Politechnika Wrocławska

Wstęp do programowania

INP001213Wcl

rok akademicki 2018/19

semestr zimowy

Wykład 3

Karol Tarnowski

karol.tarnowski@pwr.edu.pl

A-1 p. 411B

Politechnika Wrocławska

Plan prezentacji

- Arytmetyka komputerowa
- Typy danych w języku C

- Algorytmy iteracyjne
- Instrukcje sterujące języka C

Na podstawie:

- D. Kincaid, W. Cheney, *Analiza numeryczna*
- M. M. Sysło, *Algorytmy*
- G. Perry, D. Miller, *Język C Programowanie dla początkujących*

Arytmetyka komputerowa
Zapis liczb w różnych układach

$814{,}72 =$

$= 8\times10^{2} + 1\times10^{1} + 4\times10^{0} + 7\times10^{-1} + 2\times10^{-2}$

$(1110{,}10100)_2 =$

$= 1\times2^{3} + 1\times2^{2} + 1\times2^{1} + 0\times2^{0} + 1\times2^{-1} +$

$0\times2^{-2} + 1\times2^{-3} + 0\times2^{-4} + 0\times2^{-5} =$

$= 8 + 4 + 2 + 0{,}5 + 0{,}125 = (14{,}625)_{10}$

$\varphi = 1{,}618033988\ 7\ldots$

Politechnika Wrocławska

Arytmetyka komputerowa

Zapis liczb w różnych układach

$$\frac{1}{2} = (0,5)_{10} = (0,1)_2$$

$$\frac{1}{10} = (0,1)_{10} = (0,00011001100\ldots)_2$$

$$\frac{1}{3} = (0,33333\ldots)_{10} = (0,01010101\ldots)_2$$

Politechnika Wrocławska

Typy danych

- całkowite
- zmiennoprzecinkowe

Politechnika Wrocławska

Typy danych

Liczby całkowite

- relacje między typami całkowitymi:

`long long int ≥ long int ≥ int ≥ short int ≥ char`

- minimalny rozmiar zmiennej:
  - **`char`** 8 bitów,
  - **`short int`** oraz **`int`** 16 bitów,
  - **`long int`** 32 bity,
  - **`long long int`** 64 bity.

Politechnika Wrocławska

Typy danych

Liczby całkowite signed/unsigned

- **`char`** może być **`signed`** lub **`unsigned`**

- **`signed char`** taki sam rozmiar jak **`char`** ale ze znakiem,
  przedział co najmniej [−127, +127]

- **`unsigned char`** taki sam rozmiar jak **`char`** ale ze znakiem,
  przedział co najmniej [0, +255]

Politechnika Wrocławska

Typy danych

Liczby całkowite signed/unsigned

- `char`
- `signed char`
- `unsigned char`
- `short int`
- `signed short int`
- `unsigned short int`
- `int`
- `signed int`
- `unsigned int`
- `long int`
- `signed long int`
- `unsigned long int`
- `long long int`
- `signed long long int`
- `unsigned long long int`

Typy danych

limits.h

Start here | main.c

```c
/*Wstęp do programowania*/
/*Program demonstruje stale z pliku
  nagłówkowego limits.h*/

#include <stdio.h>
#include <limits.h>

int main(){
    printf("%10s %21d\n","CHAR_BIT",CHAR_BIT);
    printf("%10s %21d\n","CHAR_MIN",CHAR_MIN);
    printf("%10s %21d\n","CHAR_MAX",CHAR_MAX);
    printf("%10s %21d\n","INT_MIN", INT_MIN);
    printf("%10s %21d\n","INT_MAX", INT_MAX);
    printf("%10s %21ld\n","LONG_MIN",LONG_MIN);
    printf("%10s %21ld\n","LONG_MAX",LONG_MAX);
    printf("%10s %21lld\n","LLONG_MIN",LLONG_MIN);
    printf("%10s %21lld\n","LLONG_MAX",LLONG_MAX);
    printf("%10s %21d\n","SCHAR_MIN",SCHAR_MIN);
    printf("%10s %21d\n","SCHAR_MAX",SCHAR_MAX);
    printf("%10s %21d\n","SHRT_MIN",SHRT_MIN);
    printf("%10s %21d\n","SHRT_MAX",SHRT_MAX);
    printf("%10s %21d\n","UCHAR_MAX",UCHAR_MAX);
    printf("%10s %21u\n","UINT_MAX",UINT_MAX);
    printf("%10s %21lu\n","ULONG_MAX",ULONG_MAX);
    printf("%10s %21d\n","USHRT_MAX",USHRT_MAX);
}

