Programowanie C (CP1S01005)

Politechnika Białostocka - Wydział Elektryczny

Cyfryzacja przemysłu, sem. I, studia stacjonarne I stopnia

Rok akademicki 2024/2025

Wykład nr 5 (28.11.2024)

dr inż. Jarosław Forenc

Plan wykładu nr 5

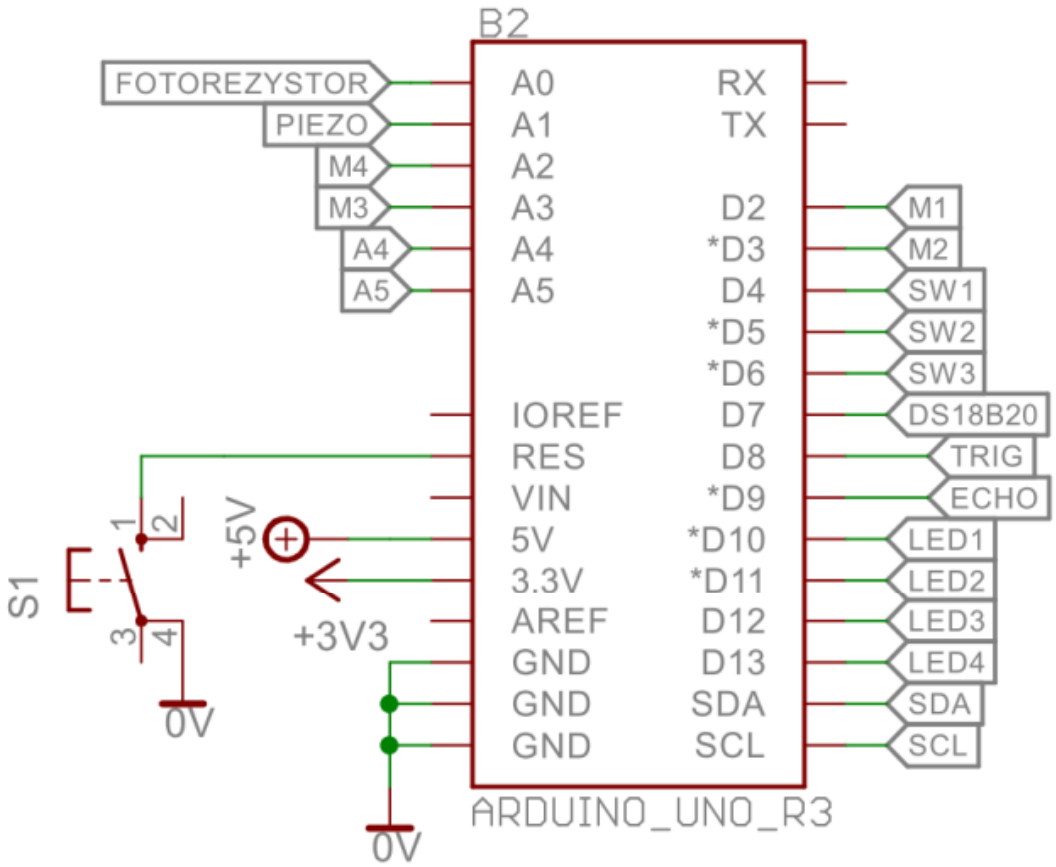
- Arduino
  - moduł wykorzystywany na zajęciach
  - sterowanie diodą LED
  - sterowanie przyciskiem
- Język C
  - tablice jednowymiarowe (wektory)

Arduino - moduł stosowany na zajęciach

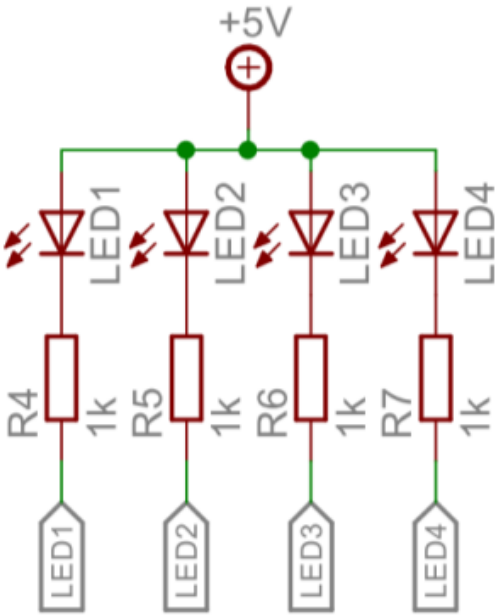
Arduino - lista elementów i czujników

- LED1, LED2, LED3, LED4 - diody LED w różnych kolorach
- Silnik krokowy
- Moduł BH1750 - czujnik natężenia światła
- Fotorezystor
- Moduł BMP280 - czujnik ciśnienia z wbudowanym termometrem
- Moduł MPU-6050 - 3-osiowy akcelerometr i żyroskop
- Moduł wyświetlacza OLED
- PIEZO - buzzer
- SW1, SW2, SW3 - przyciski
- Moduł DS18B20 - czujnik temperatury
- Moduł HC-SR04 - ultradźwiękowy czujnik odległości

