Lab 03 (8%)
Topics: ADT Map

Problem 01 (1%)

Write a program using class map from C++ standard library which reads arbitrary sequence of words from standard input and after that prints all words and numbers of their occurrences in following format: <word>: <counter>. Words must be printed in lexicographical order. Give performance characteristics of map's operations.

Problem 02 (1%)

Write a program using classes map and set from C++ standard library which reads arbitrary sequence of lines of text from standard input and prints for each unique word from that text all numbers of lines this word appeared in increasing order. If word appeared more than once in some line you have to print this number only once. Words must be printed in lexicographical order.

Problem 03 (2%)

Write a program using class map from C++ standard library which manages list of caches in some imaginary computer game. Program has to work with following commands:

insert <x-coordinate> <y-coordinate> <item1> <item2> …

    Command adds new cache with specified coordinates and corresponding list of items.

erase <x-coordinate> <y-coordinate>

    Command removes cache with specified coordinates and prints “removed”.
    If such a cache does not exist program prints “does not exist”.

check <x-coordinate> <y-coordinate>

    If such a cache exists program has to output “found” and print all items of that cache.

Your program has to use type: map<Point, vector<string>, CmpPoints> where Point is a simple structure to store x, y coordinates of some cache, vector<string> represents list of items and CmpPoints is a criterion which class map has to use to compare Points.

Problem 4 (4%)

Solve Problem 01 using your own class MapStrInt. Your class has to use Binary Search Tree data structure and have following methods:

MapStrInt()
Constructor: creates an empty map

~MapStrInt()
Destructor.

int& operator[](const std::string& k):
Returns reference to integer which associates with the key k.

void printInOrder() const;
Prints the content of the map in the format
MapStrInt: <size>, { (k1, v1) (k2, v2) … }
Strings k1, k2, … must be in increasing lexicographical order.

void clear();
Removes all elements from the map.

int size() const;
Returns the number of elements in the map.