Meeting 3

Avicena Yosanta Muchtar 24/536710/PA/22764

12 September 2024

Github:

https://github.com/TheKurusUGM/Praktikum-Pemograman-UGM-/tree/33ed71818c17480c77c8570ba425897dc0a8a459/Week_3

1 First Number

Question:
Write a program that calculates the determinant of a quadratic equation and determines its roots.

Based on the value of the determinant, there are three possible cases for the roots of the equation:
a. If D > 0, then x1 and x2 are real and distinct. The formulas for calculating them are: x1 = (-b + sqrt(D)) / 2a
x2 = (-b - sqrt(D)) / 2a
b. If D = 0, then x1 and x2 are real and equal.
c. If D < 0, then x1 and x2 are imaginary.

Answer:

```
1
2 #include <iostream>
3 #include <cmath>
4
5 int main() {
6
7 double a, b, c;
8
9 std::cout << "Enter the coefficients a, b, and c for the
      ↪quadratic equation ax^2 + bx + c = 0:\n";
10 std::cout << "a: ";
11 std::cin >> a;
12 std::cout << "b: ";
13 std::cin >> b;
14 std::cout << "c: ";
15 std::cin >> c;
16
```

```
double D = b * b - 4 * a * c;

if (D > 0) {

double x1 = (-b + sqrt(D)) / (2 * a);
double x2 = (-b - sqrt(D)) / (2 * a);
std::cout << "The equation has two real and distinct
    ↪roots:\n";
std::cout << "x1 = " << x1 << "\n";
std::cout << "x2 = " << x2 << "\n";
} else if (D == 0) {

double x = -b / (2 * a);
std::cout << "The equation has two real and equal
    ↪roots:\n";
std::cout << "x1 = x2 = " << x << "\n";
} else {

double realPart = -b / (2 * a);
double imaginaryPart = sqrt(-D) / (2 * a);
std::cout << "The equation has two imaginary roots:\n"; std::cout << "x1 = "
<< realPart << " + " <<
    ↪imaginaryPart << "i\n";
std::cout << "x2 = " << realPart << " - " <<
    ↪imaginaryPart << "i\n";
}

return 0;
}
```

This C++ program designed to solve a quadratic equation of the form: ax^2 + bx + c = 0

The program calculates the discriminant D = b^2 - 4ac , which determines the nature of the roots. Based on the value of D , the program identifies whether the roots are real and distinct, real and equal, or complex (imaginary).

2 Second number

Write a program that determines the largest number among three numbers the user enters. Example:

```
Input number 1: 15
Input number 2: 7
Input number 3: 22
The largest number is: 22
```

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

int main() {

	int num1, num2, num3;


	cout << "Input number 1: ";
	cin >> num1;
	cout << "Input number 2: ";
	cin >> num2;
	cout << "Input number 3: ";
	cin >> num3;

	int largest = num1;
	if (num2 > largest) {
		largest = num2;
	}

	if (num3 > largest) {
		largest = num3;
	}


	cout << "The largest number is: " << largest << endl;

	return 0;
}
```

Explanation:
- The program asks the user to input three numbers.
- It starts by assuming the first number (num1) is the largest.
- It then compares num1 with num2 and num3 to find the actual largest number. - Finally, it prints out the largest number.
- cin: This is used to take input from the user.

The user enters each number when prompted by the cout statements like "Input number 1:", etc.

After this phase, the program will have the three numbers stored in num1, num2, and num3. The variable largest is introduced to store the largest number so far, and it is initially set to num1.
At this point, the program assumes num1 is the largest number. However, it hasn't checked the other two numbers yet. Now, the program checks if num2 is larger than the current value- Condition: The if statement checks if num2 > largest.
then:
- If the condition is true (i.e., num2 is greater than num1), the program updates the largest variable to be num2. This means that at this point, num2 is now considered the largest.
- If num2 is not greater than num1, the value of largest remains as num1.

3 Third and fourth question

3. List the branching instructions in the C++ programming language!
4. Explain the characteristics of each type of branching instruction! Branching instruction in C++ language:
- Conditional Statements:
These allow you to make decisions in your code based on conditions.

- if: Executes a block of code if a condition is true.
- if-else: Executes one block if the condition is true, and another if it's false. - if-else: Executes one block if the condition is true, and another if it's false. - switch: Selects one of many code blocks to execute based on the value of an expression.
- Ternary (conditional) operator ?:: A shorthand for if-else.

- Jump Statements
These control the flow by altering the normal sequence of execution.
- break: Exits from a loop or switch statement.
- continue: Skips the current iteration of a loop and moves to the next one. - goto: Jumps to a labeled statement in the code. Use sparingly, as it can make code hard to read and maintain.

- return: Exits from a function, optionally returning a value.

- Loops
Loops are used to execute a block of code multiple times.
- for: Repeats a block of code for a specific number of times.
- while: Repeats a block of code while a condition is true.

- do-while: Similar to while, but guarantees at least one execution of the code block.

- Function Calls
Functions allow you to organize code into reusable blocks and enable branching through different function calls.
- Function Declaration: You define a function and call it based on certain conditions. - Recursive Function Calls: A function can call itself, useful for tasks that can be broken down into similar sub-tasks.

- Exception handling used for dealing with errors or unexpected situations in a program. - try block: Wraps code that might throw an exception.
- catch block: Catches and handles exceptions thrown in the try block.
- throw: Used to manually throw an exception.
- Conditional Branching with Loops
Loops can branch within themselves using break, continue, and return. For example:

```
for (int i = 0; i < 10; ++i) {
        if (i == 5) {
                continue; // Skip the rest of this loop iteration }
        if (i == 8) {
                break; // Exit the loop
        }
}
```