Arduino - sterowanie diodą LED



Schemat podłączenia
wyprowadzeń modułu Arduino



Schemat podłączenia
diod LED

Arduino - sterowanie diodą LED

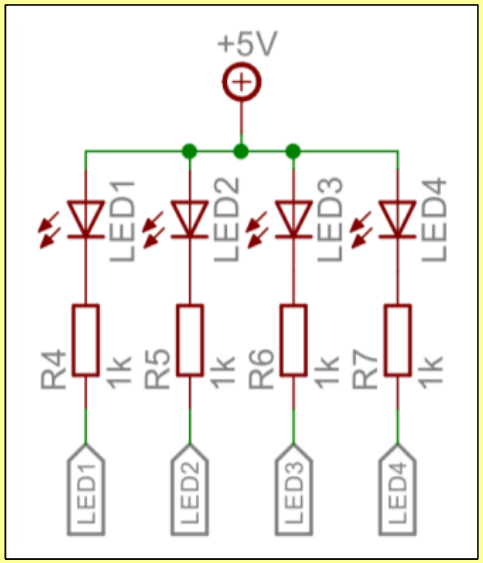
Migająca dioda LED.

```
#include <Arduino.h>

#define LED1_PIN 10

void setup() {
  pinMode(LED1_PIN, OUTPUT);
  digitalWrite(LED1_PIN, HIGH);
}

void loop() {
  digitalWrite(LED1_PIN, LOW);
  delay(500);
  digitalWrite(LED1_PIN, HIGH);
  delay(500);
}
```



Arduino - funkcja pinMode()

```
pinMode(pin, mode)
```

- Konfiguruje określony pin jako wejście lub wyjście
- pin - numer pinu Arduino, dla którego ma zostać ustawiony tryb
- mode - tryb działania określonego pinu Arduino
  - INPUT - pin działa jako wejście, umożliwia odczytanie stanu logicznego pinu (wysoki lub niski)
  - OUTPUT - pin działa jako wyjście, umożliwia ustawienie sygnału cyfrowego pinu (0 V dla stanu LOW lub 5 V dla stanu HIGH) za pomocą funkcji DigitalWrite()
  - INPUT_PULLUP - pin działa jako wejście z włączonym wewnętrznym rezystorem podciągającym
- Funkcja nie zwraca żadnej wartości

