Wojskowa Akademia Techniczna
Wydział Elektroniki
Instytut Telekomunikacji

Studia I°

Programowanie w języku C cz.2

Materiały pomocnicze do zajęć

dr inż. Jarosław Krygier

Warszawa 2017

Spis treści

1 Wskaźniki do funkcji

```
/*
 ============================================================================
 Name        : W_PRC2.c
 Author      : Krygier
 Version     :
 Copyright   : Tylko na potrzeby zajęć
 Description : Wyklady C cz.2, Wskazniki na funkcje
 ============================================================================
 */

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

//fukcja dodawnia
int dodawanie(int x, int y) {
	return (x+y);
}

//funkcja odejmowania
int odejmowanie (int x, int y) {
	return (x-y);
}

int operacja (int (*) (int, int), int, int); // definicja ponizej fukcji main()

int main(void) {
	puts("Wyklad1"); /* Wyklad1 PRC2*/
	//Zaawansowane operacje na wskaznikach

	int a;
	int b=5;
	int c=6;
	int* wsk;

	////////////////////
	//wskazniki
	////////////////////

	wsk = &a;
	printf ("Adres zmiennej 'a': 0x%x\n", (unsigned int) wsk);  // rzutowanie na unsigned int,
poniewaz printf wymaga takiego typu (nie ma warningow)
	wsk = &b;
	printf ("Adres zmiennej 'b': 0x%x\n", (unsigned int) wsk);
	wsk = &c;
	printf ("Adres zmiennej 'c': 0x%x\n", (unsigned int) wsk);

	////////////////////
	//wskazniki do funkcji
	////////////////////
	//deklaracja
	int (*funkcja) (int, int);
	//jaka jest roznica z int *funkcja (int, int);

	funkcja = &dodawanie;
	// teraz wskaznik wskazuje na fukncje 'dodawanie'
	printf ("1) Wskaznik na funkcje 'dodawanie': 0x%x\n",(unsigned int) funkcja);

	//lub alternatywnie
	funkcja = dodawanie;
	// teraz wskaznik wskazuje na fukncje 'dodawanie'
	printf ("2) Wskaznik na funkcje 'dodawanie': 0x%x\n", (unsigned int) funkcja);
```

```
        //Wywolanie funkcji przez wskanik
        a = funkcja (b,c);
        printf ("1) Wynik: %d \n", a);
        // lub alternatywnie
        a = (*funkcja) (b,c);
        printf ("2) Wynik: %d \n", a);

        // przypisanie do wskazniaka roznych funkcji
        //teraz ten sam wskaznik wskazuje na inna funkcje
        funkcja = odejmowanie; //mozna przypisac funkcje bez &
        printf ("3) Wskaznik na funkcje 'odejmowanie': 0x%x\n", (unsigned int) funkcja);
        //wywolanie jest identyczne, ale wynik rozny
        a = (*funkcja) (b,c);
        printf ("3) Wynik: %d \n", a);

        ////////////////////
        //tablica wskaznikow do funkcji
        ////////////////////

        // deklaracja tablicy wskaznikow do fukncji
        int (*tablica_wskaznikow_do_funkcji [3]) (int, int);

        //przypisanie adresow fukcji do komorek tablicy
        tablica_wskaznikow_do_funkcji [0] = dodawanie;
        tablica_wskaznikow_do_funkcji [1] = odejmowanie;
        tablica_wskaznikow_do_funkcji [2] = NULL;

        printf ("Wskaznik na funkcje 'dodawanie' [0]: 0x%x\n", (unsigned int)
tablica_wskaznikow_do_funkcji [0]);
        printf ("Wskaznik na funkcje 'odejmowanie'[1]: 0x%x\n", (unsigned int)
tablica_wskaznikow_do_funkcji [1]);
        printf ("Wskaznik na funkcje 'NULL'[2]: 0x%x\n", (unsigned int)
tablica_wskaznikow_do_funkcji [2]);

        //wywolanie funkcji z tablicy

        a = (*tablica_wskaznikow_do_funkcji [0]) (b,c);
        printf ("1) Wynik: %d \n", a);
        a = (*tablica_wskaznikow_do_funkcji [1]) (b,c);
        printf ("2) Wynik: %d \n", a);

        //lub alternatywnie
        a = tablica_wskaznikow_do_funkcji [0] (b,c);
        printf ("1) Wynik: %d \n", a);
        a = tablica_wskaznikow_do_funkcji [1] (b,c);
        printf ("2) Wynik: %d \n", a);

        /*UWAGA!!!*/
        /*
         * Wskaniki do fukcji mozna:
        - przekazywac do innych fukcji jako argumenty
        - zwracac jako wynik dzialania fukcji
        - porownywac z NULL
        * Nie mozma
        - wykonywac operacji matematycznych
        */

        //Przekazanie wskaznika do fukcji jako argumentu innej fukcji
        funkcja = dodawanie;
        a = operacja (funkcja, 3, 5);
        printf ("1) Wynik: %d \n", a);
        //lub bezposrednio nazwa fukcji jest tez wskaznikiem
        a = operacja (dodawanie, 3, 5);
        printf ("2) Wynik: %d \n", a);

        // a teraz przekazujemy wskznik innej fukcji
```

```
        a = operacja (odejmowanie, 3, 5);
        printf ("3) Wynik: %d \n", a);

        //sprawdzenie
        //Napisac fukcje 'operacja_arytm', ktora wykonuje y=(a/b)+c
        //Napisac fukcje o nazwie 'operacje_new' z argumentami (typ_operacji, arg1, arg2, arg3)
zwracajaca wynik:
        // dodawania lub odejmowania, lub wynik fukcji 'operacja_arytm' w zaleznosci od
przekazanego typu operacji



        return EXIT_SUCCESS;
}

// fukcja z przekazanym wskaznikiem do fukcji
int operacja (int (*wsk_do_funkcji_przekazany) (int, int), int a, int b){ // jako pierwsza bedzie
przekazywany wskaznik do fukcji
        int wynik;

        wynik = wsk_do_funkcji_przekazany (a,b);
        // lub
        //wynik = (*wsk_do_funkcji_przekazany) (a,b);

        return wynik;
}
```

