POPRO – lab 5
Funkcje i wprowadzenie do wskaźników

Sebastian Kozłowski, Daniel Paczesny,
Łukasz Dąbała, Krystian Ignasiak

1 Tematyka

Laboratorium 5 z przedmiotu POPRO dotyczy implementowania własnych funkcji, deklarowania i definiowania zmiennych, w tym zmiennych globalnych, oraz przekazywania zmiennych do funkcji przez wartość lub przez wskaźnik.

2 Przygotowanie do zajęć laboratoryjnych

W ramach przygotowania do laboratorium warto zapoznać się z materiałami z zakresu implementowania funkcji, tworzenia zmiennych oraz podstawowego wykorzystania wskaźników.

1. https://beginnersbook.com/2014/01/c-functions-examples/
2. https://beginnersbook.com/2014/01/c-pointers/
3. https://beginnersbook.com/2014/01/c-passing-pointers-to-functions/

3 Sprawozdanie

Sprawozdanie stanowią pliki *.c* z rozwiązaniami zadań, po jednym dla każdego zadania. Pliki powinny zostać umieszczone w repozytorium. Komentarze wymagane w zadaniu nr 2 powinny znaleźć się w funkcji *main*.

4 Przebieg laboratorium

W trakcie laboratorium należy napisać sprawdzian wejściowy (wejściówka) wg. wskazówek prowadzącego oraz wykonać trzy zadania. Wejściówka, zadanie pierwsze i drugie warte są 1 punkt każde, zadanie trzecie 2 punkty.

4.1 Zadanie 1 – Implementowanie własnych funkcji (1 pkt.)

Wzór Herona na pole dowolnego trójkąta ma postać:

$$S = \sqrt{p(p-a)(p-b)(p-c)} \tag{1}$$

gdzie p oznacza **połowę** obwodu, zaś a, b i c to długości boków tego trójkąta. Zaimplementuj program obliczający pole trójkąta zdefiniowanego przez współrzędne trzech punktów płaszczyzny. Wyniki obliczeń wyprowadź na standardowe wyjście. Program powinien obejmować funkcje:

- **triSide**, przyjmującą współrzędne dwóch punktów płaszczyzny a i b w kolejności a_x, a_y, b_x, b_y i zwracającą długość odcinka łączącego te punkty;
- **triPeri**, przyjmującą długości trzech boków i zwracającą obwód trójkąta;
- **triArea**, przyjmującą współrzędne trzech punktów tworzących wierzchołki trójkąta i zwracającą pole trójkąta; ta funkcja powinna korzystać z obu powyższych funkcji do zrealizowania swojego zadania obliczeniowego.

Uwaga: do obliczania pierwiastka kwadratowego można użyć funkcji *sqrt* zdefiniowanej w pliku nagłówkowym *math.h.*

4.2 Zadanie 2 – Deklarowanie i definiowanie zmiennych, w tym zmiennych globalnych (1 pkt.)

Przeanalizuj i skomentuj komunikaty wyprowadzanie na standardowe wyjście na skutek działania załączonego kodu. Szczególną uwagę zwróć na:

- *W bloku funkcji zmienna a=...*
- *Zmienna a po wywolaniu zwieksz_liczbe=...*

```c
#include <stdio.h>

int a = 8;

void info(char *nazwa_bloku, int zmienna) {
        printf("W bloku %s zmienna a = %d\n", nazwa_bloku, zmienna);
}

void funkcja() {
        info("funkcji", a);
}

int suma(int x, int y) {
        return x + y;
}
```

Projekt „NERW 2 PW. Nauka – Edukacja – Rozwój – Współpraca" współfinansowany jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

Fundusze Europejskie Wiedza Edukacja Rozwój
Rzeczpospolita Polska
Politechnika Warszawska
Unia Europejska Europejski Fundusz Społeczny

```
void czy_parzysta(const int);

int liczba_zwiekszona(int x) {
        return ++x;
}

void zwieksz_liczbe(int a) {
        ++a;
}

void zwieksz_przez_adres(int* x) {
        (*x)++;
}

int main() {
        int a = 20;

        info("main", a);
        funkcja();

        int b = 9;
        int wynik = suma(a, b);

        printf("\na + b = %d\n\n", wynik);

        czy_parzysta(a);
        czy_parzysta(b);

        printf("\na + 1 = %d\n", liczba_zwiekszona(a));
        zwieksz_liczbe(a);
        printf("Zmienna 'a' po wywolaniu zwieksz_liczbe = %d\n", a);

        zwieksz_przez_adres(&a);
        printf("Zmienna 'a' po zwieksz_przez_adres = %d\n", a);

        return 0;
}

void czy_parzysta(const int x) {
        if(x % 2 == 0)
                printf("Liczba %d jest parzysta!\n", x);
        else
                printf("Liczba %d jest nieparzysta!\n", x);
}
```

Projekt „NERW 2 PW. Nauka – Edukacja – Rozwój – Współpraca” współfinansowany jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

4.3 Zadanie 3 – Przekazywanie argumentów do funkcji za pomocą wskaźników (2 pkt.)

Napisz program wyznaczający pierwiastki równania kwadratowego wraz z podaniem informacji o typie tego równanie (np. sprzeczne). Program powinien analizować również przypadki szczególne w postaci równania liniowego i tożsamości. Wykorzystaj poniższy kod.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

//Funkcja "pierwiastki" zwraca liczbę pierwiastków równania kwadratowego
//postaci a*x^2 + b*x + c=0
//oraz wpisuje wartości tych pierwiastków do zmiennych
//wskazywanych przez wskaźniki wsk_x1 i wsk_x2.
//
//Uwaga!
//W przypadku równania o jednym rozwiązaniu (funkcja zwraca 1)
//do zmiennych wskazywanych przez wsk_x1 i wsk_x2 ma być wpisana ta sama wartość.
//W przypadku równania, które nie ma rozwiązań (funkcja zwraca 0),
//zmienne wskazywane przez wsk_x1 i wsk_x2 nie powinny być modyfikowane.
int pierwiastki(double a, double b, double c, double *wsk_x1, double *wsk_x2) {

        //implementacja funkcji

}

/*
        funkcja main wyświetla liczbę i wartość pierwiastków
        oraz informcję o rodzaju wielomianu
*/
int main( void ) {
    double a, b, c, x1, x2;
    int liczba;

        //implementacja funkcji

    return 0;
}
```

Projekt „NERW 2 PW. Nauka – Edukacja – Rozwój – Współpraca” współfinansowany jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego