Assignment No.5

Name: Omkar Kulkarni
PRN: B25CE2012
Class: SY1
Batch: C

TITLE:-

Write a program using a stack for push, pop, peek, and isEmpty operations. Write is Balanced() Function that Iterates through the input expression, Pushes opening brackets onto the stack. For closing brackets, it checks the top of the stack for a matching opening bracket. Ensures that all opening brackets are matched by the end of the traversal. Main Function: Accepts a string expression from the user. Uses isBalanced() to determine if the parentheses in the expression are balanced.

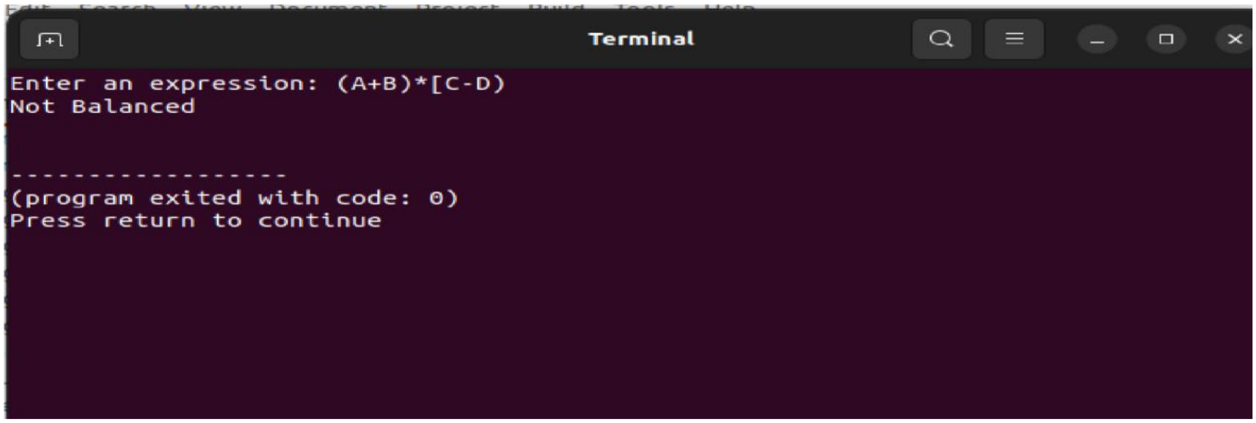
CODE:-

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

#define MAX 100 // maximum size of stack

// Custom stack implementation
class Stack { char arr[MAX];
int top; public:
Stack() { top = -1; }

bool isEmpty() { return top == -1; }
bool isFull() { return top == MAX - 1;
}

void push(char c) {
if (!isFull()) {
arr[++top] = c;
}
}

char pop() { if
(!isEmpty()) {
```

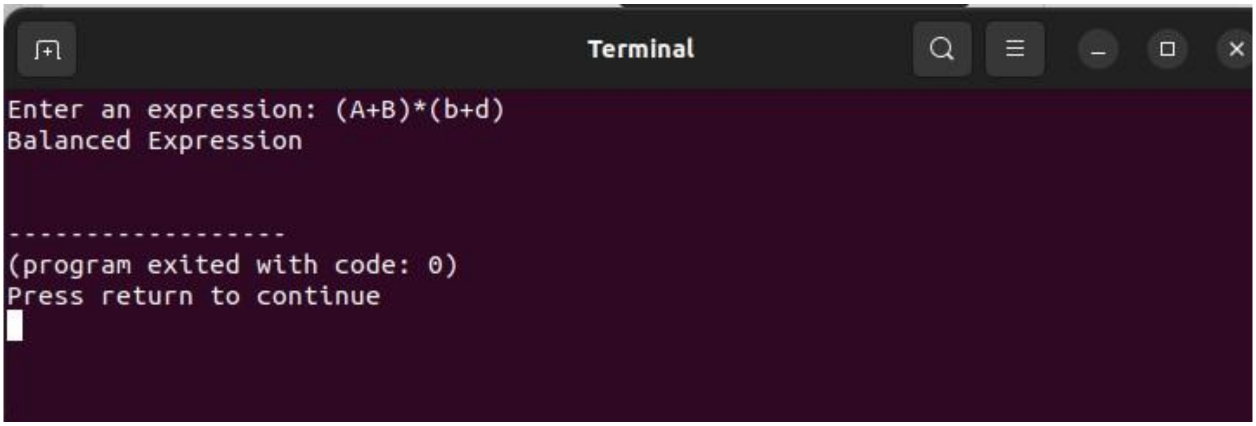
```
return arr[top--
];
}
return '\0'; // return null char if empty
}

char peek() {
if (!isEmpty())
{ return
arr[top];
} return
'\0';
}
};

// Function to check if brackets are balanced bool
isBalanced(string expr) {
Stack s;

for (char ch : expr) { if (ch == '('
|| ch == '{' || ch == '[') {
s.push(ch);
} else if (ch == ')' || ch == '}' || ch ==
']') { if (s.isEmpty()) return false;

char top = s.pop(); if ((ch
== ')' && top != '(') || (ch
== '}' && top != '{') || (ch
== ']' && top != '[')) {
return false;
}
}
}
return s.isEmpty();
}

int main() { string expression;
cout << "Enter an expression: ";
cin >> expression;
```

```
if (isBalanced(expression)) cout <<
"Balanced Expression " << endl; else
cout << "Not Balanced " << endl;

return 0;
}
```

OUTPUT :



```
Enter an expression: (A+B)*[C-D)
Not Balanced


-----------------
(program exited with code: 0)
Press return to continue
```



```
Enter an expression: (A+B)*(b+d)
Balanced Expression


-----------------
(program exited with code: 0)
Press return to continue
```