```

Politechnika Wrocławska

Typy danych

`limits.h`

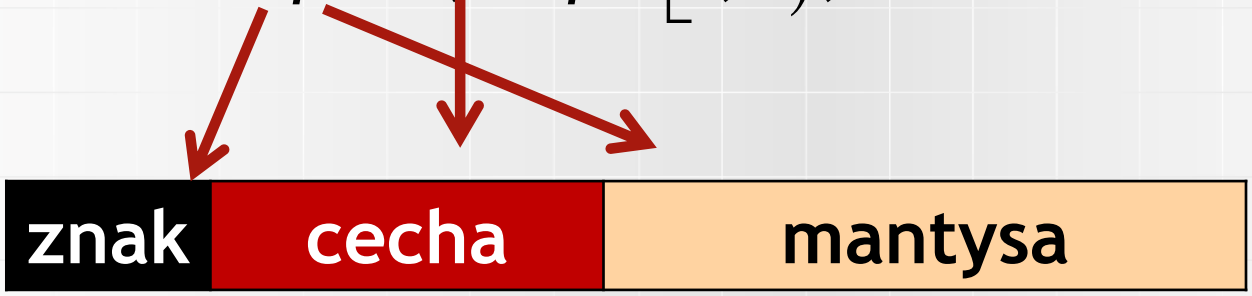
CHAR_BIT	8
CHAR_MIN	-128
CHAR_MAX	127
INT_MIN	-2147483648
INT_MAX	2147483647
LONG_MIN	-2147483648
LONG_MAX	2147483647
LLONG_MIN	-9223372036854775808
LLONG_MAX	9223372036854775807
SCHAR_MIN	-128
SCHAR_MAX	127
SHRT_MIN	-32768
SHRT_MAX	32767
UCHAR_MAX	255
UINT_MAX	4294967295
ULONG_MAX	4294967295
USHRT_MAX	65535

Typy danych

Liczby zmiennoprzecinkowe

$$x = \pm r \times 10^n, \quad r \in [1,10), \quad n \in C$$

$$x = \pm q \times 2^n, \quad q \in [1,2), \quad n \in C$$



$$x = (-1)^s \cdot (1, q_1 q_2 q_3 \ldots) \times 2^{n-b}$$

bias
przesunięcie

Politechnika Wrocławska

Typy danych

Liczby zmiennoprzecinkowe

- rozmiar i zachowanie zależy od implementacji
- `long double ≥ double ≥ float`
- standard IEEE 754 określa arytmetykę liczb pojedynczej (32 bity) oraz podwójnej (64 bity) precyzji

Arytmetyka komputerowa

Reprezentacja zmiennoprzecinkowa liczb

Liczby pojedynczej precyzji (przesunięcie 127)

z	c	c	c	c	c	c	c	c	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2

Liczby podwójnej precyzji (przesunięcie 1023)

z	c	m
1	2-12	13-64

8-bitowa liczba zmiennopozycyjna (przesunięcie 3)

z	c	c	c	m	m	m	m
1	2	3	4	5	6	7	8

Arytmetyka komputerowa

Reprezentacja zmiennoprzecinkowa liczb

0 0 1 1 0 0 0 0	+1. 0 0 0 0	$\times 2^{3-3}$	= 1,
0 0 1 1 1 0 0 0	+1. 1 0 0 0	$\times 2^{3-3}$	= 1,5
0 0 1 1 0 0 0 1	+1. 0 0 0 1	$\times 2^{3-3}$	= 1,0625 $=1+2^{-4}$
0 0 1 0 0 0 0 0	+1. 0 0 0 0	$\times 2^{2-3}$	= 0,5
0 0 0 1 0 0 0 0	+1. 0 0 0 0	$\times 2^{1-3}$	= 0,25 $= 2^{-2}$
0 1 1 0 0 0 0 0	+1. 0 0 0 0	$\times 2^{6-3}$	= 8,0
0 1 1 0 1 1 1 1	+1. 1 1 1 1	$\times 2^{6-3}$	= 15,5 $= 2^{3}(2-2^{-4})$
1 1 0 1 0 0 1 0	-1. 0 0 1 0	$\times 2^{5-3}$	= -4,5

Arytmetyka komputerowa

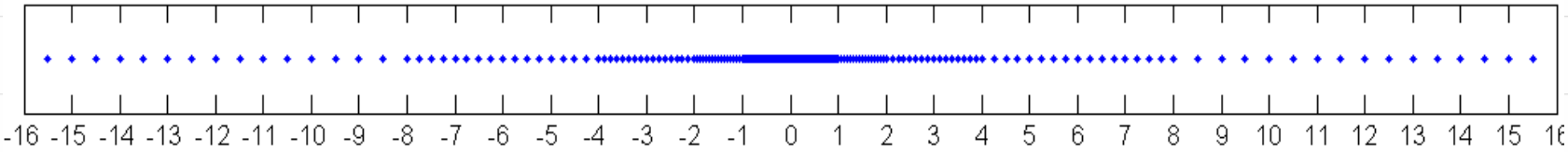
Reprezentacja zmiennoprzecinkowa liczb

0	0	0	0	0	0	0	0	= +0,
1	0	0	0	0	0	0	0	= -0,
0	1	1	1	0	0	0	0	= +Inf (infinity)
1	1	1	1	0	0	0	0	= -Inf (infinity)
0	0	0	0	1	0	0	0	$+0.1000 \times 2^{1-3}$ = 0,125
0	0	0	0	0	0	0	1	$+0.0001 \times 2^{1-3}$ = 0,015625 $=2^{-2}\times 2^{-4}$
0	1	1	1	1	0	0	0	= NaN (not a number)

Arytmetyka komputerowa

Reprezentacja zmiennoprzecinkowa liczb

Rozmieszczenie liczb zmiennopozycyjnych



- precyzja arytmetyki (epsilon maszynowy ε) $0{,}0625 = 2^{-4}$
- największa liczba zmiennopozycyjna $15{,}5 = (2-2^{-4})2^3$
- najmniejsza liczba zmiennopozycyjna
  - znormalizowana $0{,}25 = 2^{-2}$
  - zdenormalizowana $0{,}015625 = 2^{-6}$

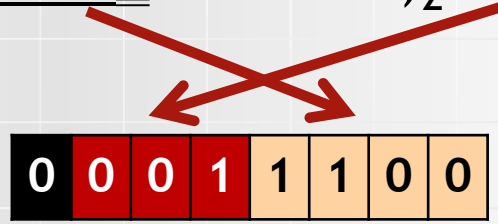
Arytmetyka komputerowa
Reprezentacja zmiennoprzecinkowa liczb

Jak wygląda reprezentacja liczby $^4/_9$ w rozważanej arytmetyce?

