Question1.cpp

```cpp
#include <iostream>
#include <string>

using namespace std;

class stack
{
    float arr[20];
    int top = -1;

public:
    stack()
    {
        for (int i = 0; i < 20; i++)
        {
            arr[i] = 0;
        }
    }
    bool full()
    {
        return top == 19;
    }
    bool empty()
    {
        return top == -1;
    }

    void push(float value)
    {
        if (!full())
        {
            top++;
            arr[top] = value;
        }
        else
        {
            cout << "Array is Full.\n";
            return;
        }
    }
    float pop()
    {
        if (!empty())
        {
            int temp = arr[top];
            top--;
            return temp;
        }
        else
        {
            cout << "Array is Empty.\n";
            return -1;
        }
```

```
    }
};

float solveeq(stack &obj, string exp)
{
    string curr = "";
    for (int i = 0; i < exp.length(); i++)
    {
        curr = exp[i];
        if (curr >= "0" && curr <= "9")
        {
            obj.push(stoi(curr));
        }
        else if (exp[i] == '+')
        {
            float temp = obj.pop();
            obj.push(obj.pop() + temp);
        }
        else if (exp[i] == '-')
        {
            float temp = obj.pop();
            obj.push(obj.pop() - temp);
        }
        else if (exp[i] == '/')
        {
            float temp = obj.pop();
            obj.push(obj.pop() / temp);
        }
        else if (exp[i] == '*')
        {
            float temp = obj.pop();
            obj.push(obj.pop() * temp);
        }
        else
        {
            cout << "Invalid Character!" << endl;
        }
    }
    return obj.pop();
}

int main()
{
    string exp;
    stack e;
    cout << "Enter Postfix Expression without Exponents (Numbers will be read as Single Digits and as INTEGERS): \n";
    cin >> exp;
    cout << "Answer of your Expression: " << solveeq(e, exp) << endl;
}
```

Question2.cpp

```cpp
#include <iostream>
#include <string>

using namespace std;

class stack
{
    char arr[20];
    int top = -1;

public:
    stack()
    {
        for (int i = 0; i < 20; i++)
        {
            arr[i] = 0;
        }
    }
    bool full()
    {
        return top == 19;
    }
    bool empty()
    {
        return top == -1;
    }

    void push(char value)
    {
        if (!full())
        {
            top++;
            arr[top] = value;
        }
        else
        {
            cout << "Array is Full.\n";
            return;
        }
    }
    char pop()
    {
        if (!empty())
        {
            char temp = arr[top];
            top--;
            return temp;
        }
        else
        {
            cout << "Array is Empty.\n";
            return -1;
        }
```

```
    }
    void display()
    {
        cout << "\n--";
        for (int i = 0; i <= top; i++)
        {
            cout << arr[i];
        }
        cout << "--\n";
    }
};

bool checkp(string s)
{
    stack word;
    for (int i = 0; i < s.length(); i++)
    {
        word.push(s[i]);
    }
    string reversed = "";
    for (int i = 0; i < s.length(); i++)
    {
        reversed += word.pop();
    }
    return s == reversed;
}

int main()
{
    string str;
    cout << "Enter String to check for palindrome:\n";
    getline(cin, str);
    if (checkp(str))
    {
        cout << "String is Palindrome\n";
    }
    else
    {
        cout << "String is NOT Palindrome\n";
    }
}
```

Question3.cpp

```cpp
#include <iostream>
#include <string>

using namespace std;

class stack
{
    char arr[20];
    int top = -1;

public:
    stack()
    {
        for (int i = 0; i < 20; i++)
        {
            arr[i] = 0;
        }
    }
    bool full()
    {
        return top == 19;
    }
    bool empty()
    {
        return top == -1;
    }

    void push(char value)
    {
        if (!full())
        {
            top++;
            arr[top] = value;
        }
        else
        {
            cout << "Array is Full.\n";
            return;
        }
    }
    void pop()
    {
        if (!empty())
        {
            top--;
        }
        else
        {
            cout << "Stack is Empty!" << endl;
        }
    }
    char Top()
    {
```

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

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