## Question1.cpp

```
#include <iostream>
    #include <string>
    using namespace std;
 6
    class stack
 7
 8
        float arr[20];
 9
        int top = -1;
10
    public:
11
        stack()
12
13
        {
             for (int i = 0; i < 20; i++)</pre>
14
15
                 arr[i] = 0;
16
17
18
        bool full()
19
20
21
             return top == 19;
22
23
        bool empty()
24
25
             return top == -1;
26
27
28
        void push(float value)
29
             if (!full())
30
31
             {
32
                 top++;
33
                 arr[top] = value;
34
             else
35
36
37
                 cout << "Array is Full.\n";</pre>
38
                 return;
39
40
        float pop()
41
42
43
             if (!empty())
44
45
                 int temp = arr[top];
                 top--;
46
47
                 return temp;
48
             }
             else
49
50
51
                 cout << "Array is Empty.\n";</pre>
52
                 return -1;
53
```

```
12/11/23, 10:10 PM
   54
   55
        };
   56
   57
        float solveeq(stack &obj, string exp)
   58
   59
   60
   61
   62
   63
   64
   65
   66
   67
   68
   69
   70
   71
   72
   73
   74
   75
   76
   77
   78
   79
   80
   81
   82
   83
   84
   85
   86
   87
   88
   89
   90
   91
```

92

93 94 95

96 97

98

99

100

101 102 string curr = "";

}

}

{

string exp;

as INTEGERS): \n";

cin >> exp;

stack e;

int main()

else

return obj.pop();

curr = exp[i];

for (int i = 0; i < exp.length(); i++)</pre>

**if** (curr >= "0" && curr <= "9")

obj.push(stoi(curr));

float temp = obj.pop();

float temp = obj.pop();

float temp = obj.pop(); obj.push(obj.pop() / temp);

float temp = obj.pop();

obj.push(obj.pop() \* temp);

cout << "Invalid Character!" << endl;</pre>

cout << "Answer of your Expression: " << solveeq(e, exp) << endl;</pre>

cout << "Enter Postfix Expression without Exponents (Numbers will be read as Single Digits and

obj.push(obj.pop() - temp);

obj.push(obj.pop() + temp);

**else if** (exp[i] == '+')

**else if** (exp[i] == '-')

**else if** (exp[i] == '/')

**else if** (exp[i] == '\*')

```
localhost:62373/6bcd7efb-c489-4321-ae8c-16e319439646/
```

## Question2.cpp

```
#include <iostream>
 2
    #include <string>
 3
 4
    using namespace std;
 5
6
    class stack
7
8
        char arr[20];
9
        int top = -1;
10
    public:
11
12
        stack()
13
        {
             for (int i = 0; i < 20; i++)
14
15
                 arr[i] = 0;
16
17
18
19
        bool full()
20
21
             return top == 19;
22
        }
23
        bool empty()
24
25
             return top == -1;
26
        }
27
28
        void push(char value)
29
30
             if (!full())
31
             {
32
                 top++;
33
                 arr[top] = value;
34
35
             else
36
37
                 cout << "Array is Full.\n";</pre>
38
                 return;
39
40
        }
41
        char pop()
42
43
             if (!empty())
44
45
                 char temp = arr[top];
46
                 top--;
47
                 return temp;
48
             }
             else
49
50
                 cout << "Array is Empty.\n";</pre>
51
52
                 return -1;
53
             }
```

```
12/11/23, 10:11 PM
  54
  55
          void display()
  56
               cout << "\n--";
  57
  58
               for (int i = 0; i <= top; i++)</pre>
  59
  60
                    cout << arr[i];</pre>
  61
               cout << "--\n";
  62
  63
           }
  64
      };
  65
      bool checkp(string s)
  66
  67
  68
          stack word;
  69
          for (int i = 0; i < s.length(); i++)</pre>
  70
  71
               word.push(s[i]);
  72
          }
           string reversed = "";
  73
  74
          for (int i = 0; i < s.length(); i++)</pre>
  75
               reversed += word.pop();
  76
  77
  78
          return s == reversed;
  79
  80
  81
      int main()
  82
      {
  83
          string str;
           cout << "Enter String to check for palindrome:\n";</pre>
  84
  85
           getline(cin, str);
  86
          if (checkp(str))
  87
               cout << "String is Palindrome\n";</pre>
  88
  89
           }
  90
          else
  91
           {
  92
               cout << "String is NOT Palindrome\n";</pre>
  93
  94 }
```

## Question3.cpp

```
1
    #include <iostream>
    #include <string>
    using namespace std;
 6
    class stack
 7
 8
        char arr[20];
 9
        int top = -1;
10
    public:
11
12
        stack()
13
        {
14
             for (int i = 0; i < 20; i++)
15
                 arr[i] = 0;
16
17
18
        bool full()
19
20
21
             return top == 19;
22
        bool empty()
23
24
25
             return top == -1;
26
27
28
        void push(char value)
29
30
             if (!full())
31
             {
32
                 top++;
33
                 arr[top] = value;
34
             else
35
36
37
                 cout << "Array is Full.\n";</pre>
38
                 return;
39
             }
40
41
        void pop()
42
             if (!empty())
43
44
             {
45
                 top--;
             }
46
             else
47
48
49
                 cout << "Stack is Empty!" << endl;</pre>
50
51
52
        char Top()
53
```

```
54
              if (!empty())
 55
 56
                  return arr[top];
 57
              }
              else
 58
 59
                  cout << "Array Empty, Returning 'X'...\n";</pre>
 60
 61
                  return 'X';
 62
              }
 63
 64
          void display()
 65
              cout << "\n--";
 66
 67
              for (int i = 0; i <= top; i++)</pre>
 68
 69
                  cout << arr[i];</pre>
 70
 71
              cout << "--\n";
 72
 73
     };
 74
 75
     bool MatchDelim(string ex)
 76
     {
 77
         stack delims;
         for (int i = 0; i < ex.length(); i++)</pre>
 78
 79
              if (ex[i] == '{' || ex[i] == '(' || ex[i] == '[')
 80
 81
              {
 82
                  delims.push(ex[i]);
 83
              else if (ex[i] == ')' || ex[i] == '}' || ex[i] == ']')
 84
 85
 86
                  if (ex[i] == ')' && delims.Top() == '(')
 87
                  {
 88
                       delims.pop();
 89
                  else if (ex[i] == ']' && delims.Top() == '[')
 90
 91
                       delims.pop();
 92
 93
                  else if (ex[i] == '}' && delims.Top() == '{')
 94
 95
 96
                       delims.pop();
 97
                  else
 98
99
                  {
100
                       cout << "Open and Close Brackets Mismatched at position " << i << endl;</pre>
101
                       return false;
102
                  }
103
104
          }
105
          if (delims.empty())
106
107
              cout << "Expression is valid\n";</pre>
108
              return true;
109
```

12/11/23, 10:11 PM

```
110
         else
111
         {
112
             cout << "Unclosed Delimiters Remain\n";</pre>
113
             delims.display();
             return false;
114
115
         }
116
117
    int main()
118
119
     {
120
         string exp;
         cout << "Enter Expressions to match the Delimiters:\n";</pre>
121
         getline(cin, exp);
122
123
         MatchDelim(exp);
124 }
```