$^4/_9 = (0{,}0111000(111000)\ldots)_2$

po normalizacji

$^4/_9 = (1{,}11000111000\ldots)_2 \times 2^{-2} = (1{,}11000111000\ldots)_2 \times 2^{1-3}$



$fl(^4/_9) = 1{,}1100 \times 2^{-2} = (1+0{,}5+0{,}25)\times 0{,}25 = 0{,}4375$

błąd względny $|\delta| \leq \frac{\varepsilon}{2}$

Arytmetyka komputerowa

Działania na liczbach zmiennoprzecinkowych

Wynikiem operacji matematycznych na liczbach maszynowych zwykle nie jest liczba maszynowa. Przyjmujemy, że po wykonaniu działania mantysa jest normalizowana, a cecha odpowiednio korygowana.

W celu ilustracji rozpatrzymy arytmetykę liczb dziesiętnych z mantysą pięciocyfrową.
Niech $x = 9{,}7541\times10^{2}$, $y = 2{,}7849\times10^{4}$, wtedy

$x + y = 2{,}882441000\times10^{4}$, $fl(x + y) = 2{,}8824\times10^{4}$, $\delta = 1{,}43\times10^{-5}$

$x - y = -2{,}687359000\times10^{4}$, $fl(x - y) = -2{,}6874\times10^{4}$, $\delta = 1{,}53\times10^{-5}$

$x \times y = 2{,}716419309\times10^{7}$, $fl(x \times y) = 2{,}7164\times10^{7}$, $\delta = 7{,}1\times10^{-6}$

$x / y = 3{,}502495601\times10^{-2}$, $fl(x / y) = 3{,}5025\times10^{-2}$, $\delta = 1{,}3\times10^{-6}$

Arytmetyka komputerowa

Działania na liczbach zmiennoprzecinkowych

Przykładem sytuacji, w której mogą pojawić się duże błędy względne jest odejmowanie bliskich sobie liczb

$$
\begin{aligned}
x &= 8{,}14786922317801 5,\\
fl(x) &= 8{,}14786,\\
y &= 8{,}14723686393179 0,\\
fl(y) &= 8{,}14724,\\
x - y &= 0{,}000632359246225,\\
fl(x) - fl(y) &= 0{,}00062,\\
fl(fl(x) - fl(y)) &= 6{,}2000\times 10^{-4}
\end{aligned}
$$

$$\left|\frac{(x-y)-fl\left[fl(x)-fl(y)\right]}{x-y}\right| = \left|\frac{0{,}000632359246225-0{,}00062}{0{,}000632359246225}\right| \approx 0{,}0195$$

Arytmetyka komputerowa

Działania na liczbach zmiennoprzecinkowych

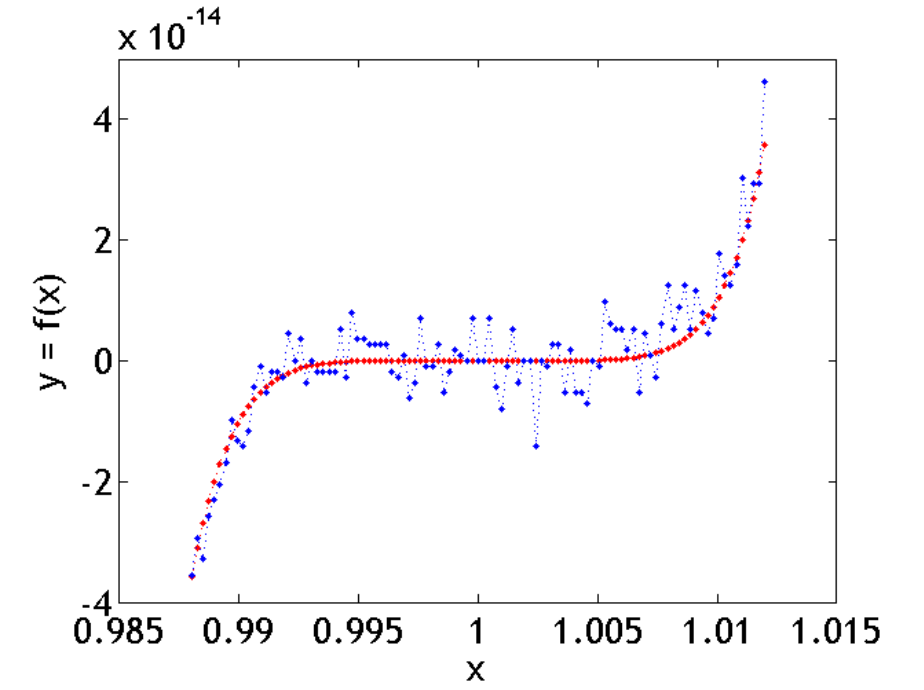
$$y \leftarrow \sqrt{x^2+1}-1$$

$$\sqrt{x^2+1}-1=\left(\sqrt{x^2+1}-1\right)\frac{\sqrt{x^2+1}+1}{\sqrt{x^2+1}+1}=\frac{x^2+1-1}{\sqrt{x^2+1}+1}=\frac{x^2}{\sqrt{x^2+1}+1}$$