Arduino - funkcja DigitalWrite()

```
DigitalWrite(pin, value)
```

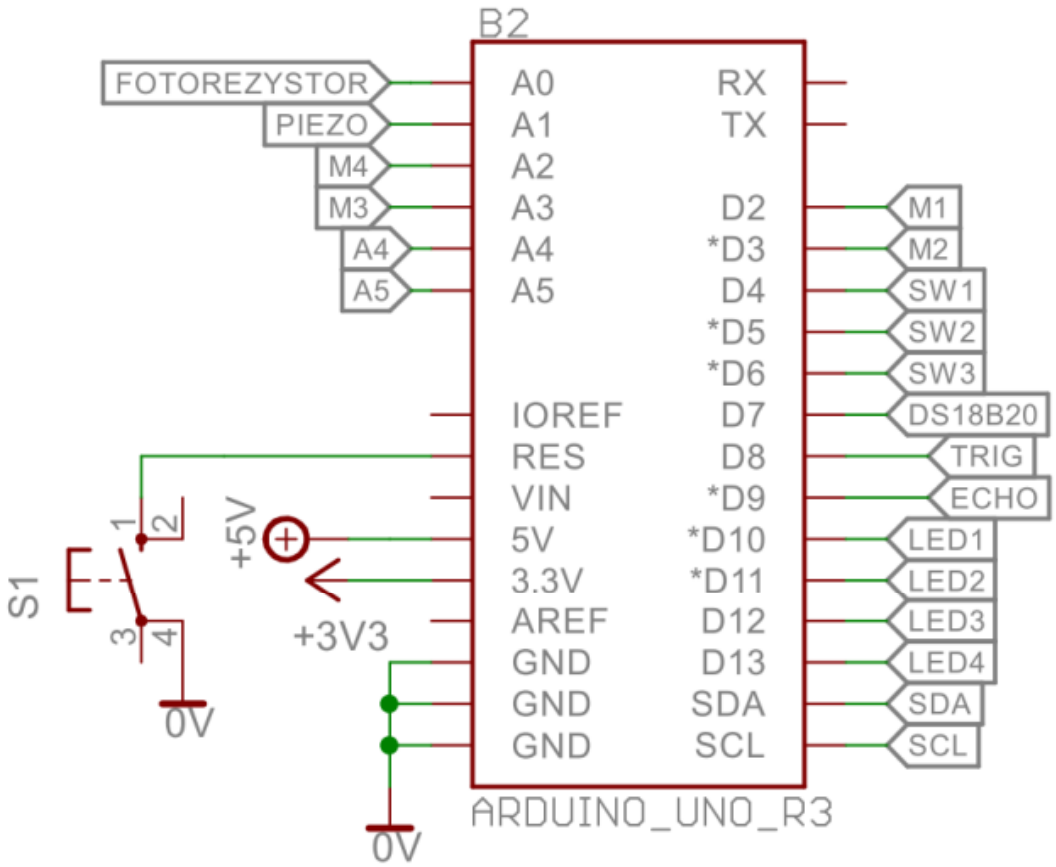
- Zapisuje wartość HIGH lub LOW na cyfrowym pinie
- pin - numer pinu Arduino
- value - zapisywana wartość na cyfrowym pinie (HIGH lub LOW)
- Jeśli pin został skonfigurowany jako OUTPUT za pomocą funkcji pinMode(), jego napięcie zostanie ustawione na wartość 5 V dla stanu HIGH lub 0 V (masa) dla stanu LOW
- Jeśli pin został skonfigurowany jako INPUT, funkcja digitalWrite() włącza (HIGH) lub wyłącza (LOW) wewnętrzny rezystor podciągający na pinie wejściowym
- Funkcja nie zwraca żadnej wartości

Arduino - funkcja delay()

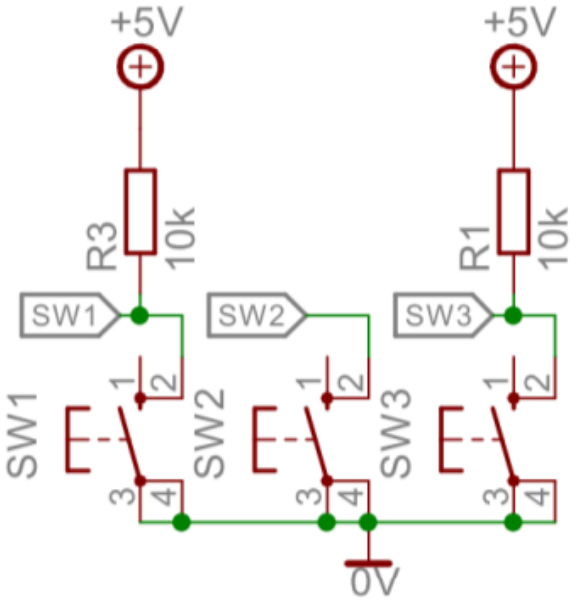
```
delay(ms)
```

- Wstrzymuje wykonanie programu na czas (w milisekundach) określony jako parametr (jedna sekunda to 1000 milisekund)
- ms - liczba milisekund, na którą program zostanie wstrzymany, dozwolone typy danych: unsigned long
- Funkcja nie zwraca żadnej wartości

Arduino - sterowanie przyciskiem



Schemat podłączenia
wyprowadzeń modułu Arduino



Schemat podłączenia
przycisków

Arduino - sterowanie przyciskiem

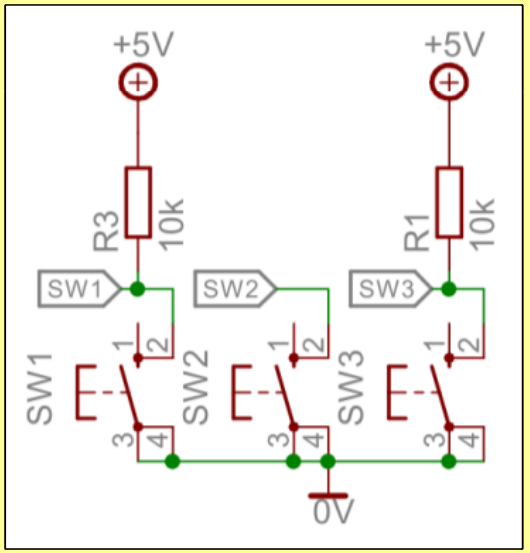
Przycisk zapalający diodę LED na 2 sekundy.

```
#include <Arduino.h>

#define LED1_PIN 10
#define SW1_PIN 4

void setup() {
  pinMode(LED1_PIN, OUTPUT);
  pinMode(SW1_PIN, INPUT);
  digitalWrite(LED1_PIN, HIGH);
}

void loop() {
  if (digitalRead(SW1_PIN) == LOW)
  {
    digitalWrite(LED1_PIN, LOW);
    delay(2000);
    digitalWrite(LED1_PIN, HIGH);
  }
}
```



