Politechnika Białostocka
Wydział Elektryczny
Katedra Elektrotechniki Teoretycznej i Metrologii

Instrukcja
do pracowni specjalistycznej z przedmiotu

Informatyka 1

Kod przedmiotu: **ES1D200 009**

(studia stacjonarne)

JĘZYK C - TABLICE JEDNOWYMIAROWE

Numer ćwiczenia

INF07

Autor:
dr inż. Jarosław Forenc

Białystok 2017

Spis treści

Materiały dydaktyczne przeznaczone dla studentów Wydziału Elektrycznego PB.

1. Opis stanowiska

1.1. Stosowana aparatura

Podczas zajęć wykorzystywany jest komputer klasy PC z systemem operacyjnym Microsoft Windows (XP/Vista/7).

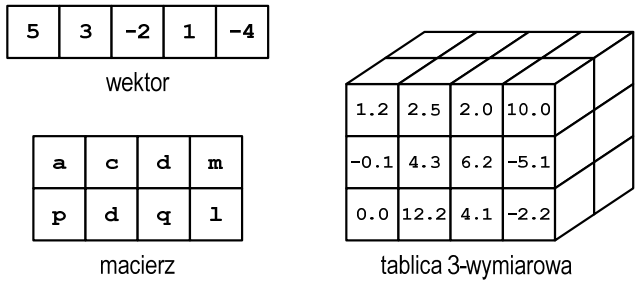
1.2. Oprogramowanie

Na komputerach zainstalowane jest środowisko programistyczne Microsoft Visual Studio 2008 Standard Edition lub Microsoft Visual Studio 2008 Express Edition zawierające kompilator Microsoft Visual C++ 2008.

2. Wiadomości teoretyczne

2.1. Tablica elementów

Tablica elementów jest ciągłym obszarem pamięci, w którym te elementy są umieszczone. W tablicy mogą znajdować się elementy tylko jednego typu. Wyróżnia się tablice jednowymiarowe (wektory), dwuwymiarowe (macierze) i tablice o większej liczbie wymiarów (Rys. 1).

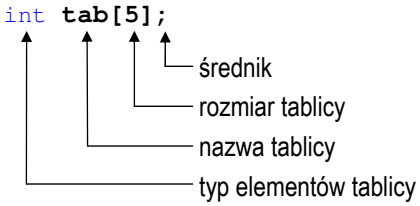


Rys. 1. Tablice elementów w języku C

Głównym celem stosowania tablic jest zastąpienie wielu zmiennych tego samego typu jedną tablicą.

2.2. Tablica jednowymiarowa (wektor)

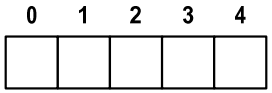
Deklarując tablicę jednowymiarową należy podać: typ elementów, nazwę tablicy i liczbę jej elementów, np.



Wyrażenie podane w nawiasach kwadratowych, określające rozmiar tablicy, musi dawać w wyniku dodatnią stałą całkowitoliczbową. Ponadto musi to być wartość znana już w fazie kompilacji (nie może to być zmienna). Jako rozmiar tablicy można podać nazwę stałej zdefiniowanej dyrektywą preprocesora **#define** lub z użyciem słowa kluczowego **const**.

`int tab[5];`	`#define N 5` `int tab[N];`	`const int n = 5;` `int tab[n];`

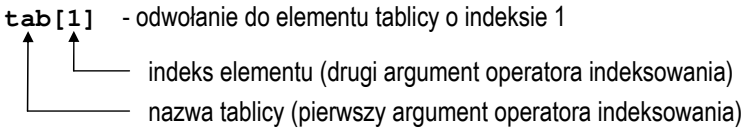
Powyższe deklaracje definiują tablicę pięciu elementów typu **int** (Rys. 2). Jest to tablica jednowymiarowa, czyli tzw. **wektor**.



