global variables are declared just below the preprocessor directive.

3. The parameters passed at the time of declaring a function are called formal parameters. On the other hand, the parameters passed at the time of calling a function are called actual arguments.

4. Do it yourself.

F. 1.
```
int div(int a, int b)
{
    a = a * b;
    return(a / b);
}
```

2.
```
int sum(int i, int j)
```

```
    {
        return i + j;
    }
3. void disp() {
        cout << "program";
    }
```

IN THE LAB

Do it yourself.

9. Header Files and Library Functions

EXERCISE

A. 1. a. 2. a. 3. a. 4. b. 5. a.

B. 1. T 2. T 3. T 4. F 5. F

C. 1. header 2. log() 3. floor() 4. isdigit() 5. extension

D. 1. A header file is a collection of predefined functions that help improve a program's productivity and save development time.

2. Three functions provided by math.h header file are: ceil(a), fabs(a), floor(a).

3. To use any library function in your program, respected header file should be included at the beginning of your program code. You can include a header file in your program by using the preprocessor directive #include followed by name of the header file in between the angular brackets.

E. 1.

Function Name	Use	Example	Output
isalpha()	It returns true if the character c is an uppercase or lowercase letter.	isalpha('B')	True
Isalnum()	It returns true if the character c is a digit (0-9) or a letter (either uppercase or lowercase). Otherwise, it returns false.	isalnum('9')	True

2.

Function Name	Use	Example	Output
ceil()	It returns a after rounding up it to the nearest integer.	Ceil(2.3)	3
floor()	It returns a after rounding down it to the nearest integer.	floor(-25.86)	25.86

IN THE LAB

Do it yourself.

10. App Development

EXERCISE

A. 1. c. 2. b. 3. c. 4. d. 5. a.

B. 1. T 2. F 3. F 4. F 5. T

C. 1. Web browser 2. Gmail 3. Component blocks 4. Emulator
5. E-commerce

D. 1. Google Play Store and Apple App Store.

2. The commonly used Android Emulators are Android Studio, LD Player.

3. Examples of e-commerce apps are Amazon and Blinkit.

4. Yes, App Inventor is a free app development tool.

5. An app is a type of software that is designed specifically for portable smart devices like tablets and smartphones.

E. 1. While developing an app, we should keep some points in mind:

- The idea of developing an app should be unique.
- The app we are going to develop should contain all the relevant features.
- The app should be user friendly.

2. App Inventor has two basic views:

Design View: This view contains all the components required to design an application.

Block Editor View: This view is the place where we combine blocks to execute an application.

3. Web apps are different from websites. The major difference is that a web app can be a small part of a website that provides a particular functionality.

4. Native apps are platform-dependent which means that these apps are primarily developed for a specific platform. The address book, gallery, microphone, and camera are a few examples of native applications.

IN THE LAB

Do it yourself.

Periodic Assessment-4

(Based on chapters 8 & 10)

A.

1. Mobiles App	2. Google Play Store	3. iOS	4. Web app
5. App store	6. Hybrid app	7. Games	8. Educational apps
9. Social media apps	10. Web apps		

B. 1. T 2. T 3. F 4. F 5. T

C.

```
#include<iostream.h>
#include<conio.h>
void convertDistance(float km)
{
    float meters = km * 1000;
    float centimeters = km * 100000;

    cout << "Distance in meters: " << meters << " m" << endl;
    cout << "Distance in centimeters: " << centimeters << " cm" << endl;
}
void main()
{
    clrscr();
    float km;
    cout << "Enter distance in kilometers: ";
    cin >> km;
    convertDistance(km);
    getch();
}
```

Test Sheet-2

(Based on chapters 6 to 10)

Section A

A. 1. a. 2. a. 3. c. 4. a. 5. d.
6. d. 7. d. 8. c.

B. 1. isdigit() 2. do-while 3. web browser 4. reference 5. functions
6. header 7. nested

C. 1. T 2. F 3. T 4. T 5. T
6. F 7. T

Section B

A. 1. The return statement is a jump statement used to exit from a function and pass control back to the calling function.

2. Inheritance is one of the most important features of object-oriented programming that allows a class to acquire or inherits all the properties and behaviours of its parent class.

3. C++ offers four jump statements — break, continue, return and goto.

4. To use any library function in your program, respected header file should be included at the beginning of your program code. You can include a header file in your program by using the preprocessor directive #include followed by name of the header file in between the angular brackets.

5. C++ allows you to declare and initialise a variable inside the for loop.

   Syntax: for (int i = 0; i < 10; i++)

6. Examples of e-commerce apps are Amazon and Blinkit.

B. 1. Function Overloading: The process of defining multiple functions with the same name but with different numbers, types, or sequences of parameters.

   Function Overriding: The process of redefining a function with the same name, return type, and argument list in both the subclass and the superclass.

2. Yes, a while loop can contain another while loop, also known as a nested while loop.

   Syntax of a Nested While Loop:

```
while (condition1)
{
    // Outer loop body
    while (condition2)
    {
        // Inner loop body
    }
    // Outer loop continues...
}
```

3. The parameters passed at the time of declaring a function are called formal parameters. On the other hand, the parameters passed at the time of calling a function are called actual arguments.

4.

Function Name	Use	Example	Output
isalpha()	It returns true if the character c is an uppercase or lowercase letter.	isalpha('B')	True
Isalnum()	It returns true if the character c is a digit (0-9) or a letter (either uppercase or lowercase). Otherwise, it returns false.	isalnum('9')	True

5. While developing an app, we should keep some points in mind:
   - The idea of developing an app should be unique.
   - The app we are going to develop should contain all the relevant features.
   - The app should be user friendly.