Arduino - funkcja DigitalRead()

```
DigitalRead(pin)
```

- Odczytuje wartość logiczną (HIGH lub LOW) z określonego cyfrowego pinu
- pin - numer pinu Arduino
- Funkcja zwraca wartość:
  - HIGH - wysoki stan logiczny (5 V)
  - LOW - niski stan logiczny (0 V)

Arduino - sterowanie przyciskiem

Przycisk zapalający i gaszący diodę LED.

```
#include <Arduino.h>

#define LED1_PIN 10
#define SW1_PIN 4

bool led_on = false;

void setup() {
  pinMode(LED1_PIN, OUTPUT);
  pinMode(SW1_PIN, INPUT);
  digitalWrite(LED1_PIN, HIGH);
}
```

Arduino - sterowanie przyciskiem

Przycisk zapalający i gaszący diodę LED.

```
void loop() {
  if (digitalRead(SW1_PIN) == LOW)
  {
    delay(10);
    while (digitalRead(SW1_PIN) == LOW);
    delay(10);

    if (led_on == true)
    {
      led_on = false; digitalWrite(LED1_PIN, HIGH);
    }
    else
    {
      led_on = true; digitalWrite(LED1_PIN, LOW);
    }
  }
}
```

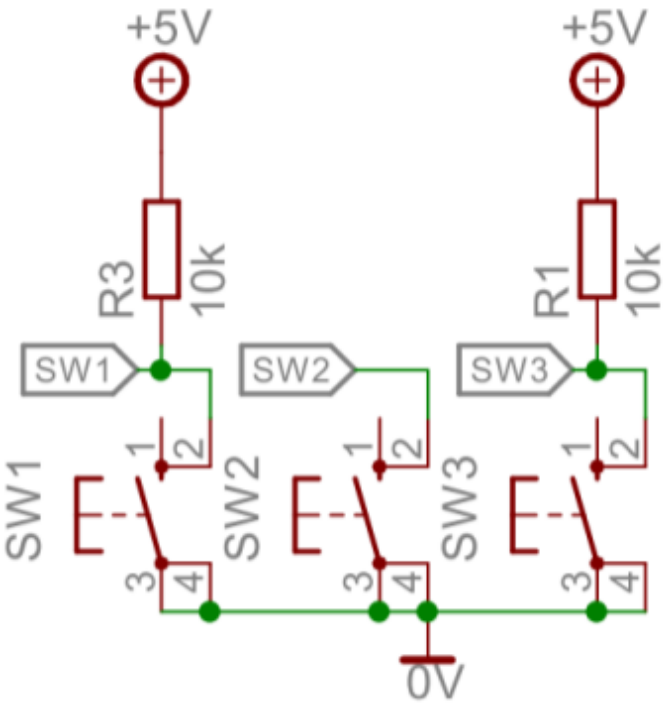
Arduino - sterowanie przyciskiem

- Eliminacja zjawiska drgania styków przycisku

```
delay(10);
while (digitalRead(SW1_PIN) == LOW);
delay(10);
```

- Kiedy styki przycisku stykają się lub rozłączają, zwarcie styków nie następuje do razu, ale może dochodzić do wielokrotnych krótkotrwałych połączeń i rozłączeń
- Dodanie opóźnienia delay(10); pozwala pominąć czas trwania tego zjawiska
- Pętla while powoduje, że program czeka w niej na zwolnienie przycisku - dioda LED1 jest zapalana i gaszona w momencie zwalniania przycisku

Arduino - rezystor podciągający



- Rezystor podciągający jest używany do ustabilizowania sygnału na wejściu cyfrowym układu logicznego
- Jeśli pin wejściowy Arduino nie jest podłączony ani do masy, ani do napięcia zasilania), to na wejściu mogą pojawiać się przypadkowe stany logiczne
- Rezystor podciągający łączy pin wejściowy (SW1, SW3) z napięciem zasilania (+5V), zapewniając stan logiczny HIGH, gdy przycisk (SW1, SW3) nie jest wciśnięty
- Gdy przycisk (SW1, SW3) zostanie wciśnięty, to pin wejściowy (SW1, SW3) zostanie podłączony do masy czyli sygnał na pinie zmieni się na LOW