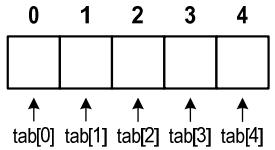
Rys. 2. Wektor 5-elementowy

Każdy element tablicy ma swój numer zwany **indeksem**, element zerowy (znajdujący się na początku tablicy) ma indeks **0** (zero), zaś ostatni **N-1**, gdzie **N** - rozmiar tablicy. Wartość indeksu pokazuje o ile elementów jest dany element odległy od początku tablicy. Nazwa tablicy jest adresem jej zerowego elementu (o indeksie **0**) w pamięci komputera.

Odwołania do elementów tablicy (odczytanie lub zapisanie wartości) wykonuje się za pomocą dwuargumentowego operatora indeksowania **[ ]**, np.



Odwołania do kolejnych elementów tablicy **tab** mają postać pokazaną na Rys. 3.



Rys. 3. Odwołania do elementów tablicy

Zapisanie wartości **5** do elementu tablicy **tab** o indeksie **1** oraz odczytanie tego elementu i przypisanie jego wartości zmiennej o nazwie **x**:

```
tab[1] = 5;
x = tab[1];
```

Jako indeks może występować:

- stała liczbowa, np. **0**, **1**, **5**;
- nazwa zmiennej przechowującej liczbę całkowitą, np. **i**, **idx**;
- wyrażenie dające w wyniku liczbę całkowitą, np. **i*j+5**;

Przy odwołaniach do elementów tablicy kompilator nie sprawdza, czy zapis lub odczyt odbywa się poza obszarem pamięci przydzielonym na tablicę, np.

```
int tab[5];
tab[5] = 10;
```

W powyższym fragmencie programu zadeklarowano 5-elementową tablicę o nazwie **tab**. Odwołanie **tab[5]** jest błędne, gdyż nie istnieje element o indeksie **5**.

Kompilator nie zasygnalizuje błędu, tylko w obszarze pamięci za tablicą zapisze wartość **10**.

Operacje na tablicach wykonywane są najczęściej przy wykorzystaniu pętli **for**. Załóżmy, że do wszystkich elementów tablicy **tab** należy zapisać wartość **10**. Kod realizujący taką operację może mieć następującą postać:

```
int tab[5];

tab[0] = 10;
tab[1] = 10;
tab[2] = 10;
tab[3] = 10;
tab[4] = 10;
```

Można to samo zrobić znacznie prościej, stosując pętlę **for**:

```
int tab[5], i;

for (i=0; i<5; i++)
    tab[i] = 10;
```

Zmienna **i** przyjmuje wartości od **0** do **4**, czyli takie same jak kolejne indeksy elementów tablicy.

W poniższym programie przedstawiono najczęściej wykonywane operacje na tablicy jednowymiarowej (wektorze) przechowującej liczby całkowite.

Program wykonujący wybrane operacje na wektorze liczb całkowitych.

```
#include <stdio.h>
#define N 10
#pragma warning(disable:4996)

int main(void)
{
    int   tab[N], i, j, min, suma = 0, tmp;
    float srednia;
```

```c
    /* wczytanie elementow tablicy */

    for (i=0; i<N; i++)
    {
        printf("Podaj liczbe nr %d: ",i+1);
        scanf("%d",&tab[i]);
    }

    /* wyswietlenie elementow tablicy */

    printf("\nElementy tablicy:\n");
    for (i=0; i<N; i++)
        printf("%d  ",tab[i]);
    printf("\n\n");

    /* wyswietlenie tablicy w odwrotnej kolejnosci */

    printf("Tablica w odwrotnej kolejnosci:\n");
    for (i=N-1; i>=0; i--)
        printf("%d  ",tab[i]);
    printf("\n\n");

    /* wyszukanie elementu o najmniejszej wartosci */

    min = tab[0];
    for (i=1; i<N; i++)
        if (tab[i]<min)
            min = tab[i];
    printf("Wartosc elementu najmniejszego: %d\n",min);

    /* indeksy elementow o najmniejszej wartosci */

    printf("Indeksy elementu najmniejszego: ");
    for (i=0; i<N; i++)
        if (tab[i]==min)
            printf("%d  ",i);
    printf("\n\n");

    /* suma i srednia arytmetyczna elementow tablicy */

    for (i=0; i<N; i++)
        suma = suma + tab[i];
    srednia = (float) suma/N;
    printf("Suma: %d, srednia: %f\n\n",suma,srednia);
```



```c
    /* sortowanie i wyswietlenie elementow tablicy */

    for (i=0; i<N-1; i++)
        for (j=i+1; j<N; j++)
            if (tab[i] > tab[j])
            {
                tmp = tab[i];
                tab[i] = tab[j];
                tab[j] = tmp;
            }

    printf("Elementy tablicy po sortowaniu:\n");
    for (i=0; i<N; i++)
        printf("%d  ",tab[i]);
    printf("\n");

    return 0;
}
```

