PRITHIVI NARAYAN CAMPUS

Bagar-17,Pokhara



Labwork of OOP

Submitted by: Submitted to:

Shikshya Adhikari Gyaneshwor Dhungana

Roll no:67

Level:2nd semester

LAB-1:2081-02-13

OBJECTIVE: To write simple programs using c++

THEORY:

C++ is a popular programming language that was developed as an extension of the C programming language. It supports both procedural and object-oriented programming paradigms, making it a versatile language used for developing a wide range of applications, including system software, game development, and more. C++ provides features such as classes, inheritance, polymorphism, data abstraction, and encapsulation. It is known for its performance and is used in many high-performance applications where efficiency is crucial.

A basic structure of a C++ program typically includes the following components:

1. **Preprocessor Directives:** These are instructions to the compiler that start with a # symbol. Common preprocessor directives include including header files (#include <iostream>), defining constants, and conditional compilation.
2. **Main Function:** Every C++ program must have a main function, which serves as the entry point of the program. It's where the program execution begins. The main function returns an integer value indicating the exit status of the program.
3. **Statements and Expressions:** These are the actual instructions that the program will execute. These can include variable declarations, function calls, control structures (like if-else statements, loops), and output statements (using std::cout for output to the console).

LAB WORK:

1. WAP to display the message ”Hello World”

Source code:

#include <iostream>

// Main function

int main() {

// Statements

// Output "Hello, World!"

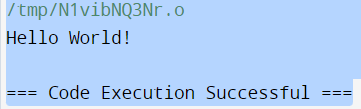
std::cout << "Hello, World!" << std::endl;

// Return 0 to indicate successful execution

return 0;

}

OUTPUT



WAP to input a number and display even or odd.

Source code

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter an integer: ";

cin >> n;

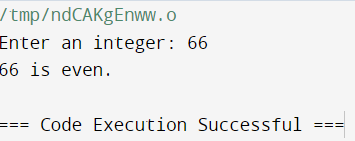
if ( n % 2 == 0)

cout << n << " is even.";

else

cout << n << " is odd.";

return 0; Output:

} 

d.WAP to input a number and display its table upto 10 terms.

Source code:

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter a positive integer: ";

cin >> n;

// run a loop from 1 to 10

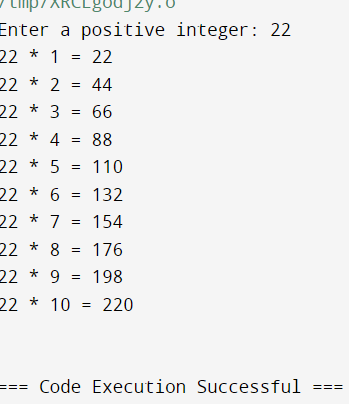
// print the multiplication table

for (int i = 1; i <= 10; ++i) {

cout << n << " \* " << i << " = " << n \* i << endl;

}

return 0; Output:

} 

d.WAP to input two numbers and display their sum,difference,product and division.

Source code:

#include <iostream>

using namespace std;

int main() {

double num1, num2;

cout << "Enter the first number: ";

cin >> num1;

cout << "Enter the second number: ";

cin >> num2;

cout << "Sum: " << (num1 + num2) << endl;

cout << "Difference: " << (num1 - num2) << endl;

cout << "Product: " << (num1 \* num2) << endl;

if(num2 != 0) {

cout << "Division: " << (num1 / num2) << endl;

}

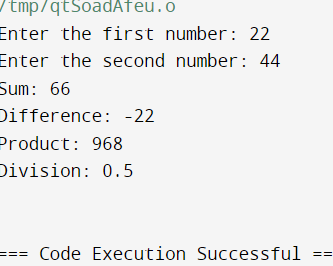
else {

cout << "Cannot divide by zero!" << endl;

}

return 0;

Output



e.WAP to input three numbers and display the middle value.

Source code:

#include <iostream>

using namespace std;

int main(){

int num1=1234,num2=9876,num3=4563;

if(num1>num2){

if(num2>num3){

cout<<num2<<" is a middle number";

}

else if(num3>num1){

cout<<num1<<" is a middle number";

}

else{

cout<<num3<<" is a middle number";

}

}

else{

if(num2<num3){

cout<<num2<<" is a middle number";

}

else if(num3<num1){

cout<<num1<<" is a middle number";

}

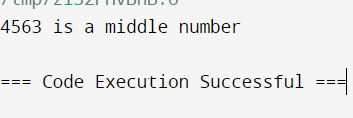
else{

cout<<num3<<" is a middle number";

}

Return 0;

} Output:



f.WAP to input a number and display prime or composite.

