* Assignment - 9 *

Aim :-

In any language program mostly syntax errors occurs due to unbalancing delimiter such as () {}, [] . write c++ program using stack to check whether given expression is well parenthsized or not.

objective :-

1. To understand the concept of stack data structure,
2. To implement stack data structure and its operations for given application.

outcomes :-

After executing Assignment,
- student will be able to identify and use stack data structure for given application
- student design and implement stack data structure and its operation.

os / programing tools:

64- bit open source operating system & programing tool with c++ IDE.

Theory :-

A stack is an ADT, commonly used in most programming tool with c++ IDE language. It is named stack as it have like areal-world stack for example - a deck of cards or a pile of plates, etc. It is based upon LIFO operation i.e.

last-in-first out, A stack can be implemented by mean, of Array, structure, pointer → linked list.

Basic operations:-

① push()- storing element.
② pop()- Removing element.
③ peek()- get the top element, without removing
④ isFull() check if it is full.
⑤ isEmpty()- checks if it is Empty.

1, 2, 3, 4, 5 are user defined functions.

Application of stacks:-
Expression evolution.
Expression conversion infix to postfix, infix to prefix, postfix to infix, prefix to infix pursing.

Algorithm:-

Step 1: Declare a character stack S.
step 2: Now transverse the expression string exp. If the current character is a starting bracket ('(' or '{' or '[') then.
push it to stack.
If the current character is daring bracket then, pop from stack and
if the popped character is matching then fine.

else
Paranthesis are not balanced.

Step 3:- After complete transversal, if there is some starting bracket left in stack then "not balanced"
Step 4: Exit.

Pseudo Code:

① isEmpty()

```
Algorithm isEmpty() {
        if (top == -1) {
            return 1;
        else
            return 0;
        }
```

② isfull()

```
Algorithm isfull() {
    if (top == Max-1)
        return 1;
    else
        return 0;
```

③ push()

```
Algorithm push (char ch) {
        if (!isfull()) {
            top++;
            s[top] = ch;
        }
    }.
```

④ pop()

```
Algorithm pop(){
    if (!isEmpty()){
        char ch = s[top];
        top--;
        return ch;
    }
    else.
        return '\0';
```

Conclusion:-

Different stack operations are implemented successfully.