პროგრამირების აბსტრაქციები
სემინარის ამოცანები #12

1 ზურგჩანთის ჩალაგების ამოცანა

ამ ამოცანაში უნდა ვიპოვოთ მოცემული ტევადობის ზურგჩანთაში როგორ ჩავალაგოთ მაქსიმალური ღირებულების ნივთები. მოცემული გვექნება ნივთების სია, სადაც თითოეულ ნივთს აქვს წონა და ღირებულება და ასევე მოცემული გვექნება თუ რა მაქსიმალური წონის ნივთების ჩადება შეიძლება ზურგჩანთაში. დაწერეთ ფუნქცია, რომელიც იპოვის ამ პირობებში რა მაქსიმალური ღირებულების ნივთების ჩალაგება შეიძლება და ჩასალაგებელი ნივთების სიას მესამე პარამეტრად გადაცემულ ვექტორში ჩაწერს. ერთი ნივთის გამოყენება ორჯერ არ შეიძლება.

```
#include "vector.h"

struct itemT {
    int weight;
    int value;
};

int maxValueFromItems(Vector<itemT>& items,
                      int maxWeight,
                      Vector<itemT>& selectedItems) {
    if (items.isEmpty()) {
        selectedItems.clear();
        return 0;
    }

    itemT currentItem = items[items.size() - 1];
    items.remove(items.size() - 1);

    Vector<itemT> selectedItemsWithout;
    int maxValueWithout = maxValueFromItems(items,
                                            maxWeight,
                                            selectedItemsWithout);

    Vector<itemT> selectedItemsWith;
    int maxValueWith = maxValueFromItems(items,
                                         maxWeight - currentItem.weight,
                                         selectedItemsWith);
    maxValueWith += currentItem.value;
    selectedItemsWith.add(currentItem);

    items.add(currentItem);

    if (maxWeight < currentItem.weight
        || maxValueWithout > maxValueWith) {
        selectedItems = selectedItemsWithout;
        return maxValueWithout;
    } else {
        selectedItems = selectedItemsWith;
        return maxValueWith;
    }
}
```

2 მოგზაურის ამოცანა

ამ ამოცანაში მოცემული გვაქვს ცხრილი, რომელის i-ური სტრიქონის და j-ური სვეტის თანაკვეთაზე წერია ქალაქი i-დან ქალაქ j-ში წასვლა რა ჯდება. ჩვენ გვინდა რომ ყველა ქალაქი ზუსტად ერთხელ მოვინახულოთ (ბოლოს საწყის ქალაქში უნდა დავბრუნდეთ) და ეს გავაკეთოთ მინიმალური დანახარჯით. თქვენ უნდა დაწეროთ რეკურსიული ფუნქცია, რომელიც იპოვის რა მინიმალური თანხით შეიძლება ამის გაკეთება და ქალაქების მიმდევრობას, რომლითაც უნდა ვიმოგზაუროთ რომ ამ თანხაში ჩავეტიოთ, ცალკე გადაცემულ პარამეტრში ჩაწერს.

```
#include "grid.h"
#include "vector.h"
#include "set.h"

int minValueTravel(Grid<int>& costs,
                   Vector<int>& cities,
                   Set<int>& visited) {
    if (cities.size() == costs.numCols()) {
        int cost = 0;
        for (int i = 0; i < cities.size(); i++) {
            cost += costs[cities[i]][cities[(i + 1) % cities.size()]];
        }
        return cost;
    }

    int minCost = INT_MAX;
    for (int city = 0; city < costs.numCols(); city++) {
        if (!visited.contains(city)) {
            cities.add(city);
            visited.add(city);
            int currentCost = minValueTravel(costs, cities, visited);
            if (currentCost < minCost)
                minCost = currentCost;
            cities.remove(cities.size() - 1);
            visited.remove(city);
        }
    }
    return minCost;
}
```