DSBA Introduction to Programming // Midterm Programming test

Spring semester 2020/21

Date: Mar 25, 2021

General Info

The test problem is organized as a set of related procedures (functions) that are needed to implement in order to earn points. The solution is checked by Yandex.Contest environment. Is requires you to upload a `solution.h` containing all the necessary *declarations* and *definitions* of types and functions created by you.

Your solution **must not** contain any definition of the `main()` method! The main method is pre-defined by us and consists of a set of tests procedures testing your solution in a unit-test-like manner. Some of these tests will be available for you in order to provide some hints what and how we are going to test.

Each subtask in the Contest checks only some of the methods. We explicitly state which methods must be provided in your solution and what is their interface. We give you exact prototype for each method and you must follow it, otherwise your solution will not be compatible with the testbed and will not compile.

We design the task (and, hence, the solution) so that all functions were in the interaction between each other (in order to provide the proper decomposition). For each task you are expected to solve it by adding a new mentioned function and manage its interaction with others. In your turn, you may expect that if you submit the solution that properly works at your side (with the mentioned restrictions, obviously; say, w/ no `main()` function in the `solution.h`), it also will not conflict with the stubs needed to call your code. However, *just in case*, be ready to adjust the submitted solution to only mentioned functions, commenting out all the rest code.

Tasks

There is a set of records about students, stored in a CSV file. Each student is represented by an object of a custom structure:

```
struct Student {
    std::string name;
    uint16_t group;
    std::vector<uint16_t> grades;
};
```

You don't modify the structure. If you need to store something in addition for solving the tasks below, consider any separate storage.

Task 1: Read data from CSV file [up to 0.35]

Create a function `readStudents()` loading data about students from a given stream associated with a CSV file. Function returns a collection of students as a `std::vector<Student>`.

All the needed details are provided in the skeleton code of `solution.h`. There is a predefined skeleton for the function. You are not allowed to modify it's header, however you are free to provide entirely your own implementation and, hence, remove any pre-defined code inside the body.

Consider proper decomposition, so the individual sub-tasks for this function are done by individual functions.

What is checked for this task?

Only `readStudents()` function. See `test1()` in `main()` for example.

What to submit to Yandex.Contest?

A correct program with the definition of this function and all decomposition functions and the necessary header files. There shouldn't be `main()`.

Task 2: overload `operator<<` [up to 0.15]

Overload `operator<<` for `std::ostream` and

1. `std::vector<Student>` datatype and
2. for `Student` datatype.

The former overload uses the latter one in its implementation.

For example and additional comments see the skeleton code of `solution.h`.

Consider proper decomposition!

What is checked for this task?

An ability to apply the following semantics:

```
Student s = ...
std::cout << s;

std::vector<Student> students;
std::cout << students;
```

See `test2()` in `main()` for example.

What to submit to Yandex.Contest?

A correct program with the definition of two operator functions and all decomposition functions (if needed) and the necessary header files. There shouldn't be `main()`.

Task 3: Calculate average per group [up to 0.35]

Create a function `calcGroupAverage()`, which obtains a vector of `Student` objects (collection of students) and calculates the average grade per group. The function returns the result of calculation as a mapping: *Group num to Average grade*.

For example and additional comments see the skeleton code of `solution.h`.

Consider proper decomposition!

What is checked for this task?

Only `calcGroupAverage()` function. See `test3()` in `main()` for example.

What to submit to Yandex.Contest?

A correct program with the definition of this function and all decomposition functions and the necessary header files. There shouldn't be `main()`.

Task 4: Expel weak student [up to 0.15]

Create a function `removeWeakStudents()`, which obtains a vector of `Student` objects (collection of students) and removes those ones who had at least one unsatisfactory grade (s.t. less than 4). The function returns the modified vector.

For example and additional comments see the skeleton code of `solution.h`.

Consider proper decomposition!

What is checked for this task?

Only `removeWeakStudents()` function. See `test4()` in `main()` for example.

What to submit to Yandex.Contest?

A correct program with the definition of this function and all decomposition functions and the necessary header files. There shouldn't be `main()`.