Język C - tablica elementów

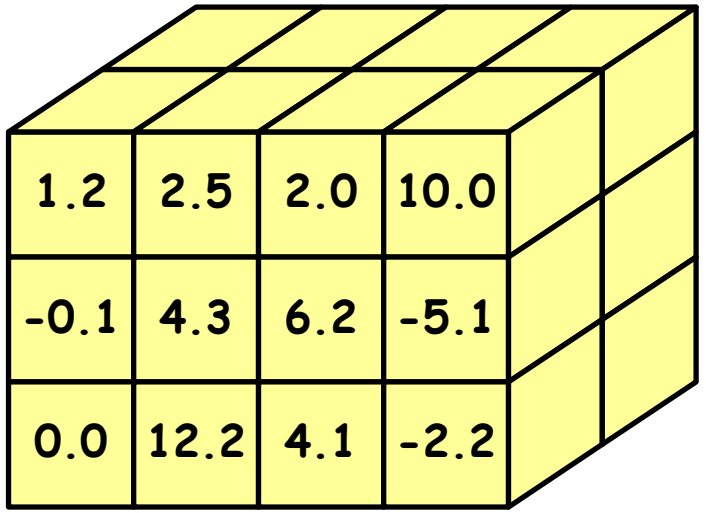
- Tablica - ciągły obszar pamięci, w którym umieszczone są elementy tego samego typu