2 Biblioteki dynamiczne

2.1 Plik: biblioteki_dyn_test.c

```
/*
 ============================================================================
 Name        : biblioteki_dyn_test.c
 Author      : Krygier
 Version     : 1.0
 Copyright   : Tylko na potrzeby zajęć
 Description : Biblioteki dynamiczne
 ============================================================================
 */
//Utworzenie biblioteki dynamicznej:
//gcc -shared biblioteka.c -o biblioteka.so

/*
Aby mieć ‘dynamic library’ (dl), linker musi łączyć z opcja -l: dl
Ustawić w Eclipsie: C/C++ -> Settings ->GCC C Linker -> Libraries
Program może ładować w trakcie pracy potrzebne mu biblioteki. Linker dostarcza w tym celu 4
funkcji:
    dlopen ładowanie biblioteki,
    dlsym zwrócenie wskaźnika do odpowiedniego symbolu (funkcji) w bibliotece,
    dlerror obsługa błędów,
    dlclose zamykanie biblioteki.
Funkcje te są udostępniane poprzez nagłówek dlfcn.h (trzeba go dodać do głównego programu).

#include <dlfcn.h>
void *dlopen(const char *filename, int flag);
char *dlerror(void);
void *dlsym(void *handle, const char *symbol);
int dlclose(void *handle);
*/

#include <stdio.h>
#include <stdlib.h>
#include <dlfcn.h>


int main(void) {
	puts("Dolaczanie bibliotek dynamicznych");
	puts("=================================");

	int status_wyjscia; // do dlclose

	void  *Biblioteka; // wskaznik do bilbioteki
	int   (*Funkcja)(int, int); //wskzanik do fukcji
	double   (*Funkcja1)(int, int); //wskzanik do fukcji1


	Biblioteka = dlopen("./biblioteka.so", RTLD_LAZY); // otwarcie biblioteki RTLD_LAZY -
wskazniki na funkcie wewnetrzne sa budowane dopiero w trakcie wywolania dlsym
	if(!Biblioteka) {
		printf ("Error otwarcia: %s\n", dlerror());
		return(1);
	}
	Funkcja = dlsym(Biblioteka, "suma"); // pobranie wskznika na odpowiednia funckje podana za
pomoca nazwy

	printf ("suma:%d \n", Funkcja (4, 2));

	Funkcja = dlsym(Biblioteka, "iloczyn");

	printf ("iloczyn:%d \n", Funkcja (4, 2));

	Funkcja = dlsym(Biblioteka, "roznica");
```

```
        if (Funkcja== NULL) {
                printf ("Blad wywolania funkcji 'roznica()': brak w bibliotece\n");
        } else {
                printf ("roznica:%d \n", Funkcja (8, 2));
        }

        status_wyjscia = dlclose (Biblioteka); // zamkniecie biblioteki - > jesli poprawnie to
status_wyjscia = 0
        if(status_wyjscia) { // jesli 0 (falsz) -> dlopen() zamknela biblioteke poprawnie
                printf ("Error zamkniecia: %s\n", dlerror());
                return(1);
        }

        //===================
        Biblioteka = dlopen("./biblioteka.so", RTLD_NOW); //RTLD_NOW - wskazniki na funkcie
wewnetrzne sa tu budowane potem dlsym je wykorzystuje
                if(!Biblioteka) {
                        printf ("Error otwarcia: %s\n", dlerror());
                        return(1);
                }
                Funkcja1 = (double (*)(int, int)) dlsym(Biblioteka, "iloraz"); //z podpowidzia
jaki wskaznik do jakiej funckji bedzie zwracany

                printf ("iloraz:%f \n", Funkcja1 (5, 3));

                status_wyjscia = dlclose (Biblioteka);
                if(status_wyjscia) { // jesli 0 (falsz) -> dlopen() zamknela biblioteke poprawnie
                        printf ("Error zamkniecia: %s\n", dlerror());
                        return(1);
                }

        return EXIT_SUCCESS;
}
```

2.2 Plik: biblioteka.c

```
/*
 * biblioteka.c
 * Author: Jaroslaw Krygier
 * Tylko na potrzeby zajęć
 */

//Utworzenie biblioteki dynamicznej:
//gcc -shared biblioteka.c -o biblioteka.so


#include <stdio.h>

int suma (int a, int b){
        return (a+b);
}

int iloczyn (int a, int b){
        return (a*b);
}

double iloraz (int a, int b){
        return ((double)a / (double) b);
}

/*int roznica (int a, int b){
        return (a-b);
}*/
```