Mini Project

AIM : Loop Syntax Analyzer (C++/C)

Pseudo code :

```
BEGIN
  PRINT "Enter the loop code to analyze:"
  READ loopCode

  IF parentheses or braces are not balanced THEN
    PRINT "Syntax Error: Unmatched parentheses or braces!"
    EXIT

  IF loopCode starts with "for" THEN
    CHECK if it follows for-loop syntax
    IF valid THEN
      PRINT "Valid Loop Syntax!"
      EXIT

  IF loopCode starts with "while" THEN
    CHECK if it follows while-loop syntax
    IF valid THEN
      PRINT "Valid Loop Syntax!"
      EXIT

  IF loopCode starts with "do" THEN
    CHECK if it follows do-while loop syntax
```

```
    IF valid THEN
        PRINT "Valid Loop Syntax!"
        EXIT

  PRINT "Syntax Error: Invalid loop syntax!"
END
```

Code :

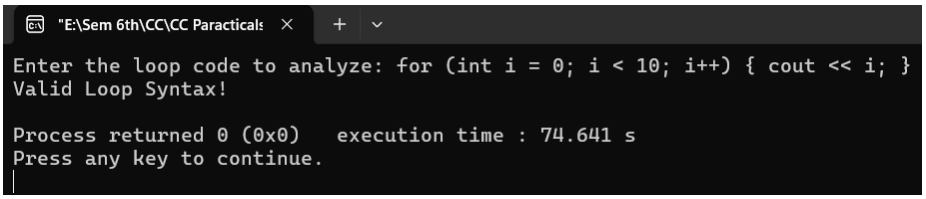
```
#include <iostream>
#include <stack>
#include <regex>
using namespace std;

bool isValidForLoopSyntax(const string& loop) {
  regex forRegex("^for\\s*\\(.*;.*;.*\\)\\s*\\{.*\\}$");
  return regex_match(loop, forRegex);
}

bool isValidWhileLoopSyntax(const string& loop) {
  regex whileRegex("^while\\s*\\(.*\\)\\s*\\{.*\\}$");
  return regex_match(loop, whileRegex);
}

bool isValidDoWhileLoopSyntax(const string& loop) {
  regex doWhileRegex("^do\\s*\\{.*\\}\\s*while\\s*\\(.*\\);$");
  return regex_match(loop, doWhileRegex);
```

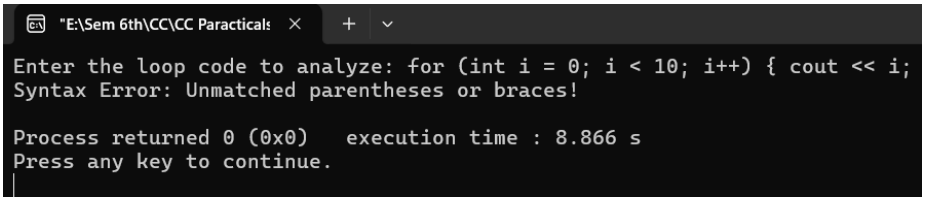
```
}

bool isValidLoop(const string& loop) {
    if (loop.find("for") == 0) {
        return isValidForLoopSyntax(loop);
    }
    if (loop.find("while") == 0) {
        return isValidWhileLoopSyntax(loop);
    }
    if (loop.find("do") == 0) {
        return isValidDoWhileLoopSyntax(loop);
    }
    return false;
}

bool areBracesBalanced(const string& loop) {
    stack<char> braces;
    for (char c : loop) {
        if (c == '{') braces.push(c);
        if (c == '}') {
            if (braces.empty()) return false;
            braces.pop();
        }
    }
    return braces.empty();
}
```

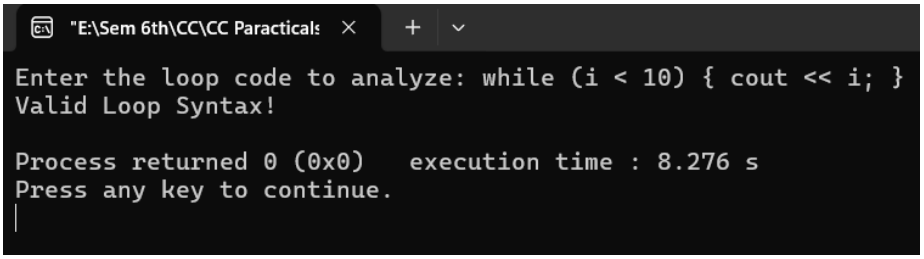
```
bool areParenthesesBalanced(const string& loop) {
    stack<char> parentheses;
    for (char c : loop) {
        if (c == '(') parentheses.push(c);
        if (c == ')') {
            if (parentheses.empty()) return false;
            parentheses.pop();
        }
    }
    return parentheses.empty();
}

int main() {
    string loopCode;
    cout << "Enter the loop code to analyze: ";
    getline(cin, loopCode);

    if (!areParenthesesBalanced(loopCode) || !areBracesBalanced(loopCode)) {
        cout << "Syntax Error: Unmatched parentheses or braces!" << endl;
        return 0;
    }

    if (isValidLoop(loopCode)) {
        cout << "Valid Loop Syntax!" << endl;
    } else {
```

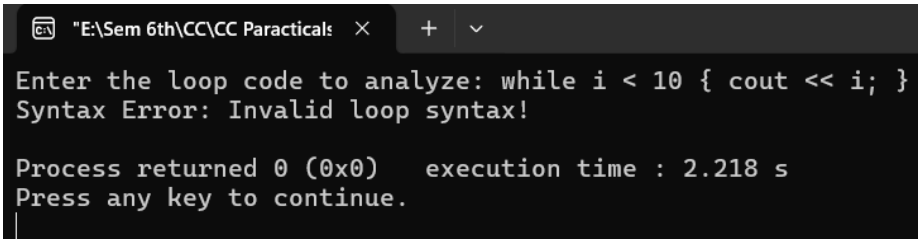
```
        cout << "Syntax Error: Invalid loop syntax!" << endl;
    }


    return 0;
}
```