wektor

5	3	-2	1	-4

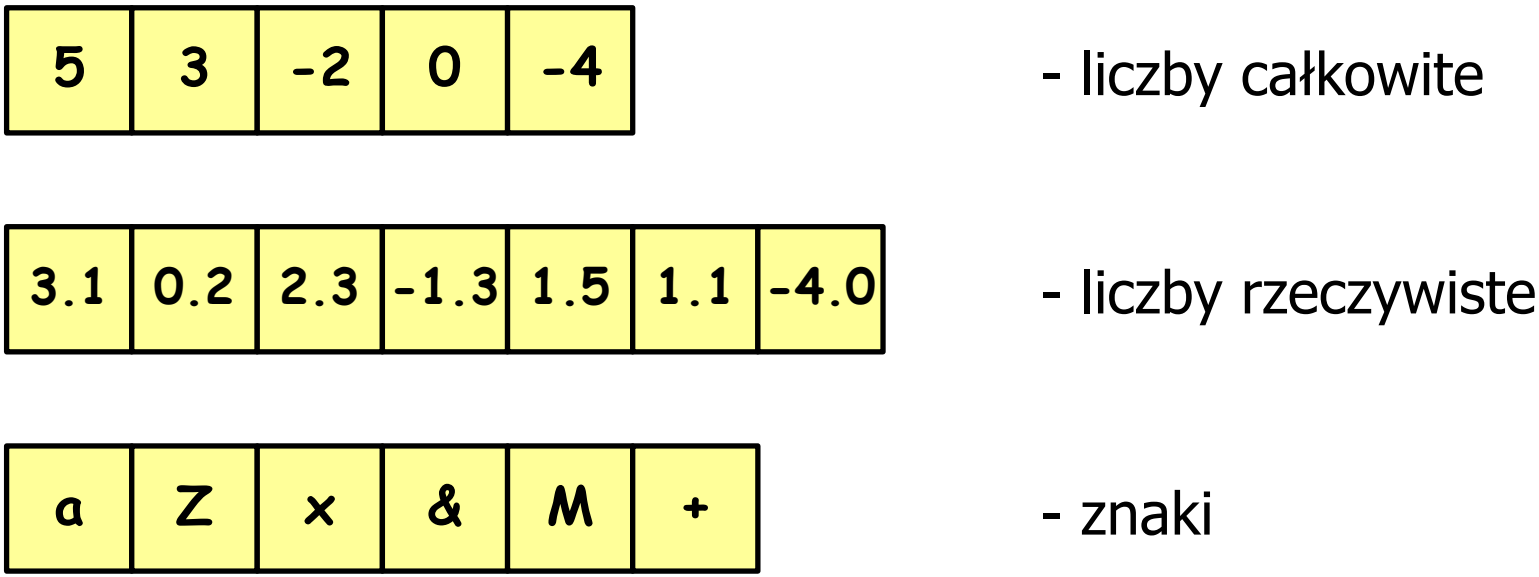
macierz

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

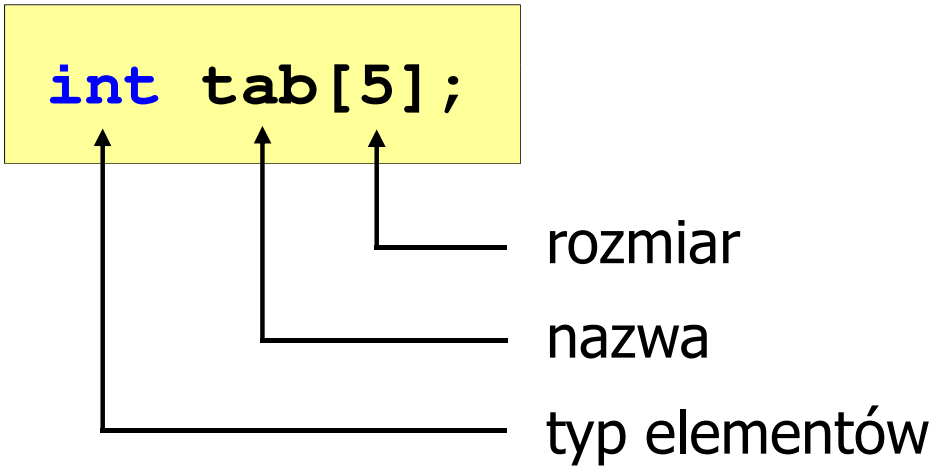


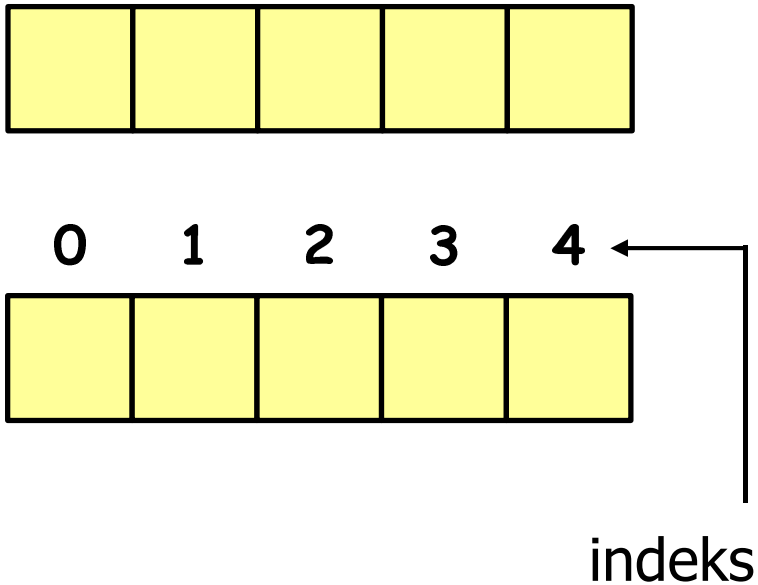
Język C - tablica jednowymiarowa

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



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;)

