პროგრამირების აბსტრაქციები
სემინარის ამოცანები #6

1 ოპერაციები სიმრავლეებზე

დაწერეთ პროგრამა, რომელიც მომხმარებელს შეაყვანინებს ორ სიტყვას და გამოიტანს შემდეგ ინფორმაციას ამ სიტყვების შესახებ:

1. იმ ასოებს რომლებიც ორივე სიტყვაში გვხვდება
2. იმ ასოებს რომლებიც მხოლოდ პირველ სიტყვაში გვხვდება (და მეორეში არა)
3. იმ ასოებს რომლებიც მხოლოდ მეორე სიტყვაში გვხვდება

```
#include <iostream>
#include <string>
#include "console.h"
#include "simpio.h"
#include "set.h"
using namespace std;

Set<char> stringToSet(const string& s) {
    Set<char> result;
    for (int i = 0; i < s.length(); i++)
        result.insert(s[i]);
    return result;
}

void printCharSet(Set<char>& set) {
    foreach(char c in set)
        cout << c;
    cout << endl;
}

int main() {
    string s1, s2;
    s1 = getLine("Enter first string: ");
    s2 = getLine("Enter second string: ");

    Set<char> set1, set2;
    set1 = stringToSet(s1);
    set2 = stringToSet(s2);

    Set<char> intersection = set1 * set2;
    printCharSet(intersection);

    Set<char> difference1 = set1 - set2;
    printCharSet(difference1);

    Set<char> difference2 = set2 - set1;
    printCharSet(difference2);

    return 0;
}
```

2 სიტყვების დათვლა Map-ით

დაწერეთ პროგრამა, რომელიც ტექსტური ფაილიდან წაიკითხავს სიტყვებს, დათვლის თითოეული შეხვედრილი სიტყვა რამდენჯერ გამეორდა ტექსტში და ამ ინფორმაციას კონსოლში გამოიტანს.

```
#include <iostream>
#include <fstream>
#include <string>
#include "console.h"
#include "tokenscanner.h"
#include "map.h"
using namespace std;

int main() {
    ifstream inputFile;
    inputFile.open("input.txt");

    TokenScanner scanner(inputFile);
    scanner.ignoreWhitespace();

    Map<string, int> wordCount;

    while (scanner.hasMoreTokens()) {
        string nextWord = scanner.nextToken();

        if (wordCount.containsKey(nextWord)) {
            wordCount[nextWord]++;
        } else {
            wordCount[nextWord] = 1;
        }
    }

    inputFile.close();

    foreach (string word in wordCount) {
        cout << word << " - " << wordCount[word] << endl;
    }

    return 0;
}
```

3 შუაში შეხვედრის მეთოდი

ჩვენი მიზანია გავიგოთ მომხმარებლის შეყვანილი 2 სიტყვიდან შეგვიძლია თუ არა ერთ-ერთი სიტყვა მეორედ გადავაკეთოთ მაქსიმუმ 4 სიმბოლოების გაცვლის ოპერაციის გამოყენებით. ამისთვის გამოვიყენებთ ე.წ. შუაში შეხვედრის მეთოდს, ანუ ვიპოვით ყველა სიტყვას რომელიც პირველი სიტყვიდან მაქსიმუმ 2 ოპერაციით მიიღწევა და ყველა სიტყვას რომელიც მეორე სიტყვიდან მიიღწევა მაქსიმუმ 2 ოპერაციით. თუ ამ ორი სიმრავლის თანაკვეთა არ არის ცარიელი, მაშინ საწყისი ორი სიტყვა ერთმანეთისგან 4 ოპერაციით მიღწევადი ყოფილა.

```
#include <iostream>
#include <string>
#include "console.h"
#include "simpio.h"
#include "set.h"
using namespace std;

string swapCharsAtIndex(string s, int index) {
    string result = s;
    result[index] = s[index + 1];
    result[index + 1] = s[index];
    return result;
}

void extendSet(Set<string>& set) {
    Set<string> newWords;
    foreach (string word in set) {
        for (int i = 0; i < word.length() - 1; i++)
            newWords.insert(swapCharsAtIndex(word, i));
    }
    set += newWords;
}

void extendSetByN(Set<string>& set, int n) {
    for (int i = 0; i < n; i++)
        extendSet(set);
}

int main() {
    string s1, s2;
    s1 = getLine("Enter first string: ");
    s2 = getLine("Enter second string: ");

    Set<string> reachedFromS1, reachedFromS2;
    reachedFromS1.insert(s1);
    extendSetByN(reachedFromS1, 2);
    reachedFromS2.insert(s2);
    extendSetByN(reachedFromS2, 2);

    if ((reachedFromS1 * reachedFromS2).isEmpty()) {
        cout << "cannot be reached" << endl;
    else
        cout << "can be reached" << endl;
    return 0;
}
```