| Imię i nazwisko<br><br>Marek Kubicki | Kierunek<br><br>Informatyka techniczna ITE | Rok studiów i grupa<br><br>1 rok  grupa 5 |
|---|---|---|
| Data zajęć<br><br>15.12.2022 | Numer i temat sprawozdania<br><br>Lab 10. – Struktury | |

Cel:

- Opanowanie podstaw tworzenia i wykorzystania struktur w C

Przebieg zajęć:

Utworzyłem katalog roboczy lab_10 i skopiowałem do niego pliki struktury_szablon.c , a następnie zacząłem wykonywać polecenia zawarte w tym pliku.

```
sh-4.2$ pwd
/home/METAL/markubi2/lab_10
sh-4.2$ ls
skr1  struktury_szablon.c
```

Zacząłem od stworzenia dwóch struktur (w dalszej części ćwiczenia używałem tylko jednej z nich), jedną za pomocą funkcjonalności typedef a drugą bez niej

```
struct zwierze_bez_t
{
  char* nazwa;
  int ilosc;
  double sr_waga;
};
typedef struct
{
  char* nazwa;
  int ilosc;
  double sr_waga;
} zwierze;
```

Następnie zadeklarowałem trzy funkcje potrzebne w dalszej części zadania

```
void fun_strukt(zwierze obiekt_1);
zwierze fun_strukt_out(zwierze obiekt_1);
void fun_strukt_wsk(zwierze* obiekt_2_wsk);
```

Zacząłem wykonywać polecenia zawarte w int main(void)
Na początku zadeklarowałem zmienną obiekt_1 która należała do stworzonej przeze mnie struktury zwierzeta , nadałem jej pewne wartości i wypisałem je za pomocą polecenia printf()

```
  zwierze obiekt_1;
  obiekt_1.nazwa = "koala pospolita";
  obiekt_1.ilosc = 1234;
  obiekt_1.sr_waga = 63.48;
  printf("Początkowe wartości pól obiekt_1: \nnazwa: %s\nilosc: %d\nsr_waga:
%lfkg\n", obiekt_1.nazwa, obiekt_1.ilosc, obiekt_1.sr_waga);
```

```
Początkowe wartości pól obiekt_1:
nazwa: koala pospolita
ilosc: 1234
sr_waga: 63.480000kg
```

Kolejne polecenie wymaga użycia jednej z zadeklarowanych prze zemnie funkcji a następnie wypisanie w niej i po jej wykonaniu wartości zmiennej obiekt_1

Działanie funkcji:

```
void fun_strukt(zwierze obiekt_1)
{
  printf("\n\nWewnątrz fun_strukt");
  printf("Wewnątrz fun_strukt - wartości pól obiekt_1: \nnazwa: %s\nilosc:
%d\nsr_waga: %lfkg\n", obiekt_1.nazwa, obiekt_1.ilosc, obiekt_1.sr_waga);
  obiekt_1.nazwa = "nie koala";
  obiekt_1.ilosc = 4321;
  obiekt_1.sr_waga = 48.63;
  printf("Wewnątrz fun_strukt - zmodyfikowane wartości pól obiektu argumentu:
\nnazwa: %s\nilosc: %d\nsr_waga: %lfkg\n", obiekt_1.nazwa, obiekt_1.ilosc,
obiekt_1.sr_waga);
}
```

```
Wewnątrz fun_struktWewnątrz fun_strukt - wartości pól obiekt_1:
nazwa: koala pospolita
ilosc: 1234
sr_waga: 63.480000kg
Wewnątrz fun_strukt - zmodyfikowane wartości pól obiektu argumentu:
nazwa: nie koala
ilosc: 4321
sr_waga: 48.630000kg
```

W main:

```
fun_strukt(obiekt_1);
  printf("\n\nPo wywołaniu fun_strukt");
  printf(" - wartości pól obiekt_1: \nnazwa: %s\nilosc: %d\nsr_waga: %lfkg\n",
obiekt_1.nazwa, obiekt_1.ilosc, obiekt_1.sr_waga);
```

```
Po wywołaniu fun_strukt - wartości pól obiekt_1:
nazwa: koala pospolita
ilosc: 1234
sr_waga: 63.480000kg
```