Przykładowy wynik uruchomienia programu:

```
Podaj liczbe nr 2: 6
Podaj liczbe nr 3: 4
Podaj liczbe nr 4: 2
Podaj liczbe nr 5: 1
Podaj liczbe nr 6: 7
Podaj liczbe nr 7: 4
Podaj liczbe nr 8: 6
Podaj liczbe nr 9: 3
Podaj liczbe nr 10: 5

Elementy tablicy:
3  6  4  2  1  7  4  6  3  5

Tablica w odwrotnej kolejnosci:
5  3  6  4  7  1  2  4  6  3

Wartosc elementu najmniejszego: 1
Indeksy elementu najmniejszego: 4

Suma: 41, srednia: 4.100000

Elementy tablicy po sortowaniu:
1  2  3  3  4  4  5  6  6  7
```

Rozmiar tablicy określony został przy użyciu dyrektywy preprocesora **#define**.

```
#define N 10
```

Dzięki temu zmiana rozmiaru tablicy będzie wymagała tylko zmiany wartości w dyrektywie **#define**, a nie w każdym innym miejscu programu, gdzie pojawia się jego wartość. Dotyczy to zwłaszcza warunków w pętlach **for**.

W programie wykonywane są następujące operacje na tablicy:

- wczytanie elementów tablicy - w pętli **for** wyświetlamy komunikat „**Podaj liczbę nr ...**”, a następnie funkcją **scanf()** wczytujemy liczbę:

```
for (i=0; i<N; i++)
{
    printf("Podaj liczbe nr %d: ",i+1);
    scanf("%d",&tab[i]);
}
```

- wyświetlenie elementów tablicy w jednym wierszu:

```
printf("Elementy wektora:\n");
for (i=0; i<N; i++)
    printf("%d  ", tab[i]);
```

- wyświetlenie elementów tablicy w odwrotnej kolejności - zmieniamy wyrażenia w pętli **for**, zmienna **i** będzie przyjmowała wartości od **N-1** (ostatni element tablicy) do **0** (zerowy element tablicy):

```
printf("Tablica w odwrotnej kolejnosci:\n");
for (i=N-1; i>=0; i--)
    printf("%d  ", tab[i]);
```

- wyszukanie elementu o najmniejszej wartości - zakładamy, że zerowy element tablicy jest najmniejszy (**min = tab[0]**); przeglądamy pozostałe elementy tablicy; jeśli kolejny z elementów tablicy (**tab[i]**) jest mniejszy od

dotychczasowego najmniejszego (**min**), to element ten staje się najmniejszym (**min = tab[i]**):

```
min = tab[0];
for (i=1; i<N; i++)
    if (tab[i]<min)
        min = tab[i];
printf("Wartosc elementu najmniejszego: %d\n",min);
```

- wyszukanie indeksów elementu o najmniejszej wartości - przeglądamy tablicę poszukując elementów równych najmniejszemu (**tab[i]==min**); po znalezieniu takiego elementu wyświetlamy jego indeks czyli wartość zmiennej **i**:

```
printf("Indeksy elementu najmniejszego: ");
for (i=0; i<N; i++)
    if (tab[i]==min)
        printf("%d  ",i);
printf("\n");
```

