

Plan wykładu nr 7

- Język C - tablice jednowymiarowe (wektory)
 - deklaracja, odwołania do elementów, inicjalizacja tablicy
 - generator liczb pseudolosowych, operacje na wektorze

Informatyka 1 (EZ1E2008)

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

Wykład nr 7 (07.05.2021)

dr inż. Jarosław Forenc

Język C - tablica jednowymiarowa

- **Tablica** - ciągły obszar pamięci, w którym umieszczone są elementy tego samego typu
- **Wektor** - tablica jednowymiarowa

5	3	-2	0	-4
---	---	----	---	----

- liczby całkowite

3.1	0.2	2.3	-1.3	1.5	1.1	-4.0
-----	-----	-----	------	-----	-----	------

- liczby rzeczywiste

a	Z	x	&	M	+
---	---	---	---	---	---

- znaki

Język C - tablica elementów

- **Tablica** - ciągły obszar pamięci, w którym umieszczone są elementy tego samego typu

wektor

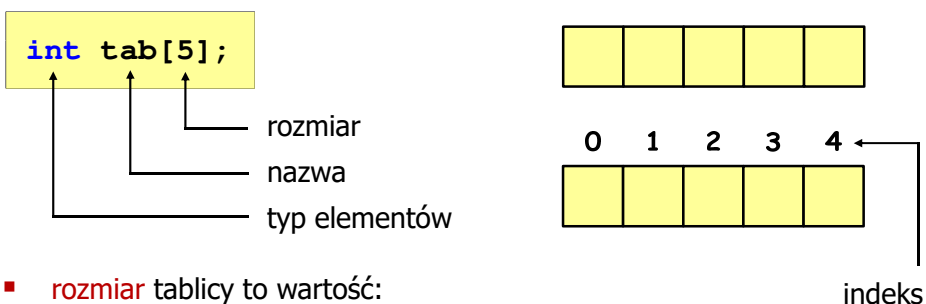
5	3	-2	1	-4
---	---	----	---	----

macierz

a	c	d	m
p	d	q	l
a	t	x	v

1.2	2.5	2.0	10.0
-0.1	4.3	6.2	-5.1
0.0	12.2	4.1	-2.2

Język C - deklaracja tablicy jednowymiarowej



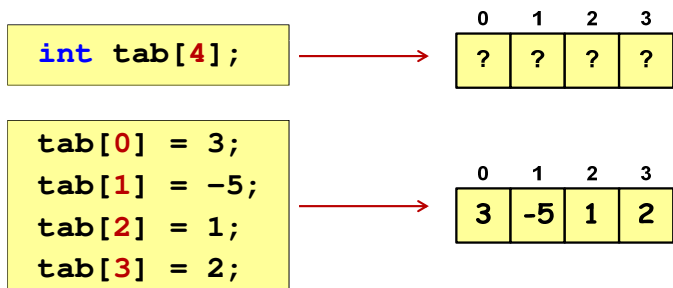
- rozmiar tablicy to wartość:
 - całkowita, dodatnia
 - znana na etapie kompilacji programu (stała liczbowa: 5, #define N 5, const int n = 5;)

```
int tab[5];
```

```
int tab[N];
```

```
int tab[n];
```

Język C - odwołania do elementów tablicy



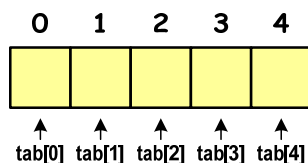
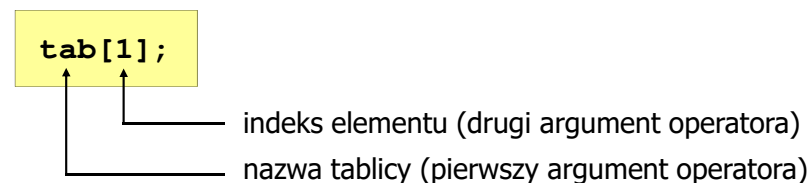
- Każdy element tablicy traktowany jest jak zmienna typu `int`

```
printf("%d", tab[0]);
```

```
scanf("%d", &tab[1]);
```

