Q5.) Write a program to implement recursive decent parser.

```
#include <iostream>
#include <string>
#include <cctype>

using namespace std;

class RecursiveDescentParser {
private:
    string s;
    int i;

    bool match(char a) {
        if (i >= s.length()) return false;
        else if (s[i] == a) {
            i++;
            return true;
        }
        else if (a == 'i'){
            if(isalpha(s[i]) || isdigit(s[i])) {
                i++;
                return true;
            }
        }
        else return false;
    }

    bool F() {
        // F -> (E)
        if (match('(')) {
            if (E()) {
                if (match(')')) return true;
                else return false;
            }
            else return false;
        }
        // F -> i
        else if (match('i')) return true;
        else return false;
    }

    bool Tx() {
        // T' -> *FT'
        if (match('*')) {
            if (F()) {
                if (Tx()) return true;
                else return false;
```

```
            }
            else return false;
        }
        // T' -> ε
        else return true;
    }

    bool T() {
        // T -> FT'
        if (F()) {
            if (Tx()) return true;
            else return false;
        }
        else return false;
    }

    bool Ex() {
        // E' -> +TE'
        if (match('+')) {
            if (T()) {
                if (Ex()) return true;
                else return false;
            }
            else return false;
        }
        // E' -> ε
        else return true;
    }

    bool E() {
        // E -> TE'
        if (T()) {
            if (Ex()) return true;
            else return false;
        }
        else return false;
    }

public:
    RecursiveDescentParser(string& str) : s(str), i(0) {}

    bool parse() {
        if(E()) {
            return (i == s.length());
        }
        return false;
    }
};
```

```cpp
int main() {
    cout << "Recursive Descent Parsing For following grammar\n";
    cout << "E -> TE'\nE'-> +TE'/@\nT -> FT'\nT'-> *FT'/@\nF -> (E) / i\n";
    cout << "Enter the string to be checked: ";

    string input;
    cin >> input;

    RecursiveDescentParser parser(input);
    if (parser.parse()) {
        cout << "String is accepted\n";
    } else {
        cout << "String is not accepted\n";
    }

    return 0;
}
```

Output)

```
Recursive Descent Parsing For following grammar
E -> TE'
E'-> +TE'/@
T -> FT'
T'-> *FT'/@
F -> (E) / i
Enter the string to be checked: 2+3*a+4
String is accepted
```

```
Recursive Descent Parsing For following grammar
E -> TE'
E'-> +TE'/@
T -> FT'
T'-> *FT'/@
F -> (E) / i
Enter the string to be checked: a-b/c+2
String is not accepted
```