Informatyka 2 (EZ1E3012)

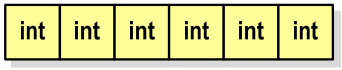
Politechnika Białostocka - Wydział Elektryczny
Elektrotechnika, semestr III, studia niestacjonarne I stopnia
Rok akademicki 2020/2021

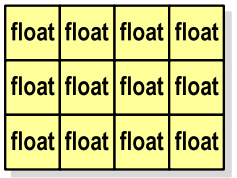
Pracownia nr 3 (24.10.2020)

dr inż. Jarosław Forenc

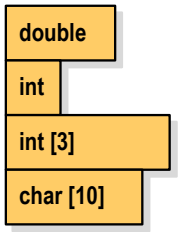
Struktury w języku C

- Tablica - ciągły obszar pamięci zawierający elementy tego samego typu





- Struktura - zestaw elementów różnych typów, zgrupowanych pod jedną nazwą



Deklaracja struktury

```
struct nazwa
{
    opis_pola_1;
    opis_pola_2;
    ...
    opis pola_n;
};
```

```
struct punkt
{
    int x;
    int y;
};
```

- Elementy struktury to pola (dane, komponenty, składowe) struktury
- Deklaracje pól mają taką samą postać jak deklaracje zmiennych
- Deklarując strukturę tworzymy nowy typ danych (struct punkt), którym można posługiwać się tak samo jak każdym innym typem standardowym

Deklaracja struktury

```
struct osoba
{
    char imie[15];
    char nazwisko[20];
    int wiek, waga;
};
```

```
struct zesp
{
    float Re, Im;
};
```

- Deklaracja struktury nie tworzy obiektu (nie przydziela pamięci na pola struktury)
- Zapisanie danych do struktury wymaga zdefiniowania zmiennej strukturalnej

Deklaracja zmiennej strukturalnej

```
struct osoba
{
    char imie[15];
    char nazwisko[20];
    int wiek, waga;
} Kowal, Nowak;
```

- Kowal, Nowak
  - zmienne strukturalne typu struct osoba

Deklaracja zmiennej strukturalnej

```
#include <stdio.h>

struct osoba
{
    char imie[15];
    char nazwisko[20];
    int wiek, waga;
};

int main(void)
{
    struct osoba Kowal;
    struct osoba Nowak;
    ...
    return 0;
}
```

Odwołania do pól struktury

- Dostęp do pól struktury możliwy jest dzięki konstrukcji typu:

```
nazwa_struktury.nazwa_pola
```

- Operator . nazywany jest operatorem bezpośredniego wyboru pola
- Zapisanie wartości 25 do pola wiek zmiennej Nowak ma postać

```
Nowak.wiek = 25;
```

- Wyrażenie Nowak.wiek traktowane jest jak zmienna typu int

```
printf("Wiek: %d\n", Nowak.wiek);
scanf("%d", &Nowak.wiek);
```


Odwołania do pól struktury

- Dostęp do pól struktury możliwy jest dzięki konstrukcji typu:

```
nazwa_struktury.nazwa_pola
```

- Operator . nazywany jest operatorem bezpośredniego wyboru pola
- Zapisanie wartości Jan do pola imie zmiennej Nowak ma postać

```
strcpy(Nowak.imie,"Jan");
```

- Wyrażenie Nowak.imie traktowane jest jak łańcuch znaków

```
printf("Imie: %s\n", Nowak.imie);
gets(Nowak.imie);
```

Struktury - przykład (osoba)

```
#include <stdio.h>

struct osoba
{
    char imie[15];
    char nazwisko[20];
    int  wiek;
};

int main(void)
{
     struct osoba Nowak;
```


Struktury - przykład (osoba)

```
     printf("Imie:     ");
     gets(Nowak.imie);

     printf("Nazwisko: ");
     gets(Nowak.nazwisko);

     printf("Wiek:     ");
     scanf("%d",&Nowak.wiek);

     printf("%s %s, wiek: %d\n", Nowak.imie,
            Nowak.nazwisko, Nowak.wiek);

     return 0;
}
```

```
Imie:     Jan
Nazwisko: Nowak
Wiek:     22
Jan Nowak, wiek: 22
```


Struktury - przykład (miernik)

```
#include <stdio.h>

struct miernik
{
   double k;   // klasa dokładności
   int d;      // liczba działek podziałki
   double Zp;  // zakres pomiarowy
};

int main(void)
{
   // Amperomierz LE-3P
   struct miernik LE3P = {0.5, 75, 12.0};
   double Dpm, p;
```


Struktury - przykład (miernik)

```
   printf("Amperomierz analogowy LE-3P\n");
   printf("Zakres pomiarowy: %g A\n",LE3P.Zp);
   printf("Liczba dzialek podzialki: %d\n",LE3P.d);
   printf("Klasa dokladnosci: %g\n",LE3P.k);
   printf("-------------------------------------\n");

   printf("Bezwzgledny maksymalny blad pomiaru:\n");
   p = 0.2;
   Dpm = LE3P.Zp*(LE3P.k/100+p/LE3P.d);
   printf("* dla p = %g, Dpm = %g A\n",p,Dpm);

   p = 0.5;
   Dpm = LE3P.Zp*(LE3P.k/100+p/LE3P.d);
   printf("* dla p = %g, Dpm = %g A\n",p,Dpm);

   return 0;
}
```

Struktury - przykład (miernik)

```
    printf("Amperomi
    printf("Zakres p
    printf("Liczba d
    printf("Klasa do
    printf("--------
    printf("Bezwzgle
    p = 0.2;
    Dpm = LE3P.Zp*(L
    printf("* dla p = %g, Dpm = %g A\n",p,Dpm);

    p = 0.5;
    Dpm = LE3P.Zp*(LE3P.k/100+p/LE3P.d);
    printf("* dla p = %g, Dpm = %g A\n",p,Dpm);

    return 0;
}
```

```
Amperomierz analogowy LE-3P
Zakres pomiarowy: 12 A
Liczba dzialek podzialki: 75
Klasa dokladnosci: 0.5
-----------------------------------
Bezwzgledny maksymalny blad pomiaru:
* dla p = 0.2, Dpm = 0.092 A
* dla p = 0.5, Dpm = 0.14 A
```


Inicjalizacja zmiennej strukturalnej

- Inicjalizowane mogą być tylko zmienne strukturalne, nie można inicjalizować pól w deklaracji struktury

```
struct osoba
{
   char imie[15];
   char nazwisko[20];
   int wiek, waga;
};

int main(void)
{
   struct osoba Nowak1 = {"Jan","Nowak",25,74};
   ...
}
```