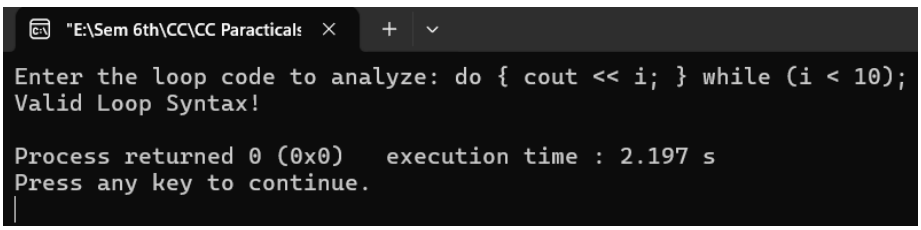
Output :

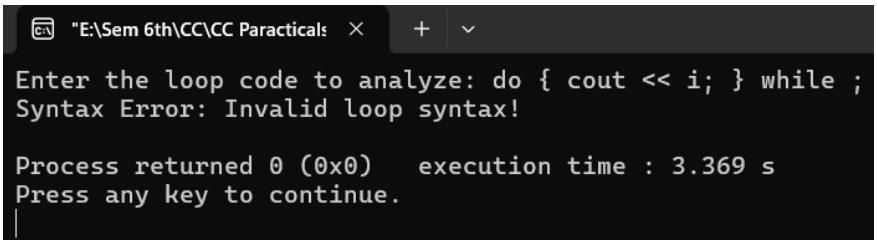


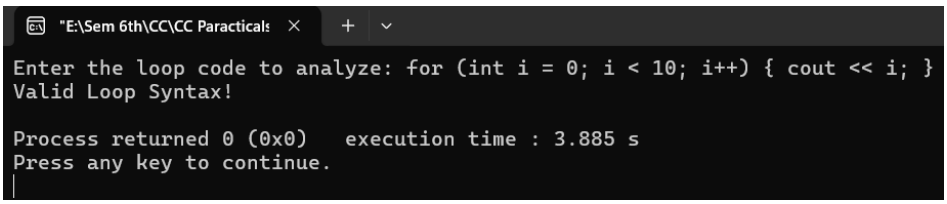


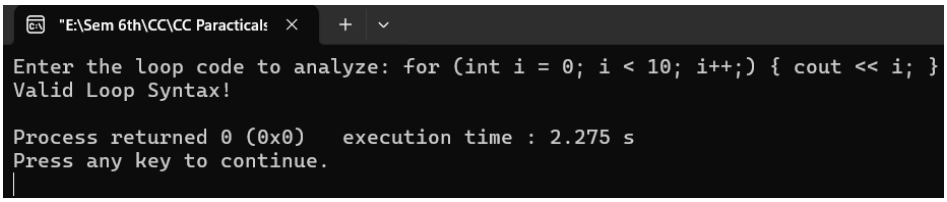


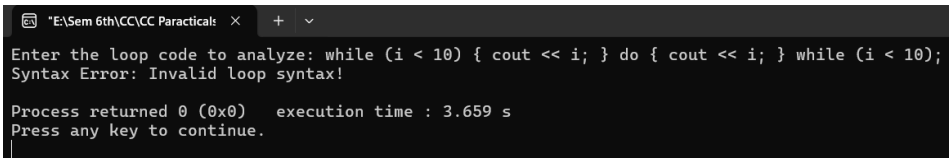


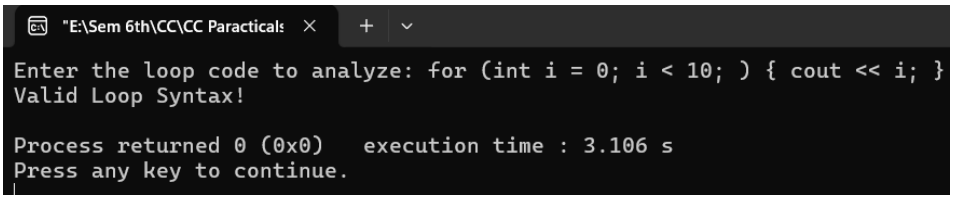



"E:\Sem 6th\CC\CC Practicals
Enter the loop code to analyze: do { cout << i; } while ;
Syntax Error: Invalid loop syntax!
Process returned 0 (0x0) execution time : 3.369 s
Press any key to continue.


"E:\Sem 6th\CC\CC Practicals
Enter the loop code to analyze: for (int i = 0; i < 10; i++) { cout << i; }
Valid Loop Syntax!
Process returned 0 (0x0) execution time : 3.885 s
Press any key to continue.


"E:\Sem 6th\CC\CC Practicals
Enter the loop code to analyze: for (int i = 0; i < 10; i++;) { cout << i; }
Valid Loop Syntax!
Process returned 0 (0x0) execution time : 2.275 s
Press any key to continue.


"E:\Sem 6th\CC\CC Practicals
Enter the loop code to analyze: while (i < 10) { cout << i; } do { cout << i; } while (i < 10);
Syntax Error: Invalid loop syntax!
Process returned 0 (0x0) execution time : 3.659 s
Press any key to continue.


"E:\Sem 6th\CC\CC Practicals
Enter the loop code to analyze: for (int i = 0; i < 10;) { cout << i; }
Valid Loop Syntax!
Process returned 0 (0x0) execution time : 3.106 s
Press any key to continue.