Informatyka 2 (ES1F2012)

Politechnika Białostocka - Wydział Elektryczny

Elektrotechnika, semestr II, studia stacjonarne I stopnia

Rok akademicki 2022/2023

Pracownia nr 1 (23.02.2023)

dr inż. Jarosław Forenc

Dane podstawowe

- dr inż. Jarosław Forenc
- Politechnika Białostocka, Wydział Elektryczny,
  Katedra Elektrotechniki, Energoelektroniki i Elektroenergetyki
  ul. Wiejska 45D, 15-351 Białystok
  WE-204
- e-mail: j.forenc@pb.edu.pl
- tel. (0-85) 746-93-97
- http://jforenc.prv.pl
  - Dydaktyka - dodatkowe materiały do pracowni
- Konsultacje
  - (termin zostanie podany w drugim tygodniu semestru)

Program przedmiotu

1. Zajęcia organizacyjne. Struktury, odwołania do pól struktury. Inicjalizacja zmiennej strukturalnej.
2. Wskaźniki. Dynamiczny przydział pamięci w języku C.
3. Funkcje, ogólna struktura funkcji. Umieszczanie definicji funkcji w programie.
4. Funkcje, przekazywanie argumentów do funkcji przez wartość i wskaźnik. Rekurencyjne wywołanie funkcji.
5. Programy wielomodułowe.
6. Kolokwium nr 1.

Program przedmiotu

7. Zaawansowane operacje wejścia-wyjścia w języku C.
8. Pliki tekstowe w języku C.
9. Pliki binarne w języku C.
10. Operacje na plikach tekstowych i binarnych.
11. Kolokwium nr 2.
12. Matlab. Skrypty i funkcje.
13. Matlab. Elementy programowania.
14. Matlab. Zastosowanie programu do rozwiązywania wybranych zagadnień elektrotechniki.
15. Operatory bitowe. Zaliczenie zajęć.

Literatura

1. S. Prata: Język C. Szkoła programowania. Wydanie VI. Helion, Gliwice, 2016.
2. B.W. Kernighan, D.M. Ritchie: Język ANSI C. Programowanie. Wydanie II. Helion, 2010.
3. P.J. Deitel, H. Deitel: Język C. Solidna wiedza w praktyce. Helion, 2020.
4. Reese R.: Wskaźniki w języku C. Przewodnik. Helion, Gliwice, 2014.
5. B. Mrozek, Z. Mrozek: Matlab i Simulink. Poradnik użytkownika. Wydanie IV. Helion, Gliwice, 2017.
6. R. Pratap: Matlab dla naukowców i inżynierów. Wydanie 2. PWN, Warszawa, 2015.
7. M. Wciślik: Technika obliczeń inżynierskich w Matlabie. Wydawnictwo Politechniki Świętokrzyskiej, Kielce, 2021.
8. Instrukcje do pracowni specjalistycznej.
9. Materiały na stronie internetowej: http://jforenc.prv.pl/dydaktyka.html

Warunki zaliczenia przedmiotu

- Obecność na zajęciach:
  - więcej niż trzy nieusprawiedliwione nieobecności skutkują niezaliczeniem pracowni
- Realizacja w trakcie zajęć zadań przedstawionych przez prowadzącego
- Zaliczenie dwóch kolokwiów - praktycznych sprawdzianów pisania programów komputerowych
  - kolokwia odbędą się na zajęciach nr 6 (30.03.2023) i 14 (11.05.2023)
  - poprawy kolokwiów odbędą się w terminie ustalonym ze studentami
  - na kolokwiach można korzystać z własnych materiałów (instrukcje do przedmiotu, książki, notatki, itp.)
  - za każde kolokwium można otrzymać od 0 do 100 pkt.
  - oba kolokwia muszą być zaliczone na ocenę pozytywną (min. 51 pkt.)