$$y \leftarrow \frac{x^2}{\sqrt{x^2+1}+1}$$

Arytmetyka komputerowa

Działania na liczbach zmiennoprzecinkowych



$$f(x) = (x-1)^7 = x^7-7x^6+21x^5-35x^4+35x^3-21x^2+7x-1$$

Typy danych

Liczby zmiennoprzecinkowe - float.h

Start here | **main01.c** | main02.c

```c
/*Wstęp do programowania*/
/*Program demonstruje stałe z pliku
  nagłówkowego float.h*/

#include <stdio.h>
#include <float.h>

int main(){
    char format_int[] = "%14s %12d\n";
    char format_flt[] = "%14s %12g\n";
    printf(format_int,"FLT_RADIX",FLT_RADIX);
    printf(format_int,"FLT_DIG",FLT_DIG);
    printf(format_flt,"FLT_EPSILON",FLT_EPSILON);
    printf(format_int,"FLT_MANT_DIG",FLT_MANT_DIG);
    printf(format_flt,"FLT_MAX",FLT_MAX);
    printf(format_int,"FLT_MAX_EXP",FLT_MAX_EXP);
    printf(format_int,"FLT_MAX_10_EXP",FLT_MAX_10_EXP);
    printf(format_flt,"FLT_MIN",FLT_MIN);
    printf(format_int,"FLT_MIN_EXP",FLT_MIN_EXP);
    printf(format_int,"FLT_MIN_10_EXP",FLT_MIN_10_EXP);
}

```

Typy danych

Liczby zmiennoprzecinkowe - float.h

Start here | main01.c | main02.c

```
1  /*Wstęp do programowania*/
2  /*Program demonstruje stałe z pliku
3    nagłówkowego float.h*/
4
5  #include <stdio.h>
6  #include <float.h>
7
8  int main(){
9      char format_int[] = "%14s %12d\n";
       = "%14s %12g\n";
       "FLT_RADIX",FLT_RADIX);
       "FLT_DIG",FLT_DIG);
       "FLT_EPSILON",FLT_EPSILON);
       "FLT_MANT_DIG",FLT_MANT_DIG);
       "FLT_MAX",FLT_MAX);
       "FLT_MAX_EXP",FLT_MAX_EXP);
       "FLT_MAX_10_EXP",FLT_MAX_10_EXP);
       "FLT_MIN",FLT_MIN);
       "FLT_MIN_EXP",FLT_MIN_EXP);
       "FLT_MIN_10_EXP",FLT_MIN_10_EXP);
```

```
     FLT_RADIX             2
       FLT_DIG             6
   FLT_EPSILON 1.19209e-007
  FLT_MANT_DIG            24
       FLT_MAX 3.40282e+038
   FLT_MAX_EXP           128
FLT_MAX_10_EXP            38
       FLT_MIN 1.17549e-038
   FLT_MIN_EXP          -125
FLT_MIN_10_EXP           -37
```

Typy danych

Liczby zmiennoprzecinkowe - float.h

Start here | main01.c | **main02.c**

```c
/*Wstęp do programowania*/
/*Program demonstruje stałe z pliku
  nagłówkowego float.h*/

#include <stdio.h>
#include <float.h>

int main(){
    char format_int[] = "%14s %12d\n";
    char format_flt[] = "%14s %12g\n";
    printf(format_int,"DBL_RADIX",FLT_RADIX);
    printf(format_int,"DBL_DIG",DBL_DIG);
    printf(format_flt,"DBL_EPSILON",DBL_EPSILON);
    printf(format_int,"DBL_MANT_DIG",DBL_MANT_DIG);
    printf(format_flt,"DBL_MAX",DBL_MAX);
    printf(format_int,"DBL_MAX_EXP",DBL_MAX_EXP);
    printf(format_int,"DBL_MAX_10_EXP",DBL_MAX_10_EXP);
    printf(format_flt,"DBL_MIN",DBL_MIN);
    printf(format_int,"DBL_MIN_EXP",DBL_MIN_EXP);
    printf(format_int,"DBL_MIN_10_EXP",DBL_MIN_10_EXP);
}
```

Typy danych

Liczby zmiennoprzecinkowe - float.h

Start here | main01.c | **main02.c**

```
1  /*Wstęp do programowania*/
2  /*Program demonstruje stałe z pliku
3    nagłówkowego float.h*/
4
5  #include <stdio.h>
6  #include <float.h>
7
8  int main(){
```

```
= "%14s %12d\n";
= "%14s %12g\n";
,"DBL_RADIX",FLT_RADIX);
,"DBL_DIG",DBL_DIG);
,"DBL_EPSILON",DBL_EPSILON);
,"DBL_MANT_DIG",DBL_MANT_DIG);
,"DBL_MAX",DBL_MAX);
,"DBL_MAX_EXP",DBL_MAX_EXP);
,"DBL_MAX_10_EXP",DBL_MAX_10_EXP);
,"DBL_MIN",DBL_MIN);
,"DBL_MIN_EXP",DBL_MIN_EXP);
,"DBL_MIN_10_EXP",DBL_MIN_10_EXP);
```

```
     DBL_RADIX            2
       DBL_DIG           15
   DBL_EPSILON 2.22045e-016
  DBL_MANT_DIG           53
       DBL_MAX 1.79769e+308
   DBL_MAX_EXP         1024
DBL_MAX_10_EXP          308
       DBL_MIN 2.22507e-308
   DBL_MIN_EXP        -1021
DBL_MIN_10_EXP         -307
```

