Laboratorium 3

Zadanie 1.
Zaprojektować i zaimplementować funkcje generujące pseudolosowe liczby:

- całkowite z wybranego przedziału,
- zmiennoprzecinkowe z wybranego przedziału.

Zaprojektować i zaimplementować funkcje wypełniające tablice jednowymiarowe z wykorzystaniem funkcji generujących liczby pseudolosowe.
Zaprojektować i zaimplementować funkcje wypisujące na ekran wypełnione tablice.
Zadanie 2.
Zaprojektować i zaimplementować funkcję realizującą mnożenie skalarne wektorów.
Zadanie 3.
Zaprojektować i zaimplementować funkcję realizującą sumowanie wektorów.

Zadanie 4.
Zaprojektować i zaimplementować funkcje:

- `wczytaj` zwracającą n-elementową tablicę wypełnioną liczbami całkowitymi wczytanymi od użytkownika.
- `wypelnij` zwracającą n-elementową tablicę wypełnioną pseudolosowymi liczbami całkowitymi z zadanego przedziału.
- `wyswietl` wypisującą na ekran wszystkie elementy tablicy oddzielone znakiem spacji.

Zdefiniowane funkcje przetestować podanym poniżej ciągiem instrukcji:

```
//...
int size = 10; int min = 7; int max = 23;
int * tab1 = wczytaj(size);
int * tab2 = wypelnij(size, min, max);
wyswietl(tab1,size);
wyswietl(tab2,size);
//...
```

W zadaniu należy pamiętać o zwolnieniu zasobów przydzielonych na stercie.
Zadanie 5.
Zaprojektować funkcje `minimum`, `maksimum` i `srednia` wyznaczającą odpowiednio minimalną, maksymalną i średnią wartość elementów tablicy.

Zdefiniowane funkcje przetestować podanym poniżej ciągiem instrukcji:

```
//...
const int size = 6;
int tab[size] = {1,3,8,5,10,6};
std::cout << "Min: " << minimum(tab,size)
          << "Max: " << maksimum(tab,size)
          << "Srednia: " << srednia(tab,size)
          << std::endl;
//...
// Min: 1 Max: 10 Srednia: 5.5
```

Zadanie 6.
Zaprojektować i zaimplementować funkcję `znajdz` poszukującą danej liczby w tablicy i zwracającą wartość logiczną określającą, czy liczba została znaleziona w tablicy czy nie.

Zdefiniowaną funkcję przetestować podanym poniżej ciągiem instrukcji:

```
//...
const int size = 4;
int tab[size] = {1,3,5,7};
std::cout << (znajdz(tab, size, 3)? "liczba 3 jest" : "liczby 3 nie ma")
          << "w tablicy" << std::endl;
std::cout << (znajdz(tab, size, 4)? "liczba 4 jest" : "liczby 4 nie ma")
          << "w tablicy" << std::endl;
//liczba 3 jest w tablicy
//liczby 4 nie ma w tablicy
```

Zadanie 7.
Zaprojektować i zaimplementować funkcję `porownaj` sprawdzającą kupon totolotka. Funkcja powinna przyjmować tablicę wylosowanych liczb oraz tablicę liczb "skreślonych" na kuponie, a zwracać liczbę trafień. Funkcję zaprojektować tak, aby możliwe było jej wywołanie dla dowolnej liczby losowanych liczb.

Zaimplementowane funkcje przetestować podanym poniżej ciągiem instrukcji:

```
//...
const int size = 6;
int kupon[size] = {1,3,5,7,9,11};
int los[size] = {1,4,5,8,9,12};
std::cout << "Liczba trafien to " << porownaj(los,kupon,size) << std::endl;
//Liczba trafien to: 3
```

Zadanie 8.
Zaprojektować i zaimplementować grę w totolotka. Należy zrealizować następujący schemat:

- wczytać od użytkownika liczbę kuponów, które chce wypełnić; przygotować odpowiedniej wielkości tablicę `wyniki`;
- wylosować zestaw 6 liczb;
- odpowiednią liczbę razy wczytywać od użytkownika zestaw sześciu liczb i porównywać je z wylosowanym zestawem;
- liczbę trafień zapisywać w tablicy wyniki; wyświetlić maksymalną, minimalną i średnią liczbę trafień;

Należy założyć, że w totolotku losujemy liczby z zakresu od 1 do 50.

W zadaniu należy pamiętać o zwolnieniu zasobów przydzielonych na stercie.

Zadanie 9.
Zaprojektować i zaimplementować funkcję zamieniającą nieujemną całkowitą liczbę zapisaną w systemie dziesiętnym na liczbę binarną (max. 32 bity).