- obliczenie sumy i średniej arytmetycznej elementów tablicy - w pętli **for** dodajemy kolejne elementy tablicy do zmiennej **suma** (przed pętlą zmienna ta musi być wyzerowana); następnie obliczamy średnią arytmetyczną dzieląc **sumę** przez ilość elementów (**N**); ponieważ **suma** i **N** są całkowite, to w celu uniknięcia dzielenia liczb całkowitych, wymuszamy zmianę typu zmiennej **suma** na typ **float**: **(float) suma**:

```
for (i=0; i<N; i++)
    suma = suma + tab[i];
srednia = (float) suma/N;
printf("Suma: %d, srednia: %f\n",suma,srednia);
```

- sortowanie elementów tablicy w kolejności od najmniejszego do największego - pętla zewnętrzna określa indeks elementu (**i**), którego wartość jest porównywana z wartościami pozostałych elementów w tablicy (o indeksach **i+1**,**i+2**,...,**N-1**) określanych w pętli wewnętrznej; jeśli kolejność elementów

jest nieprawidłowa (tzn. **tab[i] > tab[j]**), to elementy te zamieniane są miejscami (**tmp = tab[i]; tab[i] = tab[j]; tab[j] = tmp;**):

```
for (i=0; i<N-1; i++)
    for (j=i+1; j<N; j++)
        if (tab[i] > tab[j])
        {
            tmp = tab[i];
            tab[i] = tab[j];
            tab[j] = tmp;
        }
```

Po zakończeniu sortowania elementy tablicy są ponownie wyświetlane:

```
printf("Elementy tablicy po sortowaniu:\n");
for (i=0; i<N; i++)
    printf("%d  ",tab[i]);
printf("\n");
```

2.3. Generowanie pseudolosowe elementów tablicy

Elementy tablicy mogą być wygenerowane pseudolosowo, co pokazuje poniższy program.

Generowanie pseudolosowe elementów tablicy.

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

int main(void)
{
    int  tab[10], i;

    srand((unsigned int)time(NULL));
    for (i=0; i<10; i++)
    {
        tab[i] = rand();
        printf("%d  ",tab[i]);
    }
```



```
    return 0;
}
```

Do generowania pseudolosowych liczb zastosowana została funkcja **rand()**:

```
tab[i] = rand();
```

Funkcja ta zwraca pseudolosową liczbę całkowitą z zakresu: **0 … RAND_MAX** (**32767**). Zastosowanie jej w programie wymaga dołączenia pliku nagłówkowego **stdlib.h**. Przed użyciem funkcji **rand()** należy zainicjalizować generator liczb pseudolosowych wywołując funkcję **srand()**:

```
srand((unsigned int)time(NULL));
```

której argumentem jest liczba inicjalizująca generator. Aby zapewnić unikalność generowania kolejnych liczb, do funkcji **srand()** przekazywana jest wartość zwracana przez funkcję **time()**. Zastosowanie funkcji **time()** wymaga dołączenia pliku nagłówkowego **time.h**. Wymuszenie konwersji typu **(unsigned int)** likwiduje ostrzeżenie kompilatora warning C4244: 'argument' : conversion from 'time_t' to 'unsigned int', possible loss of data wynikające z niezgodności typu wartości zwracanej przez funkcję **time()** (**time_t**) i typu wartości oczekiwanej przez funkcję **srand()** (**unsigned int**).

Zmiana zakresu generowanych liczb odbywa się poprzez zastosowanie dzielenia modulo. Jeśli chcemy otrzymać liczby całkowite z zakresu **0 … 10**, to wystarczy wartość zwracaną przez funkcję **rand()** podzielić **modulo 11**:

```
int x;
x = rand() % 11;
```

Pseudolosową liczbę całkowitą z przedziału ⟨**a, b**⟩ otrzymamy używając funkcji **rand()** w następujący sposób:

```
int x;
x = rand() % (b - a + 1) + a;
```

2.4. Inicjalizacja elementów tablicy

Po zadeklarowaniu tablicy wartości jej elementów są nieokreślone. Inicjalizacja elementów tablicy jest to nadanie wartości elementom od razu przy deklaracji. Inicjalizacja taka polega na umieszczeniu w deklaracji po znaku równości, ujętej w nawiasy klamrowe, listy wartości kolejnych jej elementów, np.