Politechnika Wrocławska

Algorytmy iteracyjne

- iteracja – powtarzanie pewnych instrukcji (lub ciągów instrukcji)
- przykładowo: wyznaczanie maksymalnego elementu w zbiorze wymaga przejrzenia wszystkich elementów zbioru

Politechnika Wrocławska

Algorytmy iteracyjne

Liniowe przeszukiwanie

- Czy we wskazanym zbiorze znajduje się poszukiwany element?

- Dane: ciąg liczb w tablicy *t* rozmiaru *n*, poszukiwana liczba *x*
- Wynik: indeks pozycji w tablicy *t*, na której znajduje się element *x*, lub -1 jeśli nie ma poszukiwanego elementu

Politechnika Wrocławska

Algorytmy iteracyjne

Liniowe przeszukiwanie

- Dane: ciąg liczb w tablicy *t* rozmiaru *n*, poszukiwana liczba *x*
- Wynik: indeks pozycji w tablicy *t*, na której znajduje się element *x*, lub -1 jeśli nie ma poszukiwanego elementu

- dla *i* = 0, 1, …, *n*-1
  - jeśli *t*[*i*] == *x* zwróć *i*
- zwróć -1

Algorytmy iteracyjne

Obliczanie średniej

- Dane: ciąg liczb w tablicy *t* rozmiaru *n*, poszukiwana liczba *x*
- Wynik: średnia wszystkich elementów

Algorytmy iteracyjne

Znajdowanie elementu maksymalnego

- Dane: ciąg liczb w tablicy *t* rozmiaru *n*
- Wynik: wartość elementu maksymalnego

- wybierz jako maksimum (kandydata na maksimum) dowolny element
- dla wszystkich pozostałych elementów:
  - jeśli przeglądany element jest większy niż aktualne maksimum zaktualizuj maksimum
- zwróć maksimum

Politechnika Wrocławska

Algorytmy iteracyjne

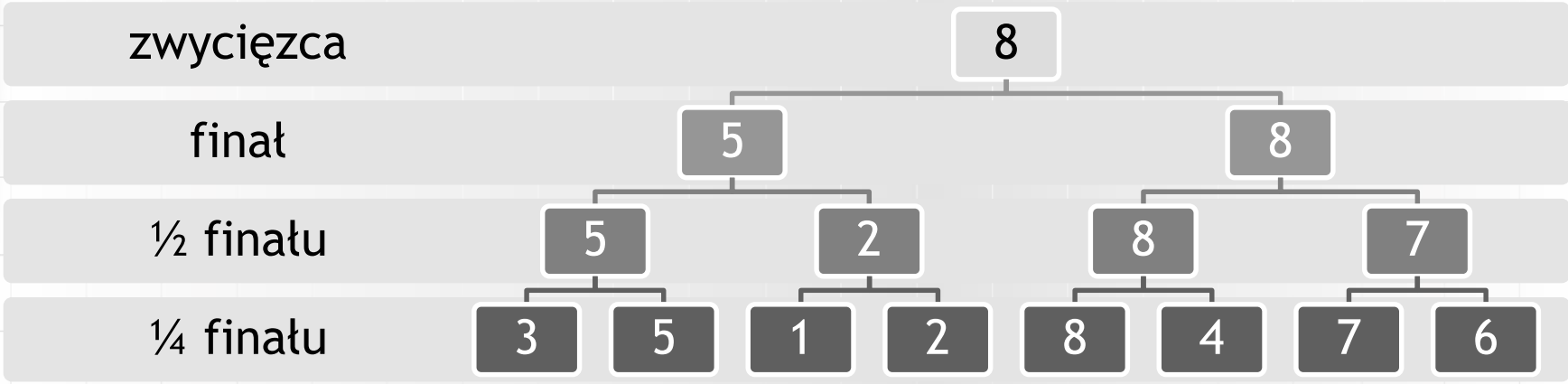
Znajdowanie elementu maksymalnego

- Liczba porównań wynosi n-1
- Czy można mniej?

