Experiment 3

Student Name:Harsh Kumar **UID:22BCS15754**
Branch:BE-CSE **Section/Group:FL_IOT_603(B)**
Semester:5th **Date of Performance:05/08/24**
Subject Name:Advanced Programming **Subject Code:22CSP-314**

1. **Aim:**: Given an expression string exp, write a program to examine whether the pairs and the orders of "{", "}", "(", ")", "[", "]" are correct in the given expression A={(a,b)}.

2. **Objective:**

The objective of this program is to determine whether the given expression string, such as A={(a,b)}, has correctly paired and properly ordered brackets.

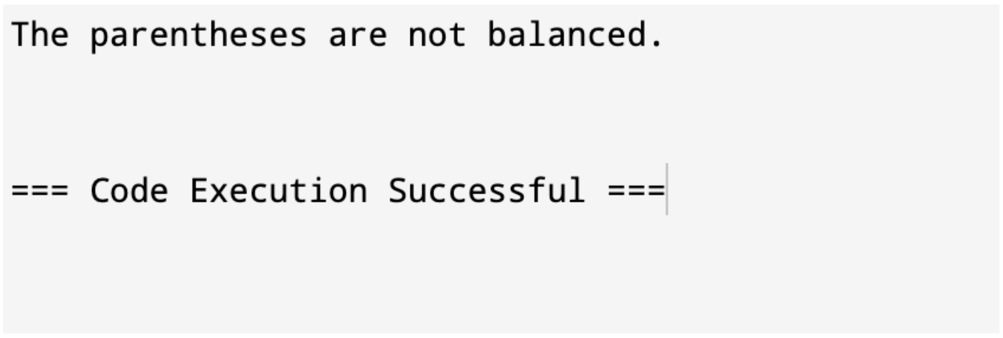
3. **Implementation/Code:**

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

bool isBalanced(const string& expression) {
   stack<char> s;
   for (char ch : expression) {
      if (ch == '(' || ch == '{' || ch == '[') {
         s.push(ch);
      } else if (ch == ')' || ch == '}' || ch == ']') {
         // If stack is empty, it's an unbalanced expression
         if (s.empty()) {
            return false;
         }
         // Check if the closing bracket matches the top of the stack
```

```
            char top = s.top();
            if ((ch == ')' && top != '(') ||
                (ch == '}' && top != '{') ||
                (ch == ']' && top != '[')) {
                return false;
            }
            s.pop();
        }
    }
    // If stack is empty, parentheses are balanced
    return s.empty();
}
int main() {
    string expression = "a={(x,y)";
    if (isBalanced(expression)) {
        cout << "The parentheses are balanced." << endl;
    } else {
        cout << "The parentheses are not balanced." << endl;
    }
    return 0;
}
```

Output



```
The parentheses are not balanced.


=== Code Execution Successful ===
```

Time Complexity: O(n)

1. **(B) Aim:** Given a number N, you can perform the following two operations to reduce N to 0: 1: If we take 2 integers a and b where N = a x b (a ≠ 1, b ≠ 1), , then we can change N = max (a,b) 2: Decrease the value of N by 1. Determine the minimum number of moves required to reduce the value of N to 0..

2. **Objective:**

The objective of the question is to determine the minimum number of moves required to reduce a given integer N to 0 by using a combination of two specific operations.

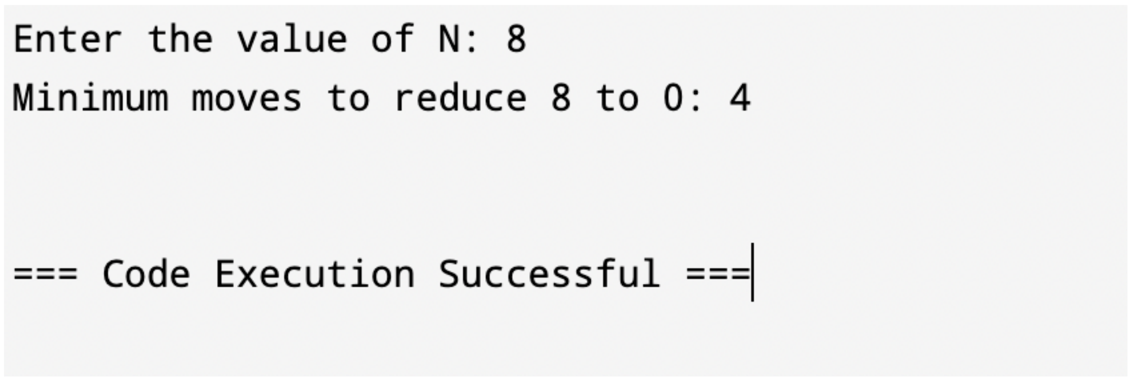
3. **Implementation/Code:**

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

int minMovesToZero(int N) {
    stack<int> s;
    s.push(N);
    int moves = 0;

    while (!s.empty()) {
        int current = s.top();
        s.pop();
        if (current == 0) {
            continue;
        }
        bool factorFound = false;
        for (int i = 2; i <= sqrt(current); ++i) {
            if (current % i == 0) {
                s.push(max(i, current / i));
                factorFound = true;
                break;
            }
        }
        if (!factorFound) {
            s.push(current - 1);
        }
        moves++;
    }
    return moves;
}
int main() {
    int N;
    cout << "Enter the value of N: ";
    cin >> N;
    int result = minMovesToZero(N);
    cout << "Minimum moves to reduce " << N << " to 0: " << result << endl;
    return 0;
}
```

OUTPUT



```
Enter the value of N: 8
Minimum moves to reduce 8 to 0: 4


=== Code Execution Successful ===
```

Time Complexity: O(N × √N)