```
int a[3] = {5,7,1};
```

0	1	2
5	7	1

Poszczególne elementy tablicy oddzielone są od siebie przecinkami. Jako kolejne elementy mogą występować liczby lub wyrażenia arytmetyczne. Tablice można inicjalizować **tylko** przy deklaracji.

Jeśli wartości podanych w trakcie inicjalizacji jest mniej niż wynosi rozmiar tablicy, to pozostałe elementy tablicy wypełniane są zerami, np.

```
int a[5] = {5,7,1};
```

0	1	2	3	4
5	7	1	0	0

Jeśli wartości podanych w trakcie inicjalizacji jest więcej niż wynosi rozmiar tablicy, to kompilator zasygnalizuje błąd, np.

```
int a[5] = {5,7,1,3,6,4};
```

Tablica zadeklarowana bez podania rozmiaru, a zainicjalizowana ma liczbę elementów równą ilości inicjatorów, np.

```
int a[] = {2,3,1,4};
```

jest równoważne:


```
int a[4] = {2,3,1,4};
```

Poniższy program używa tablicy jednowymiarowej do zapamiętania wygenerowanych pseudolosowo, niepowtarzających się liczb całkowitych.

Program generujący N niepowtarzających się liczb całkowitych.

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

#define  N      6
#define  ZAKRES 49

int main(void)
{
    int   lotto[N];
    int   i, j, x, powt;

    srand((unsigned int)time(NULL));
    for (i=0; i<N; i++)
    {
        do
        {
            powt = 0;
            x = rand() % ZAKRES + 1;
            for (j=0; j<i; j++)
                if (lotto[j]==x)
                    powt = 1;
        } while (powt==1);
        lotto[i] = x;
    }
    printf("Wylosowane liczby: ");
    for (i=0; i<N; i++)
        printf("%d  ",lotto[i]);
    printf("\n");

    return 0;
}
```

Przykładowy wynik uruchomienia programu:

```
Wylosowane liczby: 27  30  1  13  22  8
```

3. Przebieg ćwiczenia

Na pracowni specjalistycznej należy wykonać wybrane zadania wskazane przez prowadzącego zajęcia. W różnych grupach mogą być wykonywane różne zadania.

1. Zadeklaruj **N**-elementową tablicę liczba całkowitych typu **int** (**N** - stała zadeklarowana dyrektywą preprocesora **#define**). Wykonaj następujące operacje:
   - zapisz do tablicy kolejne liczby całkowite **1, 2, 3, …, N**; wyświetl elementy tablicy w jednym wierszu,
   - zapisz do tablicy liczby całkowite **N, N-1, …, 3, 2, 1**; wyświetl elementy tablicy w jednym wierszu,
   - zapisz do tablicy wygenerowane pseudolosowo liczby całkowite z zakresu ⟨**0, 9**⟩; wyświetl elementy tablicy w jednym wierszu,
   - wyświetl w jednym wierszu elementy tablicy o parzystych indeksach,
   - wyświetl w jednym wierszu elementy tablicy o nieparzystych indeksach,
   - oblicz i wyświetl sumę wszystkich elementów tablicy,
   - oblicz i wyświetl średnią arytmetyczną wszystkich elementów tablicy,
   - wyświetl wartość największego i najmniejszego elementu tablicy;
   - wczytaj z klawiatury liczbę **x**; sprawdź, czy **x** występuje w tablicy; jeśli tak, to wyświetl numer indeksu pierwszego elementu równego **x**,
   - wyświetl liczbę wystąpień **x** w tablicy,
   - wyświetl liczbę elementów tablicy mniejszych od **x** i liczbę elementów tablicy większych od **x**,
   - odwróć kolejność elementów tablicy; wyświetl elementy tablicy w jednym wierszu,

   - posortuj elementy tablicy w kolejności rosnącej; wyświetl elementy tablicy w jednym wierszu,
   - posortuj elementy tablicy w kolejności malejącej; wyświetl elementy tablicy w jednym wierszu.

