Experiment No: 1

Experiment Name: Detecting Comments in C/C++ Code

Problem Structure:

Write a C/C++ program that takes a line of code as input and determines whether it is a single-line comment (e.g.: // comment, int x; //declare variable) or a multi-line comment (e.g., /* comment */, int x = 10; /* variable declare and value assign */).

Procedure:

Input Collection:

- Use the getline function to collect a line of code from the user.

Comment Detection:

- Implement a logic that checks for single-line comments and multi-line comments in the input.
- Utilize a stack-based approach to handle multi-line comment detection.

Output Presentation:

- Display the result, indicating whether the input line is a single-line comment, a multi-line comment, or not a comment at all.

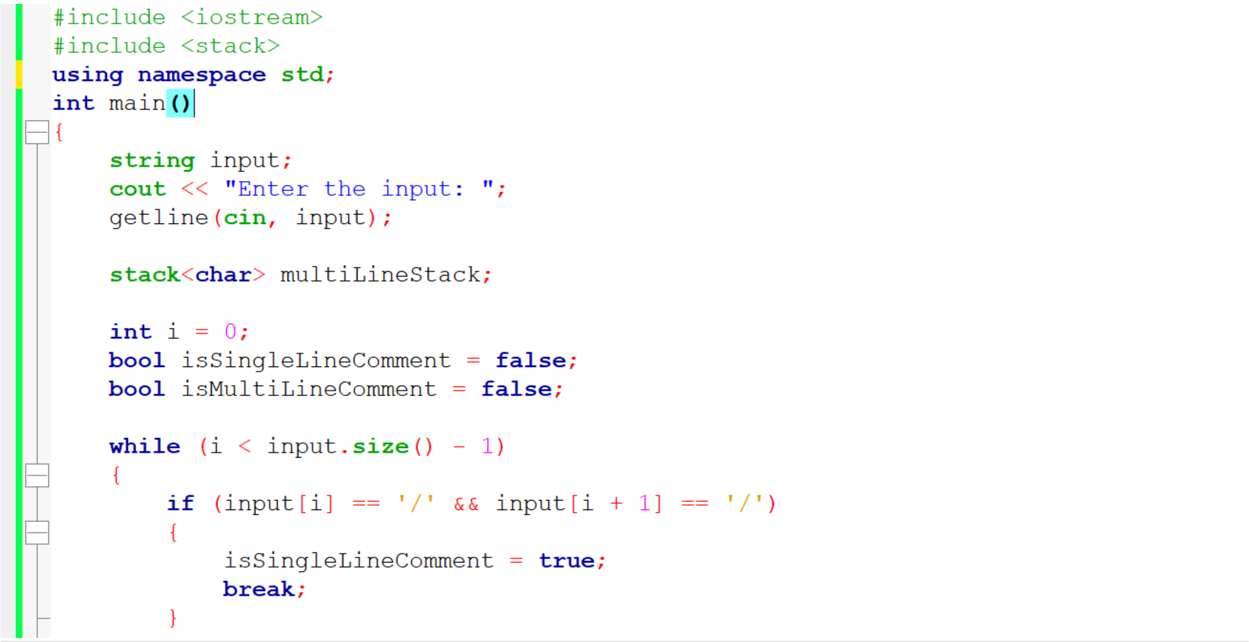
Code Structure:

- Use structured programming techniques to organize the code logically.
- Implement functions or blocks for input collection, comment detection, and output presentation.

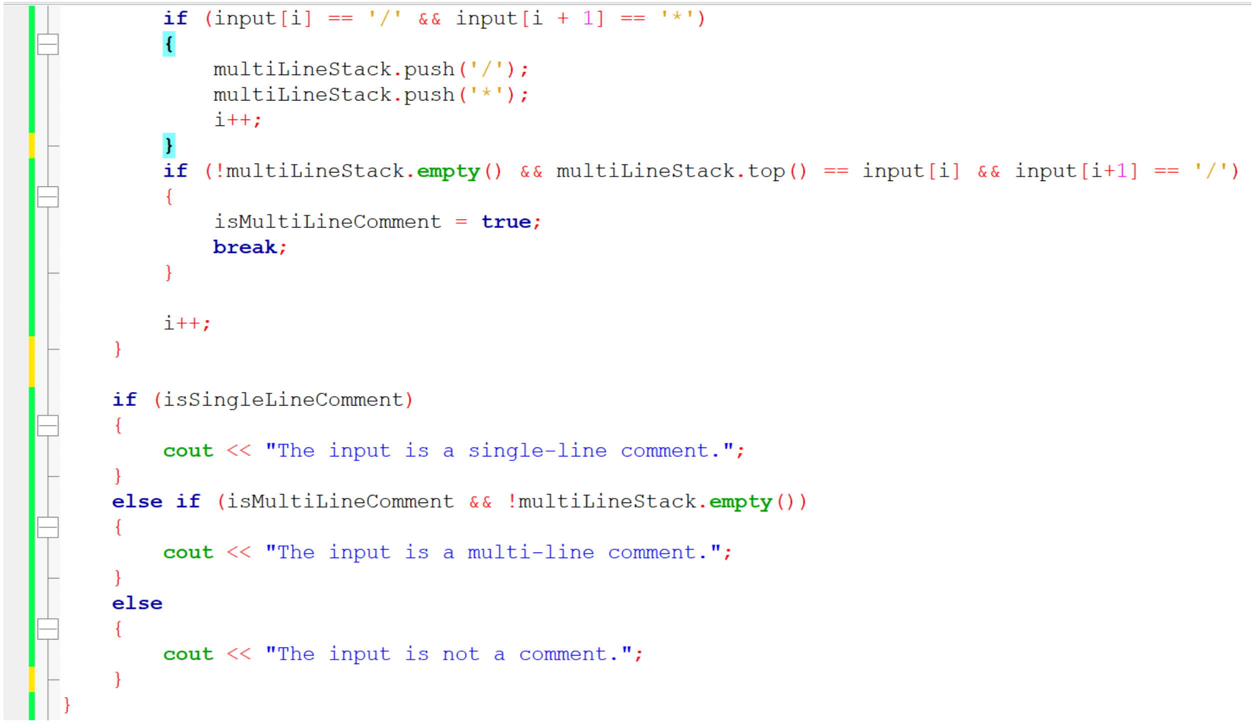
Testing:

