Zadanie 1

Napisz szablon funkcji, która w podanej tablicy elementów dowolnego typu, dla którego ma sens porównywanie za pomocą operatora '<', wyszukuje i zwraca pozycję (indeks) największego elementu tablicy. Przetestuj funkcję dla tablic typu **int[]**, **double[]** i **string[]**.

Zadanie 2

Napisz i przetestuj funkcję pobierającą

1. tablicę obiektów typu **function<double(double)>** reprezentujących funkcje **double** → **double**,
2. jej wymiar,
3. dwie liczby typu **double** definiujące przedział $[a, b]$,
4. adres istniejącej zmiennej typu **double**.

Funkcja oblicza dla każdej z funkcji będących elementami tablicy jej maksymalną wartość na odcinku $[a, b]$ i zwraca tę z przekazanych w tablicy funkcji, dla której to maksimum wypadnie największe.
Aby znaleźć maksimum funkcji na odcinku, można „przejechać" przez ten odcinek z małym krokiem (np. $\epsilon = 10^{-5}$) i znajdować w każdym punkcie wartość funkcji.
Do zmiennej wskazywanej przez ostatni argument (pxmax) należy wpisać wartość argumentu (czyli „iksa"), dla którego znaleziona funkcja miała największą wartość.
W programie testowym można użyć własnych funkcji oraz przynajmniej jednej funkcji zadanej wyrażeniem lambda.

Na przykład program

download *FunArrayMax.cpp*

```
#include <functional>
#include <iostream>

double f1(double x) { return  x/4; }
double f2(double x) { return -2*x; }

using D2D = std::function<double(double)>;

D2D maxfun(D2D funs[], size_t size,
           double a, double b, double* pxmax) {
    static constexpr double eps = 1e-6;
    // ...
}

int main() {
    D2D funcs[]{
```

```
            f1,
            // lambda expression here
            f2
        };
        double xmax;
        D2D pf = maxfun(funcs, 3, 0, 3, &xmax);
        std::cout << "xmax = " << xmax << "; f(xmax) = "
                  << pf(xmax)  << std::endl;
    }
```

dla lambdy odpowiadającej funkcji $f(x) = x^2$ powinien wydrukować

```
xmax = 3; f(xmax) = 8.99999
```