2. Zadeklaruj **N**-elementową tablicę liczba całkowitych typu **int** (**N** - stała zadeklarowana dyrektywą preprocesora **#define**). Napisz i wywołaj następujące funkcje:
   - zapisującą do tablicy wygenerowane pseudolosowo liczby całkowite z zakresu ⟨**a, b**⟩, gdzie **a** i **b** są argumentami funkcji,
   - wyświetlającą elementy tablicy w jednym wierszu,
   - obliczającą i zwracającą sumę wszystkich elementów tablicy,
   - obliczającą i zwracającą średnią arytmetyczną elementów tablicy,
   - wyszukującą i zwracającą wartość największego elementu tablicy,
   - wyszukującą i zwracającą wartość najmniejszego elementu tablicy,
   - zwracającą liczbę wystąpień liczby **x** w tablicy, gdzie **x** - argument funkcji,
   - sortującą elementy tablicy w kolejności rosnącej.

3. Tablica **U** przechowuje wyniki **N**-pomiarów wartości chwilowych napięcia na pewnym dwójniku RLC (przyjmij **N** nie mniejsze niż **15**). Napisz program który:
   - zapisze do tablicy **U** wartości chwilowe napięcia zgodnie ze wzorem:

     ```
     U[i] = 10.0f*sin((i+5.0f)/5.0f);
     ```
     (1)

   - wyświetli na ekranie zapisane wartości chwilowe napięcia,
   - obliczy i wyświetli największą, najmniejszą i średnią wartość napięcia,
   - obliczy i wyświetli liczbę pomiarów, dla których wartość chwilowa napięcia była większa od wartości średniej,
   - zastąpi w tablicy **U** wszystkie ujemne wartości napięcia wartością zero i ponownie wyświetli wartości elementów tablicy.

4. W pomieszczeniu przeprowadzono pomiar temperatury. Temperaturę mierzono co godzinę (od godz. **0** do **23**). Wyniki pomiarów umieszczono w tablicy **T**. Indeksy elementów tablicy określają jednocześnie godzinę pomiaru. Zakładamy, że wyniki pomiarów nie powtarzają się. Napisz program który:
   - zapisze do tablicy **T** wartości temperatury zgodnie ze wzorem (dla **i = 0...23**):

     ```
     T[i] = sin(i/8.0f-10)*cos(i/8.0f-10)*30+10;
     ```
     (2)

   - wyświetli na ekranie zapisane wartości temperatury,
   - poda godzinę, o której temperatura była najwyższa oraz godzinę, o której temperatura była najniższa w ciągu całej doby,
   - obliczy i wyświetli średnią temperaturę w ciągu doby,
   - obliczy i wyświetli największą różnicę temperatur,
   - poda informację, czy temperatura w ciągu doby spadła poniżej zera stopni, czy też nie spadła poniżej zera stopni.

5. Dane są dwa **N**-elementowe wektory **A** i **B** zawierające wygenerowane pseudolosowo liczby całkowite z zakresu ⟨**0, 99**⟩:
   - utwórz wektor **C** zawierający na odpowiedniej pozycji większy z elementów wektorów **A** i **B**,
   - utwórz wektor **D** będący sumą wektorów **A** i **B,**
   - oblicz i wyświetl iloczyn skalarny wektorów **A** i **B**.

   Rozmiar wektorów (**N**) zadeklaruj jako stałą (**#define**). Wyświetl wektory **A**, **B**, **C** i **D**.

