# Lab 5: Odwracanie kolejności elementów

Napisz funkcję ***reverse***, która przyjmuje referencję wektora liczb całkowitych i odwraca kolejność jego elementów. Funkcja powinna być przystosowana do użycia w przykładowym programie poniżej.

Wykonanie:

Out: 2 9 -1 3 7

```
int main() {
	std::vector<int> vec {7, 3, -1, 9, 2};
	reverse(vec);
	for (int i = 0; i < vec.size(); ++i) {
		std::cout << vec[i] << " ";
	}
	std::cout << std::endl;

	return 0;
}
```

# Lab 5: Copy Negative: Elementy ujemne

Napisz funkcję ***copy_negative***, która przyjmuje stałą referencję wektora liczb całkowitych i zwraca wektor zawierający tylko jego ujemne elementy, w niezmienionej kolejności. Funkcja powinna być przystosowana do użycia w przykładowym programie poniżej.

```
int main() {
	std::vector<int> res = copy_negative(
		std::vector<int> {3, -1, 12, -5, 7, -10}
	);
	for (int i = 0; i < res.size(); ++i) {
	    std::cout << res[i] << " ";
	}
	std::cout << std::endl;

	return 0;
}
```

Wykonanie:

```
Out: -1 -5 -10
```

# Lab 5: Sortowanie bąbelkowe

Bąbelkowe sortowanie wektora przebiega następująco. Porównujemy pierwszy element z drugim i jeśli są w niewłaściwej kolejności, to zamieniamy je miejscami. Następnie porównujemy drugi element z trzecim i tak dalej, do końca wektora. Jeżeli w takim pojedynczym przebiegu musieliśmy wykonać choćby jedną zamianę, to powtarzamy wszystko od początku. W przeciwnym razie wektor jest już posortowany. Napisz funkcję ***bubble_sort***, która przyjmuje referencję wektora liczb całkowitych i sortuje go bąbelkowo w kolejności niemalejącej. Korzystając z tej funkcji napisz program, który czyta ze standardowego wejścia liczby całkowite do napotkania końca pliku i wypisuje je na standardowe wyjście w kolejności niemalejącej.

Przykładowe wykonanie:

In: 5 7 3 6 3 5 1 9 3 1

Out: 1 1 3 3 3 5 5 6 7 9