- Test the program with various inputs, including single-line comments, multi-line comments, and code without comments.
- Verify that the program correctly identifies the nature of each input.

Code:

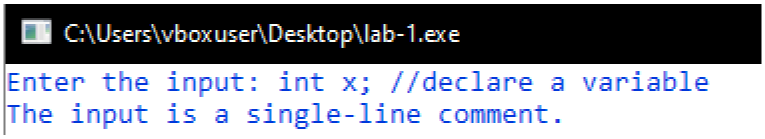


```cpp
#include <iostream>
#include <stack>
using namespace std;
int main()
{
    string input;
    cout << "Enter the input: ";
    getline(cin, input);

    stack<char> multiLineStack;

    int i = 0;
    bool isSingleLineComment = false;
    bool isMultiLineComment = false;

    while (i < input.size() - 1)
    {
        if (input[i] == '/' && input[i + 1] == '/')
        {
            isSingleLineComment = true;
            break;
        }
```

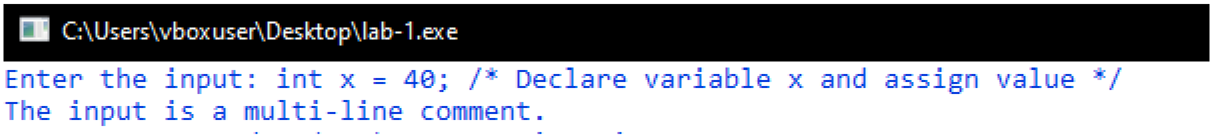


```cpp
        if (input[i] == '/' && input[i + 1] == '*')
        {
            multiLineStack.push('/');
            multiLineStack.push('*');
            i++;
        }
        if (!multiLineStack.empty() && multiLineStack.top() == input[i] && input[i+1] == '/')
        {
            isMultiLineComment = true;
            break;
        }

        i++;
    }

    if (isSingleLineComment)
    {
        cout << "The input is a single-line comment.";
    }
    else if (isMultiLineComment && !multiLineStack.empty())
    {
        cout << "The input is a multi-line comment.";
    }
    else
    {
        cout << "The input is not a comment.";
    }
}
```

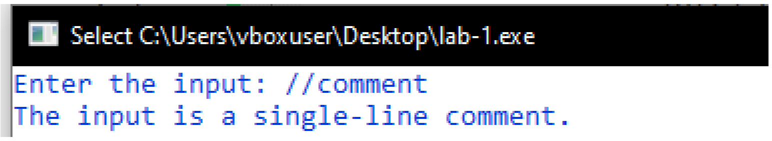
Input-Output:



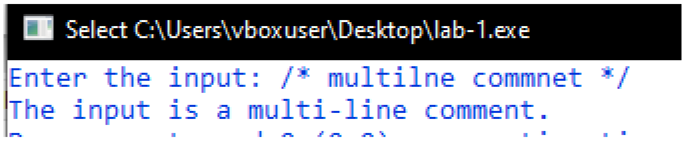
```
Enter the input: int x; //declare a variable
The input is a single-line comment.
```



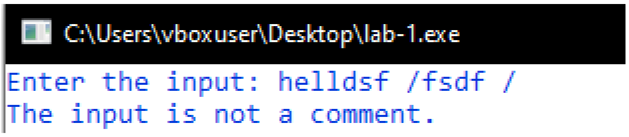
```
Enter the input: int x = 40; /* Declare variable x and assign value */
The input is a multi-line comment.
```



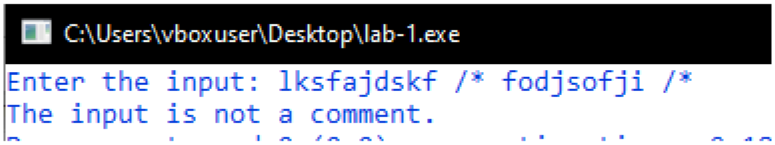
```
Enter the input: //comment
The input is a single-line comment.
```



```
Enter the input: /* multilne commnet */
The input is a multi-line comment.
```



```
Enter the input: helldsf /fsdf /
The input is not a comment.
```



```
Enter the input: lksfajdskf /* fodjsofji /*
The input is not a comment.
```