Warunki zaliczenia przedmiotu

- Zaliczenie dwóch kolokwiów - praktycznych sprawdzianów pisania programów komputerowych (c.d.):
  - na podstawie otrzymanych punktów wystawiana jest ocena:

Punkty	Ocena	Punkty	Ocena
91 - 100	5,0	61 - 70	3,5
81 - 90	4,5	51 - 60	3,0
71 - 80	4,0	0 - 50	2,0

- Zaliczenie projektu z programu Matlab:
  - projekt realizowany będzie na zajęciach nr 14
  - za projekt można otrzymać od 0 do 100 pkt.
  - ocena za projekt wyznaczana jest w taki sam sposób jak za kolokwium

Warunki zaliczenia przedmiotu

- Zaliczenie prac domowych:
  - prace domowe polegają na napisaniu programów komputerowych wskazanych przez prowadzącego zajęcia
  - pracę domową należy wysłać na adres e-mailowy: j.forenc@pb.edu.pl do końca dnia (godz. 23:59), w którym odbywają się kolejne zajęcia
  - błędy w programach mogą być poprawione do końca dnia, w którym będą odbywały się następne zajęcia
  - za każdy poprawnie działający program student otrzymuje 1 pkt.
  - w przypadku stwierdzenia niesamodzielności pracy domowej: nie jest ona zaliczana, nie można jej ponownie oddać, student otrzymuje -1 pkt.
  - należy wysyłać tylko pliki z kodem źródłowym (.c, .cpp, .txt)
  - przed wysłaniem programu należy sprawdzić czy program:
    - kompiluje się oraz kompilator nie wyświetla ostrzeżeń
    - jest poprawnym rozwiązaniem zadania

Warunki zaliczenia przedmiotu

- przed wysłaniem programu należy sprawdzić czy program:
  - nosi nazwę zgodną z poniższym wzorcem:

    Nazwisko_Imie_NrGrupy_NrInstrukcji_NrZadania.cpp

    np. Kowalski_Jan_PS1_INF21_3.cpp
  - zawiera na początku komentarz nagłówkowy:

```
/*
  Nazwa: Kowalski_Jan_PS1_INF21_3.cpp
  Autor: Jan Kowalski, gr. PS1
  Album: 123456
  Data:  23-02-2023
  Kod:   ES1F2012
  Forma: Pracownia specjalistyczna
  IDE:   Visual Studio 2008
  Opis:  Program wyświetlający tekst "Witaj swiecie"
*/
```

Warunki zaliczenia przedmiotu

- Zaliczenie prac domowych (c.d.):
  - co najmniej 70% prac domowych musi być zaliczonych
  - prowadzący zajęcia może określić dodatkowe zasady zaliczania prac domowych pozwalające otrzymać wyższą ocenę
- Ocena końcowa wyznaczana jest na podstawie sumy otrzymanych punktów (kolokwia, projekt, dodatkowe punkty):

Punkty	Ocena	Punkty	Ocena
273 - 300	5,0	183 - 212	3,5
243 - 272	4,5	153 - 182	3,0
213 - 242	4,0	0 - 152	2,0

Efekty uczenia się i system ich oceniania

Podstawę do zaliczenia przedmiotu (uzyskanie punktów ECTS) stanowi stwierdzenie, że każdy z założonych efektów uczenia się został osiągnięty.

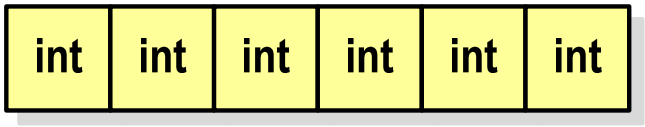
- Student, który zaliczył przedmiot potrafi:

EU1	definiować i wykorzystywać własne funkcje w programach w języku C
EU2	tworzyć programy wielomodułowe w języku C
EU3	stosować operacje zapisu i odczytu plików w samodzielnie napisanych programach komputerowych
EU4	tworzyć skrypty i funkcje w środowisku obliczeniowym rozwiązujące typowe zadania inżynierskie występujące w elektrotechnice