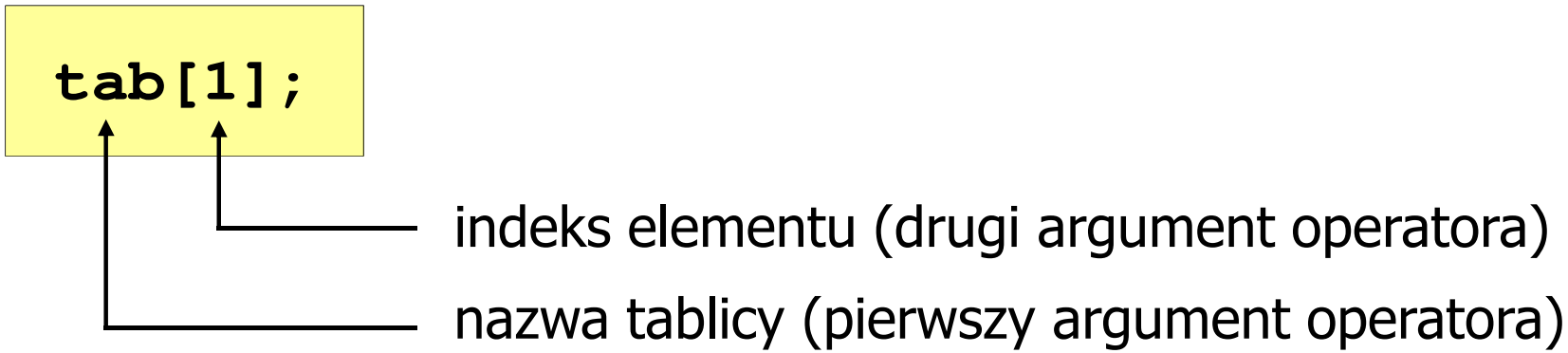
```
int tab[5];
```

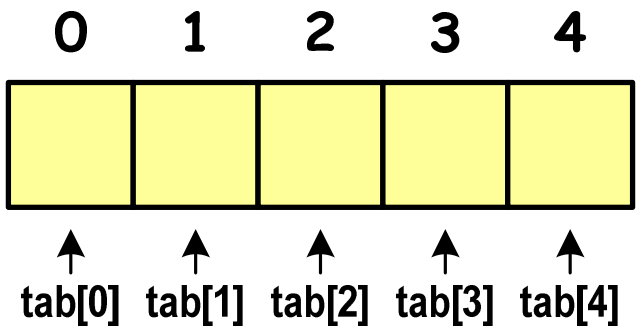
```
int tab[N];
```

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

Język C - odwołania do elementów tablicy

[ ] - dwuargumentowy operator indeksowania

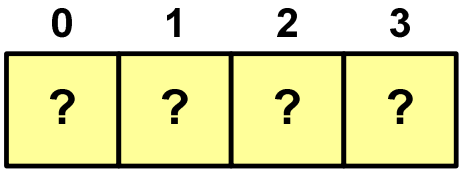




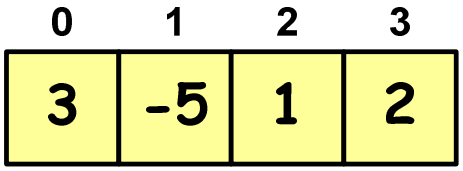
- indeks:
  - stała liczbowa, np. 0, 1, 10
  - nazwa zmiennej, np. i, idx
  - wyrażenie, np. i*j+5

Język C - odwołania do elementów tablicy

```
int tab[4];
```




```
tab[0] = 3;
tab[1] = -5;
tab[2] = 1;
tab[3] = 2;
```



- Każdy element tablicy traktowany jest tak samo jak zmienna typu int

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

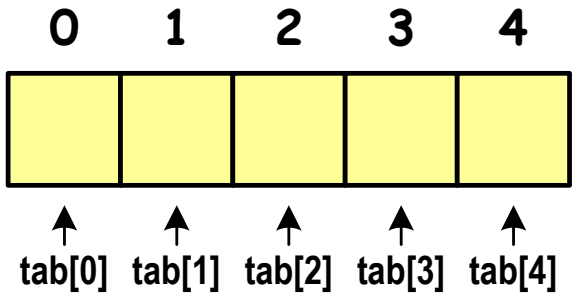
```
scanf("%d",&tab[1]);
```

Język C - odwołania do elementów tablicy

- Przy odwołaniach do elementów tablicy kompilator nie sprawdza poprawności indeksów

```
int tab[5];

tab[5] = 10;
```



- błąd!!! - nie istnieje element tab[5]

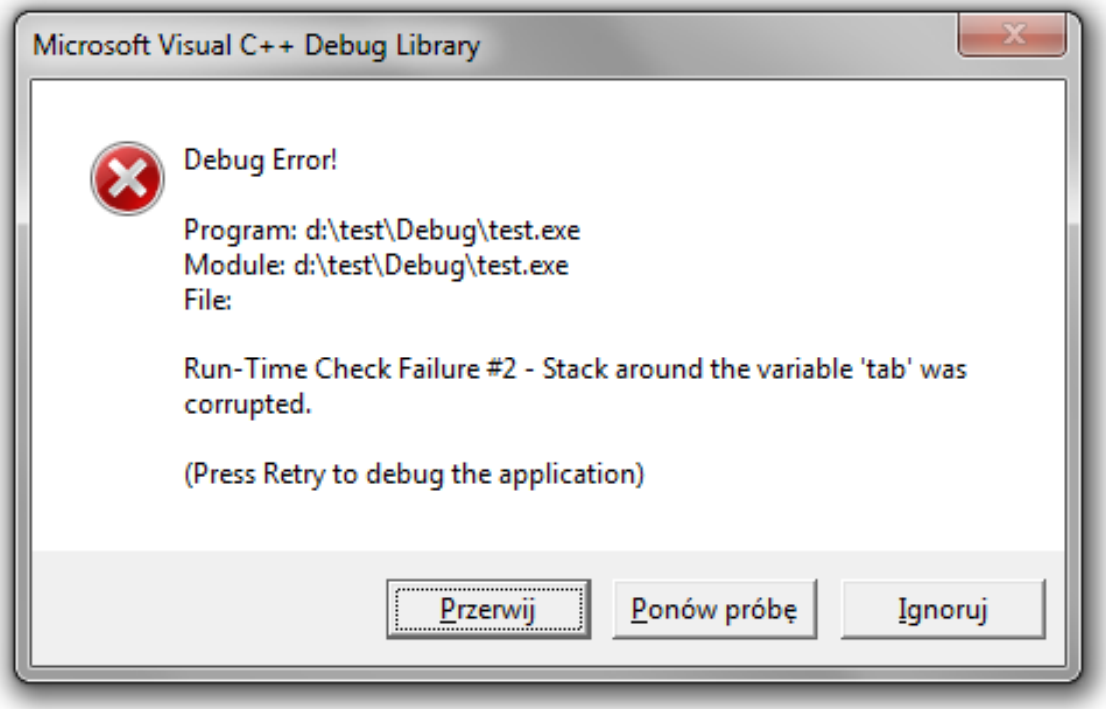
- Kompilator nie zasygnalizuje błędu
- Program wykona operację
- Środowisko programistyczne może zasygnalizować problem

