Zestaw 1A

Ocena:

ocena 3.0	Ocena 4.0	Ocena 5.0
Zadanie 1 lub Zadanie 2a	Zadanie 2a, Zadanie 2b	Zadanie 2 (a i b), Zadanie3

Zadanie 1:
Napisz funkcję sortującą nierosnąco N liczb całkowitych.
Należy zastosować algorytm sortowania przez prostą zamianę (inaczej bąbelkowe). Implementacja optymalna.

Zadanie 2:
Mając dane N elementów, z których każdy zawiera następujące informacje:

- nazwisko studenta,
- średnią ocen studenta.

a) Napisz funkcję wyszukującą wszystkich studentów ze średnią powyżej podane wartości (średnia podawana z klawiatury). Implementacja optymalna.
b) Napisz funkcję sortującą studentów alfabetycznie według nazwiska. Należy zastosować algorytm sortowania przez proste wybieranie. Implementacja optymalna.

Zadanie 3:
Napisz funkcję sortującą studentów (dane z Zadania 2) jednoczesne według dwóch kryteriów:

- nierosnąco według średnia ocen (kryterium podstawowe),
- alfabetycznie według nazwiska (kryterium dodatkowe).

Należy zastosować prosty algorytm sortowania, inny niż w Zadaniu 2. Implementacja optymalna.

Przykładowe dane do Zadania 2 i 3:

Katowicki	3.0
Nyski	4.5
Augustowski	4.5
Krakowski	5.0
Opolski	3.5
Wroclawski	4.0
Poznanski	4.0
Krakowski	4.5
Radomski	3.5
Katowicki	4.0

Wzorzec zawartości pliku programu:

```
#include <cstdlib>
#include <iostream>

using namespace std;

void zadanie1(){
  int const N = 10;
  int dane[N] = {7,3,5,1,9,3,8,2,6,4};

}

void zadanie2a(){
  int const N = 10;
  struct Student{
  };
  Student dane[N]={}; //tablicę należy zainicjalizować danymi podanymi w zadaniu

}

void zadanie2b(){
  int const N = 10;
  struct Student{
  };
  Student dane[N]={}; //tablicę należy zainicjalizować danymi podanymi w zadaniu

}

void zadanie3(){
  int const N = 10;
  struct Student{
  };
  Student dane[N]={}; //tablicę należy zainicjalizować danymi podanymi w zadaniu

}

int main(int argc, char *argv[])
{
   zadanie1();
   zadanie2a();
   zadanie2b();
   zadanie3();

   system("PAUSE");
   return EXIT_SUCCESS;
}
```