Język C - odwołania do elementów tablicy

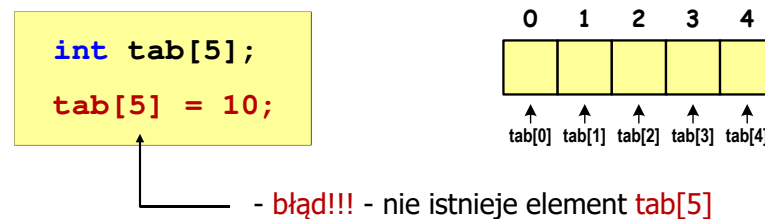
`[]` - dwuargumentowy operator indeksowania



- indeks:
 - stała liczbowa, np. 0, 1, 10
 - nazwa zmiennej, np. i, idx
 - wyrażenie, np. $i*j+5$

Język C - odwołania do elementów tablicy

- Przy odwołaniach do elementów tablicy kompilator nie sprawdza poprawności indeksów

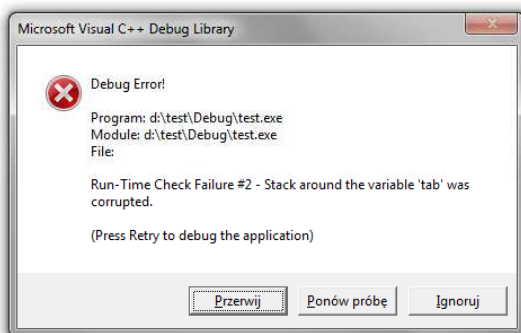


- Kompilator nie zasygnalizuje błędu
- Program wykona operację
- Środowisko programistyczne może zasygnalizować problem

Język C - odwołania do elementów tablicy

- Przy odwołaniach do elementów tablicy kompilator nie sprawdza poprawności indeksów

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



Język C - inicjalizacja tablicy jednowymiarowej

```
int tab[5] = {1, 2, 3, 4, 5};
```

0	1	2	3	4
1	2	3	4	5

```
int tab[5] = {1, 2, 3};
```

0	1	2	3	4
1	2	3	0	0

```
int tab[5] = {1, 2, 3, 4, 5, 6};
```

- błąd kompilacji

```
int tab[] = {1, 2, 3, 4, 5};
```

0	1	2	3	4
1	2	3	4	5

Język C - inicjalizacja tablicy jednowymiarowej

```
int tab[5] = {0};
```

0	1	2	3	4
0	0	0	0	0

```
int tab[5] = {};
```

0	1	2	3	4
0	0	0	0	0

Język C - odwołania do elementów tablicy

- Zapisanie wartości 1 do wszystkich elementów tablicy

```
int tab[5];
```

```
tab[0] = 1;
```

```
tab[1] = 1;
```

```
tab[2] = 1;
```

```
tab[3] = 1;
```

```
tab[4] = 1;
```

0	1	2	3	4
1	1	1	1	1

```
int tab[5], i;
```

```
for (i=0; i<5; i++)
```

```
    tab[i] = 1;
```

Przykład: operacje na dużej ilości danych (tablica)

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

```
int main(void)
```

```
{
```

```
    double U[5] = { 5.0, 10.0, 15.0, 20.0, 25.0 };
```

```
    double I[5] = { 0.16, 0.21, 0.27, 0.33, 0.36 };
```

```
    double R[5];
```

```
    int i;
```

```
    for (i=0; i<5; i++)
        R[i] = U[i]/I[i];
```

```
    for (i=0; i<5; i++)
        printf("R%d = %f\n", i+1, R[i]);
```

```
    return 0;
```

```
}
```

```
R1 = 31.250000
R2 = 47.619048
R3 = 55.555556
R4 = 60.606061
R5 = 69.444444
```