- Szczegółowy system oceniania znajduje się w systemie USOS

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;
    ...
};
```

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

- Elementy struktury to pola (dane, komponenty, składowe) struktury

Deklaracja struktury

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

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

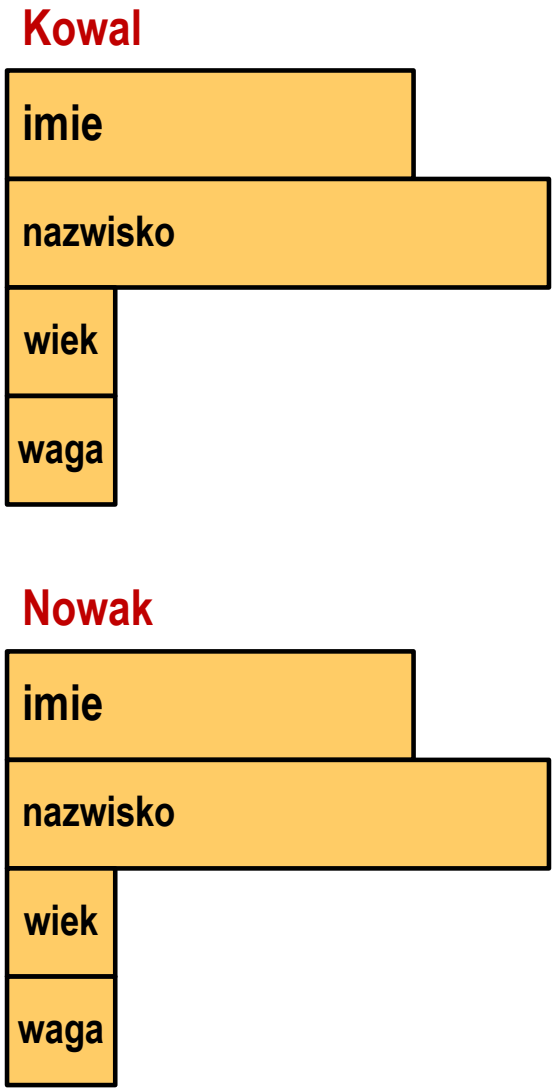
- Deklarując strukturę tworzymy nowy typ danych (struct osoba, struct zesp), którym można posługiwać się tak samo jak każdym innym typem standardowym
- Deklaracja struktury nie tworzy obiektu (nie przydziela pamięci)
- Zapisanie danych do struktury wymaga zdefiniowania zmiennej strukturalnej

Deklaracja zmiennej strukturalnej

```
#include <stdio.h>

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

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



- Kowal, Nowak - zmienne typu struct osoba

Odwołania do pól struktury

- Dostęp do pól struktury możliwy jest dzięki konstrukcji typu:

```
nazwa_zmiennej_strukturalnej.nazwa_pola
```

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

```
Nowak.wiek = 25;
strcpy(Nowak.imie,"Jan");
```

- Wyrażenie Nowak.wiek traktowane jest jak zmienna typu int, natomiast wyrażenie Nowak.imie traktowane jest jak łańcuch znaków

```
printf("%s - wiek %d\n", Nowak.imie, Nowak.wiek);
scanf("%d", &Nowak.wiek);
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 w języku C

- Inicjalizacja może dotyczyć tylko zmiennych strukturalnych, nie można inicjalizować pól w deklaracji struktury

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

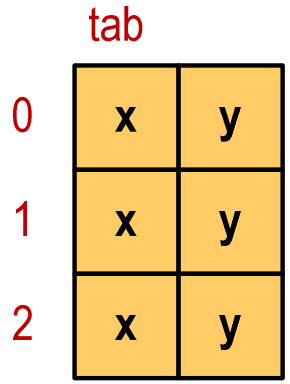
```
struct osoba Nowak = {"Jan","Nowak",25,74};
```

- Do zmiennych strukturalnych można stosować operator przypisania (=)

```
struct osoba Kowal = {"Ewa","Kowal",21,54};
struct osoba Kowal1;
Kowal1 = Kowal;
```

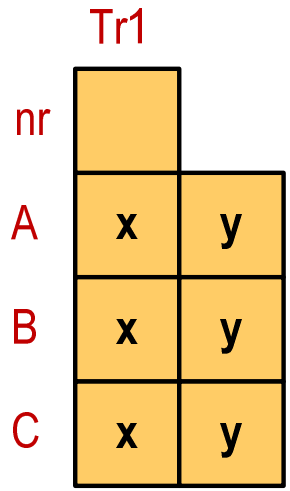
Złożone deklaracje struktur

```
struct punkt
{
    int x;
    int y;
} tab[3];
```




```
tab[0].x = 10;
tab[0].y = 20;
tab[1].x = 15;
…
```

```
struct trojkat
{
    int nr;
    struct punkt A, B, C;
} Tr1;
```




```
Tr1.nr = 1;
Tr1.A.x = 10;
Tr1.A.y = 20;
Tr1.B.x = 15;
…
```