Algorytmy iteracyjne

Znajdowanie elementu maksymalnego

- Rozważmy metodę pucharową



Politechnika Wrocławska

Algorytmy iteracyjne

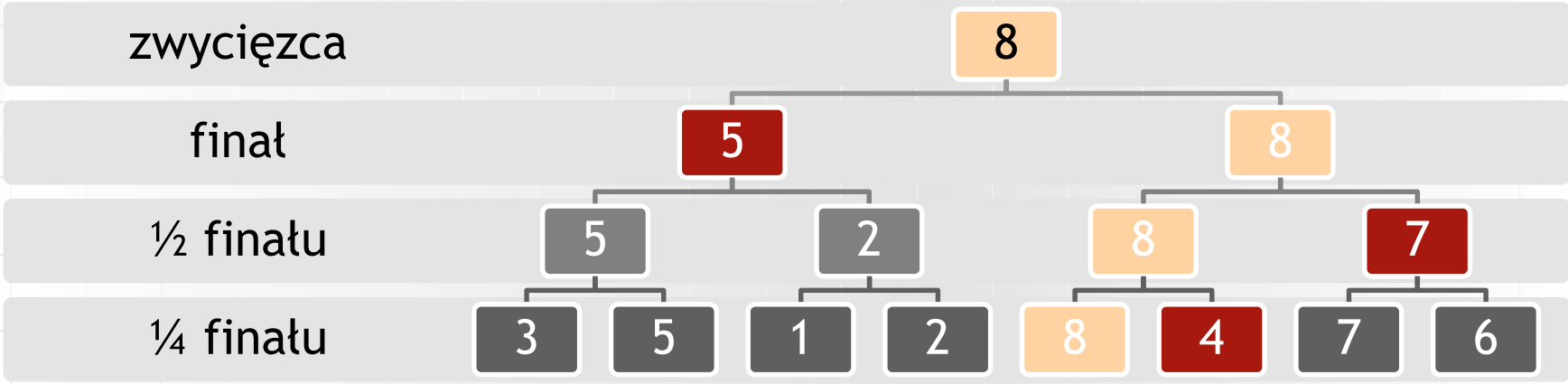
Kompletowanie podium

- Jak wybrać drugiego zawodnika turnieju?
- Druga drabinka?
  - n-1 porównań (wybór zwycięzcy)
  - n-2 porównania (wybór drugiego zawodnika)
  - łącznie $2n$-3
- Czy można mniej?