	0	1	2	3	4
U	5.0	10.0	15.0	20.0	25.0
I	0.16	0.21	0.27	0.33	0.36
R	31.25	47.62	55.56	60.61	69.44

Język C - generator liczb pseudolosowych

- `rand()` - zwraca liczbę pseudolosową - zakres: 0 ... RAND_MAX (0 ... 32767)
- `srand()` - inicjalizuje generator liczb pseudolosowych
- Plik nagłówkowy: `stdlib.h` (`time.h`)

```
int x, y, z;
```

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

```
x = rand(); // zakres <0,32767>
```

```
y = rand() % 100; // zakres <0,99>
```

```
z = rand() % (b-a+1)-a; // zakres <a,b>
```

Przykład: operacje na wektorze

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

```
#define N 10
```

```
int main(void)
```

```
{
```

```
    int tab[N], i;
```

```
    /* generowanie elementów tablicy */
```

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

```
    for (i=0; i<N; i++)
        tab[i] = rand() % 20;
```

0	1	2	3	4	5	6	7	8	9
11	12	14	9	6	11	6	18	9	10

Przykład: operacje na wektorze

```
/* wyświetlenie elementów tablicy */
```

```
printf("Elementy tablicy:\n");
```

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

```
Elementy tablicy:
11 12 14 9 6 11 6 18 9 10
```

0	1	2	3	4	5	6	7	8	9
11	12	14	9	6	11	6	18	9	10

N = 10

Przykład: operacje na wektorze

```
/* wyświetlenie elementów w odwrotnej kolejności */  
  
printf("Elementy w odwrotnej kolejności:\n");  
for (i=N-1; i>=0; i--)  
    printf("%d ", tab[i]);  
printf("\n");
```

```
Elementy w odwrotnej kolejności:  
10 9 18 6 11 6 9 14 12 11
```

0	1	2	3	4	5	6	7	8	9
11	12	14	9	6	11	6	18	9	10

N = 10

Przykład: operacje na wektorze

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

```
Wartosc elementu najmniejszego: 6
```

0	1	2	3	4	5	6	7	8	9
11	12	14	9	6	11	6	18	9	10

N = 10

Przykład: operacje na wektorze

```
/* indeksy elementów o najmniejszej wartości */  
  
printf("Indeksy elementu najmniejszego: ");  
for (i=0; i<N; i++)  
    if (tab[i]==min)  
        printf("%d ", i);  
printf("\n");
```

```
Indeksy elementu najmniejszego: 4 6
```

0	1	2	3	4	5	6	7	8	9
11	12	14	9	6	11	6	18	9	10

N = 10

Przykład: operacje na wektorze

```
/* suma i średnia arytmetyczna elementów tablicy */  
  
int suma = 0;  
float srednia;  
  
for (i=0; i<N; i++)  
    suma = suma + tab[i];  
srednia = (float) suma/N;  
printf("Suma: %d, srednia: %g\n", suma, srednia);
```

```
Suma: 106, srednia: 10.6
```

0	1	2	3	4	5	6	7	8	9
11	12	14	9	6	11	6	18	9	10

N = 10

Przykład: operacje na wektorze

```
/* liczba parzystych elementów tablicy */  
  
int ile = 0;  
  
for (i=0; i<N; i++)  
    if (tab[i]%2==0)  
        ile++;  
printf("Liczba parzystych elementow: %d\n",ile);
```

Liczba parzystych elementow: 6

0	1	2	3	4	5	6	7	8	9
11	12	14	9	6	11	6	18	9	10

N = 10

Przykład: operacje na wektorze

```
/* sortowanie elementów tablicy */  
  
int tmp, j;  
  
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;  
        }
```

0	1	2	3	4	5	6	7	8	9
11	12	14	9	6	11	6	18	9	10

→

0	1	2	3	4	5	6	7	8	9
6	6	9	9	10	11	11	12	14	18

Koniec wykładu nr 7

Dziękuję za uwagę!