# Wektory w C++:

<https://en.cppreference.com/w/cpp/container/vector>

Wektory są dynamicznie rozszerzalnymi tablicami w języku C++, które pozwalają przechowywać elementy tego samego typu danych w sekwencyjnej kolejności. Są one często używane, ponieważ oferują wiele zalet w porównaniu do tradycyjnych tablic, takich jak automatyczne zarządzanie pamięcią i elastyczność w rozmiarze.

Deklaracja wektora

#include <vector>

vector<int> vec; // deklaracja pustego wektora przechowującego elementy typu int

Dodawanie elementów do wektora

vec.push\_back(10); // dodaje element na końcu wektora

vec.push\_back(20);

Rozmiar wektora

int size = vec.size(); // zwraca liczbę elementów w wektorze

Dostęp do elementów wektora

int element = vec[0]; // dostęp do pierwszego elementu wektora

Iterowanie przez wektor

for (int i = 0; i < vec.size(); ++i) {

cout << vec[i] << " "; // wypisuje elementy wektora

}

Usuwanie elementu z wektora

vec.pop\_back(); // usuwa ostatni element wektora

Czyszczenie wektora

vec.clear(); // usuwa wszystkie elementy z wektora

Inicjalizacja wektora z określoną ilością elementów

vector<int> vec(5); // tworzy wektor o rozmiarze 5

Inicjalizacja wektora z określonymi wartościami

vector<int> vec = {1, 2, 3, 4, 5}; // tworzy wektor z określonymi wartościami

Iteratory

Wektory wspierają iteratory, które mogą być używane do przechodzenia przez elementy wektora.

vector<int>::iterator it;

for (it = vec.begin(); it != vec.end(); ++it) {

cout << \*it << " ";

}

**Zalety wektorów**

- Dynamiczne zarządzanie pamięcią: Wektory automatycznie dostosowują swoją wielkość, gdy dodawane są nowe elementy.

- Łatwe w użyciu: Wektory są prostsze w obsłudze niż tradycyjne tablice.

-Wsparcie dla wielu operacji: Wektory oferują szereg metod do wykonywania operacji na elementach, takich jak dodawanie, usuwanie i sortowanie.

**Zadanie**

1. Zdefiniuj klasę osoba:

- Utwórz plik nagłówkowy osoba.h.

- Zdefiniuj klasę osoba zawierającą prywatne pole nazwisko typu tablicowego.

- Dodaj konstruktor, który umożliwia inicjalizację obiektu nazwiskiem lub pozostawia pole puste.

- Zaimplementuj funkcje zaprzyjaźnione do przeciążenia operatorów << dla obiektów osoba i wskaźników na obiekty osoba.

2. Zdefiniuj szablon klasy wektor:

- Utwórz plik nagłówkowy wektor.h.

- Zdefiniuj szablon klasy wektor zawierający tablicę obiektów twoj\_typ oraz pola ile\_obiektow.

- Dodaj konstruktor klasy wektor, metody wstaw, usun, co\_na, rozsun oraz zsun.

* Wstaw: Ta funkcja służy do wstawiania nowego elementu do wektora.
* usun(int nr): Ta funkcja służy do usuwania elementu z wektora na podstawie podanego numeru indeksu nr.
* co\_na(int pozycja): Ta funkcja zwraca referencję do elementu znajdującego się na określonej pozycji pozycja w wektorze.
* rozsun(int pozycja): Ta funkcja służy do rozsuniecia elementów wektora, przesuwając elementy w prawo od danej pozycji pozycja.
* zsun(int nr): Ta funkcja służy do zsunięcia elementów wektora, przesuwając elementy w lewo od podanego numeru indeksu nr.

- Zaimplementuj funkcję zaprzyjaźnioną do przeciążenia operatora << dla obiektów wektor.

3. Stwórz plik źródłowy main.cpp:

- W tym pliku, zaimportuj pliki nagłówkowe osoba.h i wektor.h.

- Utwórz obiekty klasy osoba reprezentujące różnych muzyków, fizyków itp.

- Wykonaj operacje na obiektach klasy wektor, takie jak dodawanie, usuwanie i wyświetlanie zawartości.

4. Skompiluj i uruchom program:

- Skorzystaj z kompilatora C++, aby skompilować plik źródłowy main.cpp wraz z plikami nagłówkowymi.

- Uruchom skompilowany program i sprawdź, czy działa zgodnie z oczekiwaniami.

5. Testuj różne przypadki:

- Przetestuj działanie programu dla różnych typów danych, takich jak obiekty osoba, wskaźniki na osoba, liczby zmiennoprzecinkowe, znaki, itp.

- Upewnij się, że program obsługuje poprawnie operacje dodawania, usuwania i wyświetlania zawartości obiektów wektor.