Algorytmy iteracyjne

Kompletowanie podium

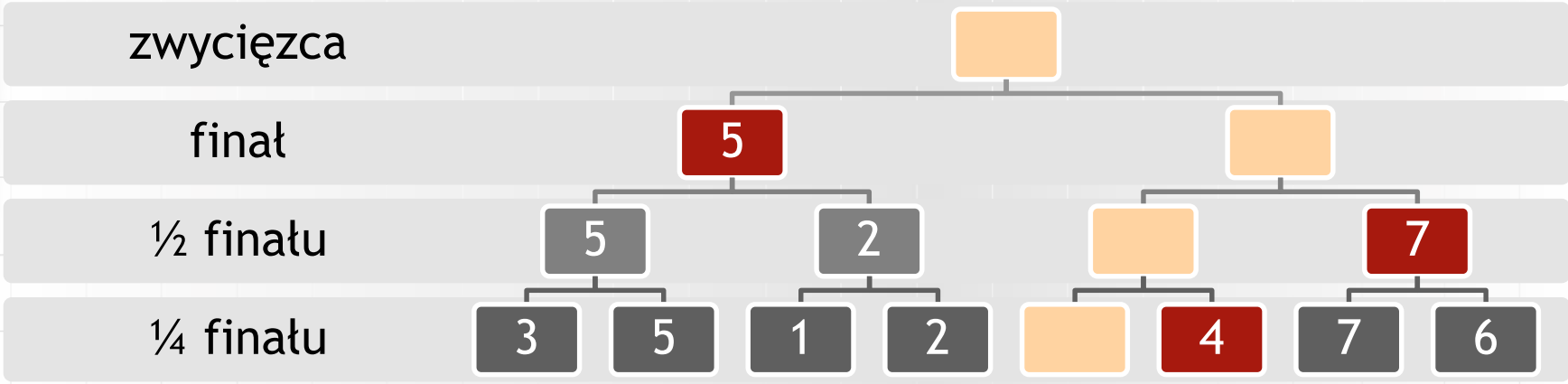
- Modyfikacja metody pucharowej



Algorytmy iteracyjne

Kompletowanie podium

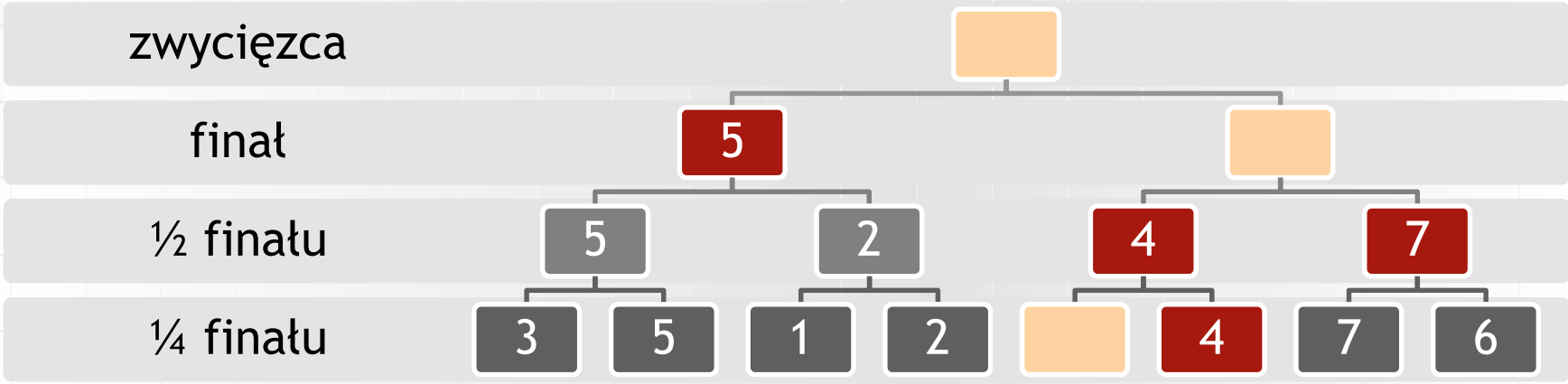
- Modyfikacja metody pucharowej



Algorytmy iteracyjne

Kompletowanie podium

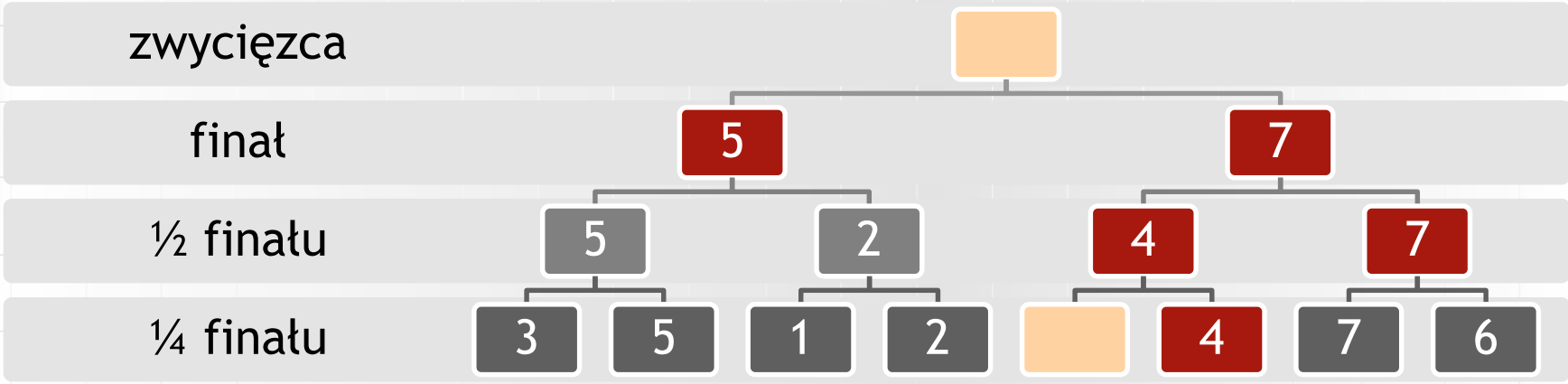
- Modyfikacja metody pucharowej



Algorytmy iteracyjne

Kompletowanie podium

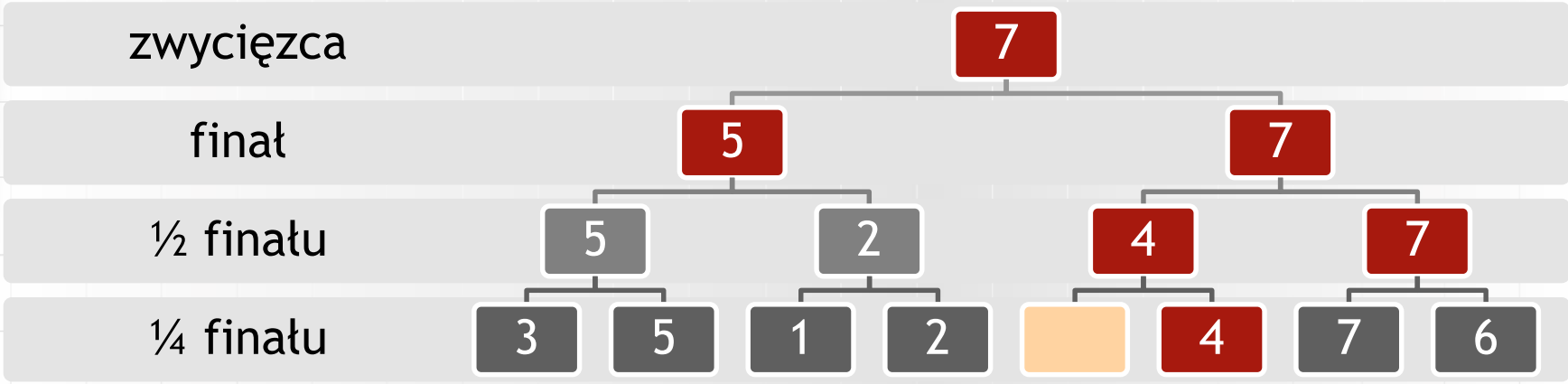
- Modyfikacja metody pucharowej



Algorytmy iteracyjne

Kompletowanie podium

- Modyfikacja metody pucharowej



Algorytmy iteracyjne

Kompletowanie podium

- Liczba porównań:
  - n-1 (wybór zwycięzcy),
  - $\log_2 n$-1 (wybór drugiego zawodnika)
  - łącznie $n + \log_2 n - 2$

