Instrukcja

W czasie pisania programu pamiętaj o:

1. dbaniu o czytelność kodu (odpowiednie formatowanie kodu, nazewnictwo zmiennych adekwatne do ich znaczenia, komentarze),
2. dbaniu o czytelność interfejsu z użytkownikiem (w sposób jawny pytaj użytkownika jakie dane ma podać oraz opisuj wyniki, które zwracasz),
3. przed fragmentem implementującym poszczególne zadania umieść komentarz: `/*Zadanie X */` oraz wypisz na ekranie analogiczny komunikat (X jest numerem zadania): `std::cout << "Zadanie X"<< std::endl;`,
4. każde zadanie umieść w oddzielnej funkcji (w niej dopiero należy odwoływać się do zaimplementowanych funkcji i klas),
5. zaimplementuj menu wyboru zadania, a następnie wykorzystując pętle **do–while** oraz konstrukcję **switch** wykonaj odpowiedni fragment kodu,
6. w zadaniach wymagających udzielenia komentarza bądź odpowiedzi, należy umieścić go w kodzie programu (np. w postaci komentarza albo wydrukować na ekranie),
7. w zadaniach polegających na zaprojektowaniu klasy należy utworzyć jej instancję i wykorzystać zaimplementowaną funkcjonalność.

Zadania

Zadanie 1

Wykorzystując tablicę `std::vector` z biblioteki standardowej STL zaimplementuj poniższą funkcjonalność:

a) wylosuj n liczb całkowitych z przedziału $\langle -20; 20 \rangle$ i umieść je w tablicy,
b) z wykorzystaniem zwykłego operatora indeksowania wyświetl całą zawartość tablicy na konsoli,
c) z wykorzystaniem iteratorów wyświetl całą zawartość tablicy na konsoli,
d) z wykorzystaniem iteratorów wyszukaj w tablicy wartości wskazanej przez użytkownika a następnie ją usuń (wyświetl ponownie zawartość tablicy),

Wskazówka Do generowania liczb losowych możesz wykorzystać następującą funkcję:

```
#include <random>

int randomInt(int min, int max) {
    static std::default_random_engine e{};
    std::uniform_int_distribution<int> d(min, max);
    return d(e);
}
```

Zadanie 2

Zaimplementuj funkcjonalność z zadania poprzedniego z wykorzystaniem `std::list` ze standardowej biblioteki.

Zadanie 3

Z wykorzystaniem algorytmu standardowego `std::find` przeimplementuj odpowiednie fragmenty z zadań poprzednich.

Zadanie 4

Wykorzystaj algorytmy standardowe `std::min_element` oraz `std::max_element` znajdź w kolekcjach z poprzednich zadań wartości największe i najmniejsze.

Zadanie 5

Wykorzystaj algorytm standardowy `sort`, odpowiednio `std::sort` lub `std::list::sort`, do posortowania obu kolekcji:

a) rosnąco,
b) malejąco, oraz
c) od największej wartości bezwzględnej do najmniejszej.

Zadanie 6*

Wykorzystując algorytm standardowy `std::count` zlicz wystąpienia każdej liczby w kolekcji.