Kolejne polecenie jest podobne ale tym razem funkcja zwraca wartość zmiennej

Działanie funkcji:

```
zwierze fun_strukt_out(zwierze obiekt_1)
{
  printf("\n\nWewnątrz fun_strukt_out");
  printf("Wewnątrz fun_strukt_out - wartości pól obiekt_1: \nnazwa: %s\nilosc:
%d\nsr_waga: %lfkg\n", obiekt_1.nazwa, obiekt_1.ilosc, obiekt_1.sr_waga);
  obiekt_1.nazwa = "nie koala";
  obiekt_1.ilosc = 4321;
  obiekt_1.sr_waga = 48.63;
  printf("Wewnątrz fun_strukt_out - zmodyfikowane wartości pól obiektu argu-
mentu: \nnazwa: %s\nilosc: %d\nsr_waga: %lfkg\n", obiekt_1.nazwa,
obiekt_1.ilosc, obiekt_1.sr_waga);
return obiekt_1;
}
```

```
Wewnątrz fun_strukt_outWewnątrz fun_strukt_out - wartości pól obiekt_1:
nazwa: koala pospolita
ilosc: 1234
sr_waga: 63.480000kg
Wewnątrz fun_strukt_out - zmodyfikowane wartości pól obiektu argumentu:
nazwa: nie koala
ilosc: 4321
sr_waga: 48.630000kg
```

W main:

```
obiekt_1 = fun_strukt_out(obiekt_1);
  printf("\n\nPo wywołaniu fun_strukt_out i przypisaniu wyniku do
obiekt_1\n");
  printf(" - wartości pól obiekt_1: \nnazwa: %s\nilosc: %d\nsr_waga: %lfkg\n",
obiekt_1.nazwa, obiekt_1.ilosc, obiekt_1.sr_waga);
```

```
Po wywołaniu fun_strukt_out i przypisaniu wyniku do obiekt_1
 - wartości pól obiekt_1:
nazwa: nie koala
ilosc: 4321
sr_waga: 48.630000kg
```

Następne polecenie polega na zadeklarowaniu kolejnej zmiennej i wykonaniu podobnych działń jak w poprzednim poleceniu, lecz tym razem przy pomocy wskaźników.

W main:

```
  zwierze obiekt_2 = { "wonz", 3042, 12.21};
  printf("\n\nPoczątkowe wartości pól obiekt_2: \nnazwa: %s\nilosc:
%d\nsr_waga: %lfkg\n",  obiekt_2_wsk->nazwa, obiekt_2_wsk->ilosc,
obiekt_2_wsk->sr_waga);
  fun_strukt_wsk(obiekt_2_wsk);
```

```
Początkowe wartości pól obiekt_2:
nazwa: wonz
ilosc: 3042
sr_waga: 12.210000kg
```

Działanie funkcji:

```
void fun_strukt_wsk(zwierze* obiekt_2_wsk)
{
  printf("\n\nWewnątrz fun_strukt_out - wartości pól obiekt_2: \nnazwa: %s\nilosc: %d\nsr_waga: %lfkg\n",  obiekt_2_wsk->nazwa, obiekt_2_wsk->ilosc, obiekt_2_wsk->sr_waga);
  obiekt_2_wsk->nazwa= "rzeczny";
  obiekt_2_wsk->ilosc = 101;
  obiekt_2_wsk->sr_waga = 45.45;
  printf("Wewnątrz fun_strukt_out - wartości pól obiekt_2 po modyfikacji: \nnazwa: %s\nilosc: %d\nsr_waga: %lfkg\n",  obiekt_2_wsk->nazwa, obiekt_2_wsk->ilosc, obiekt_2_wsk->sr_waga);
}
```

```
Wewnątrz fun_strukt_out - wartości pól obiekt_2:
nazwa: wonz
ilosc: 3042
sr_waga: 12.210000kg
Wewnątrz fun_strukt_out - wartości pól obiekt_2 po modyfikacji:
nazwa: rzeczny
ilosc: 101
sr_waga: 45.450000kg
```

W main:

```
printf("\n\nPo wywołaniu fun_strukt_wsk wartości pól obiekt_2: \nnazwa: %s\nilosc: %d\nsr_waga: %lfkg\n",  obiekt_2_wsk->nazwa, obiekt_2_wsk->ilosc, obiekt_2_wsk->sr_waga);
```

```
Po wywołaniu fun_strukt_wsk wartości pól obiekt_2:
nazwa: rzeczny
ilosc: 101
sr_waga: 45.450000kg
```

W następnym poleceniu należy skopiować dane zmiennej strukturalnej do drugiej.

```
zwierze obiekt_3 = { obiekt_2.nazwa, obiekt_2.ilosc, obiekt_2.sr_waga};
  printf("\n\nPoczątkowe wartości pól obiekt_3: \nnazwa: %s\nilosc: %d\nsr_waga: %lfkg\n", obiekt_3.nazwa, obiekt_3.ilosc, obiekt_3.sr_waga);
```

```
Początkowe wartości pól obiekt_3:
nazwa: rzeczny
ilosc: 101
sr_waga: 45.450000kg
```

Wnioski

Struktury pozwalają na łatwe uporządkowanie danych oraz na przekazywanie wielu wartości do i z zmiennych, jednocześnie nie zmniejszając możliwości manipulacji wartościami w porównaniu do normalnych zmiennych.