Source code:

#include <iostream>

using namespace std;

bool isPrime(int num) {

if (num <= 1) return false;

if (num <= 3) return true;

if (num % 2 == 0 || num % 3 == 0) return false;

for (int i = 5; i \* i <= num; i += 6) {

if (num % i == 0 || num % (i + 2) == 0)

return false;

}

return true;

}

int main() {

int number;

cout << "Enter a number: ";

cin >> number;

if (isPrime(number))

cout << number << " is a prime number.";

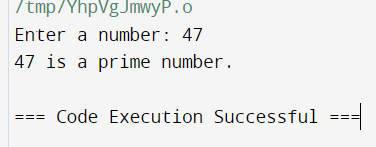
else

cout << number << " is a composite number.";

return 0;

}

Output:



Discussion:

From (a) we learned simply to print the hello world. From (b) we input a number and learned to display whether the given number is odd or even. From (c) we input a number and generate its multiplication table up to 10th terms by using a looping statement (for loop). From (d) we learned to input two inputs and display their sum, difference, multiplication, division. From (e) we learned to input three different numbers and display the middle value among them using if else From above we can see various source code of simple C++ programs with their ladder. And lastly, from (f) we input a number and learned to display either it is prime or composite by using both conditional (if else) and looping statements (for loop).

Conclusion

Hence, in this lab we studied C++ and learned to write simple programs using C++.

LAB-2: 2081/02/14

OBJECTIVE: To Illustrate the concept of Classes and Objects in C++.

THEORY

C++ is an object-oriented programming language. Everything in C++ is associated with classes and objects, along with their attributes and methods. For example: in real life, a car is an object. The car has attributes, such as weight and color, and methods, such as drive and brake. Attributes and methods are basically variables and functions that belong to the class. These are often referred to as "class members".

A class is a user-defined data type that we can use in our program, and it works as an object constructor, or a "blueprint" for creating objects.

Creating a class:

To create a class, use the class keyword:

Example: Create a class called “M

class MyClass { // The class

public: // Access specifier

int myNum; // Attribute (int variable)

string myString; // Attribute (string variable)

};

Here, the class keyword is used to create a class called MyClass. The public keyword is an access specifier, which specifies that members (attributes and methods) of the class are accessible from outside the class. Inside the class, there is an integer variable myNum and a string variable myString. When variables are declared within a class, they are called attributes. At last, end of the class definition with a semicolon ; .

Creating an Object

In C++, an object is created from a class. We have already created a class named MyClass, so now we can use this to create objects. To create an object of MyClass, specify the class name, followed by the object name. To access the class attributes (myNum and myString), use the dot syntax(.) on the object.

Example: Creating an object called “myObj” and access the attributes:

class MyClass { // The class

public: // Access specifier

int myNum; // Attribute (int variable)

string myString; // Attribute (string variable)

};

int main(){

MyClass myObj; //Create an object of MyClass

myObj.myNum=15; //Access attributes and set values

myObj.myString=” some text”;

//Print attribures values

cout<<myObj,myNum<<”\n” ;

cout<<myObject.myString;

return 0;

}

a. Simulate a Bank Account using classes and objects.

Source Code:

#include<iostream>

using namespace std;

class BankAccount{

private:

float balance;

public:

void deposite(float amount){

balance+=amount;

cout<<"Rs."<<amount<<"Deposited."<<endl;

}

void withdraw(float amount){

balance-=amount;

cout<<"Rs."<<amount<<"Withdrawn."<<endl;

}

void getBalance(){

cout<<"Your Current Balance is:Rs"<<balance<<endl;

}

};

int main(){

BankAccount B1;

B1.getBalance();

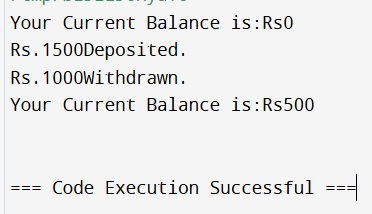
B1.deposite(1500);

B1.withdraw(1000);

B1.getBalance();

return 0;

OUTPUT



Discussion

In this program, we have created a class BankAccount. deposite(), getBalance(), withdraw() are different member functions. Here,there is only one member B1 but we can create an infinite number of accounts as per our requirement. deposite() will ask for the amount to be added in available balance, and deposite the amount. Withdraw() will ask for the amount to be withdrawn from the available, will also check the available balance, if balance is available, it will deduct the amount from the available balance. And getBalance() will show the actual amount present in our account.

Conclusion

Hence, in this lab we illustrated the concept of Classes and Objects in C++ by stimulating a Bank account using classes and objects.