6. Napisz program, który dla **N**-elementowego wektora liczb całkowitych wygeneruje pseudolosowo elementy z zakresu ⟨**0, 10**⟩, wyświetli zawartość wektora oraz obliczy ile razy każda liczba występuje w wektorze.

7. Napisz program, który do **10**-elementowej tablicy liczb całkowitych zapisze wygenerowane pseudolosowo liczby z zakresu ⟨**0, 9**⟩. Liczby nie mogą powtarzać się. Wyświetl zawartość tablicy. Podaj ile razy generowano liczbę dla kolejnego elementu tablicy.

4. Literatura

[1] Kernighan B.W., Ritchie D.M.: Język ANSI C. Programowanie. Wydanie II. Helion, Gliwice, 2010.

[2] Prata S.: Język C. Szkoła programowania. Wydanie VI. Helion, Gliwice, 2016.

[3] Prinz P., Crawford T.: Język C w pigułce. APN Promise, Warszawa, 2016.

[4] King K.N.: Język C. Nowoczesne programowanie. Wydanie II. Helion, Gliwice, 2011.

[5] Kochan S.G.: Język C. Kompendium wiedzy. Wydanie IV. Helion, Gliwice, 2015.

[6] Wileczek R.: Microsoft Visual C++ 2008. Tworzenie aplikacji dla Windows. Helion, Gliwice, 2009.

5. Zagadnienia na zaliczenie

1. Omów sposób deklarowania tablic jednowymiarowych (wektorów) w języku C oraz odwoływania się do elementów tych tablic.
2. Wyjaśnij, jak odszukać w tablicy jednowymiarowej element o najmniejszej i element o największej wartości.
3. W jaki sposób w języku C można generować pseudolosowe liczby całkowite i rzeczywiste z określonego zakresu?
4. Opisz inicjalizację elementów tablicy jednowymiarowej.

6. Wymagania BHP

Warunkiem przystąpienia do praktycznej realizacji ćwiczenia jest zapoznanie się z instrukcją BHP i instrukcją przeciw pożarową oraz przestrzeganie zasad w nich zawartych.

W trakcie zajęć laboratoryjnych należy przestrzegać następujących zasad.

- Sprawdzić, czy urządzenia dostępne na stanowisku laboratoryjnym są w stanie kompletnym, nie wskazującym na fizyczne uszkodzenie.
- Jeżeli istnieje taka możliwość, należy dostosować warunki stanowiska do własnych potrzeb, ze względu na ergonomię. Monitor komputera ustawić w sposób zapewniający stałą i wygodną obserwację dla wszystkich członków zespołu.
- Sprawdzić prawidłowość połączeń urządzeń.
- Załączenie komputera może nastąpić po wyrażeniu zgody przez prowadzącego.
- W trakcie pracy z komputerem zabronione jest spożywanie posiłków i picie napojów.
- W przypadku zakończenia pracy należy zakończyć sesję przez wydanie polecenia wylogowania. Zamknięcie systemu operacyjnego może się odbywać tylko na wyraźne polecenie prowadzącego.
- Zabronione jest dokonywanie jakichkolwiek przełączeń oraz wymiana elementów składowych stanowiska.
- Zabroniona jest zmiana konfiguracji komputera, w tym systemu operacyjnego i programów użytkowych, która nie wynika z programu zajęć i nie jest wykonywana w porozumieniu z prowadzącym zajęcia.
- W przypadku zaniku napięcia zasilającego należy niezwłocznie wyłączyć wszystkie urządzenia.
- Stwierdzone wszelkie braki w wyposażeniu stanowiska oraz nieprawidłowości w funkcjonowaniu sprzętu należy przekazywać prowadzącemu zajęcia.
- Zabrania się samodzielnego włączania, manipulowania i korzystania z urządzeń nie należących do danego ćwiczenia.
- W przypadku wystąpienia porażenia prądem elektrycznym należy niezwłocznie wyłączyć zasilanie stanowiska. Przed odłączeniem napięcia nie dotykać porażonego.