Język C - odwołania do elementów tablicy

- Przy odwołaniach do elementów tablicy kompilator nie sprawdza poprawności indeksów

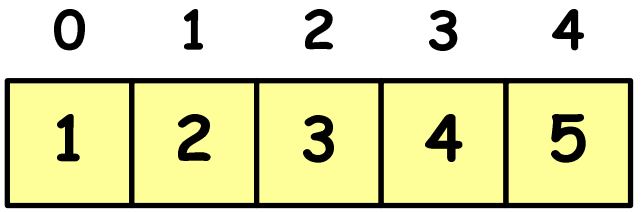
```
int tab[5];

tab[5] = 10;
```

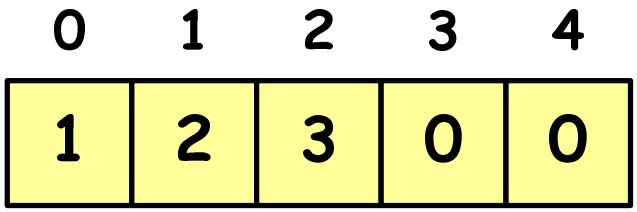


Język C - inicjalizacja tablicy jednowymiarowej

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



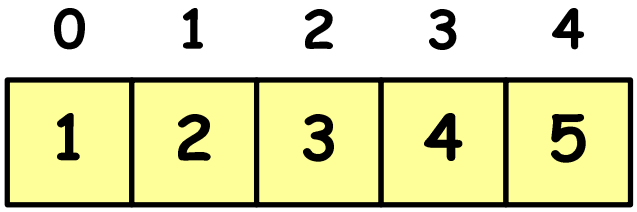

```
int tab[5] = {1,2,3};
```




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

- błąd kompilacji

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

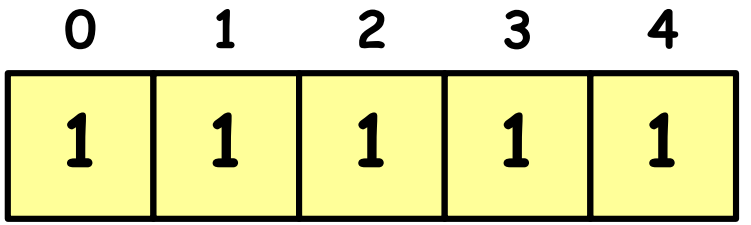


Język C - odwołania do elementów tablicy

- 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;
```




```
int tab[5], i;

for (i=0; i<5; i++)
   tab[i] = 1;
```

Język C - operacje na dużej ilości danych (tablica)

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

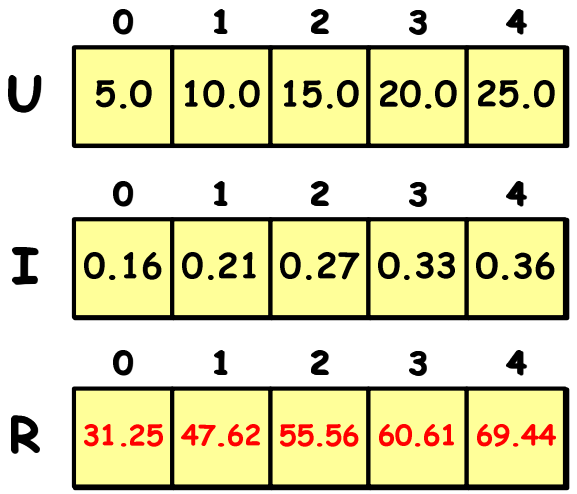
```c
#include <stdio.h>

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;
}
```



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>
```

Koniec wykładu nr 5

Dziękuję za uwagę!