Algorytmy iteracyjne

Jednoczesne znajdowanie maksimum i minimum

min								
	3	5	1	2	8	4	7	6
max								

Algorytmy iteracyjne

Jednoczesne znajdowanie maksimum i minimum

min	3							
	3	5	1	2	8	4	7	6
max		5						

Algorytmy iteracyjne

Jednoczesne znajdowanie maksimum i minimum

min	3		1					
	3	5	1	2	8	4	7	6
max		5		2				

Algorytmy iteracyjne

Jednoczesne znajdowanie maksimum i minimum

min	3		1			4		
	3	5	1	2	8	4	7	6
max		5		2	8			

Politechnika Wrocławska

Algorytmy iteracyjne

Jednoczesne znajdowanie maksimum i minimum

min	3		1			4		6
	3	5	1	2	8	4	7	6
max		5		2	8		7	

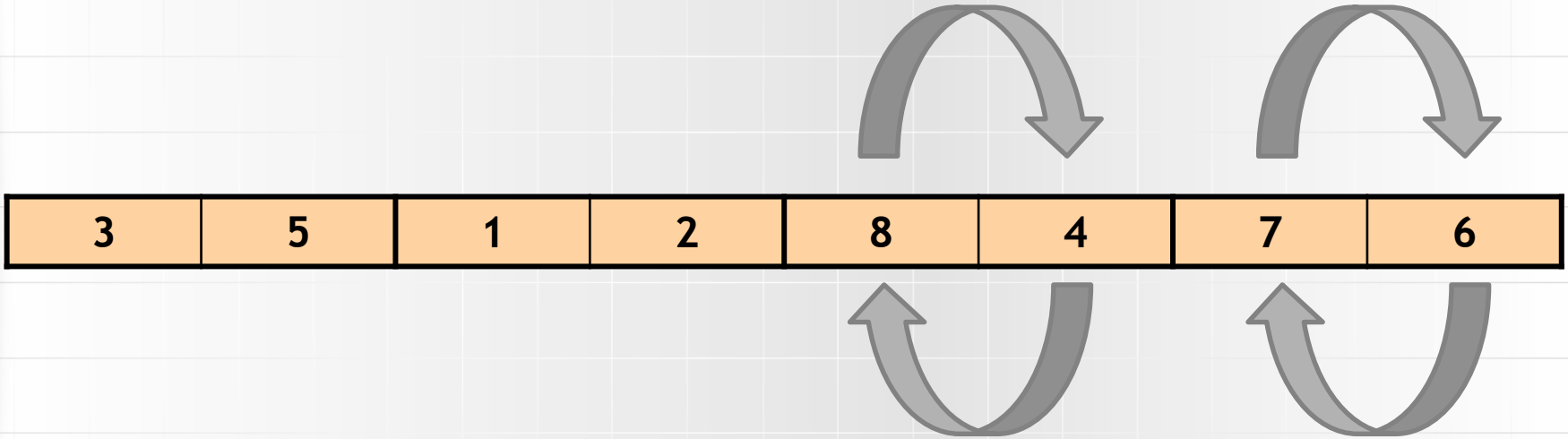
Algorytmy iteracyjne

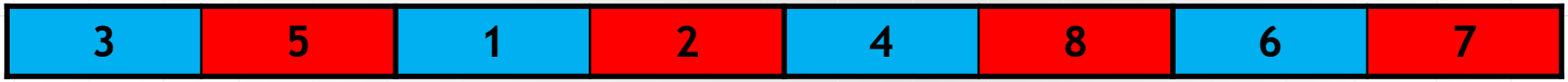
Jednoczesne znajdowanie maksimum i minimum

min	3		1			4	7
	3	5	1	2	8	4	7
max		5		2	8		7

Algorytmy iteracyjne

Jednoczesne znajdowanie maksimum i minimum





Politechnika Wrocławska

Algorytmy iteracyjne

Jednoczesne znajdowanie maksimum i minimum

- Liczba porównań dla parzystych *n*:
  - n/2 porównań par elementów
  - n/2 – 1 porównań (wybór maksimum)
  - n/2 – 1 porównań (wybór minimum)
  - łącznie 3n/2 – 2

Politechnika Wrocławska

Instrukcje sterujące

- Pętla **`while`**
- Pętla **`do...while`**
- Pętla **`for`**
- Instrukcje **`break`** i **`continue`**
- Instrukcja **`switch`**

Politechnika Wrocławska

Pętla `while`

- Ogólna postać

```
while(warunek)
{
   blok instrukcji C;
}
```

Politechnika Wrocławska

Pętla while

Start here | licznik.c

```c
/*licznik.c*/
/*Wstęp do programowania*/
/*na podstawie: G. Perry, D. Miller,
Język C Programowanie dla początkujących, Helion, 2014*/

#include <stdio.h>

main(){
    int licznik = 0;

    printf("Licznik ma wartosc %d.\n",++licznik);
    printf("Licznik ma wartosc %d.\n",++licznik);
    printf("Licznik ma wartosc %d.\n",++licznik);
    printf("Licznik ma wartosc %d.\n",++licznik);
    printf("Licznik ma wartosc %d.\n",--licznik);
    printf("Licznik ma wartosc %d.\n",--licznik);
    printf("Licznik ma wartosc %d.\n",--licznik);

    return 0;
}
```

Politechnika Wrocławska

Pętla while

Start here | licznik.c | **while.c**

```c
/*while.c*/
/*Wstęp do programowania*/
/*na podstawie: G. Perry, D. Miller,
Język C Programowanie dla początkujących, Helion, 2014*/

#include <stdio.h>

main(){
    int licznik = 0;

    while(licznik < 4){
        printf("Licznik ma wartosc %d.\n",++licznik);
    }
    while(licznik > 1){
        printf("Licznik ma wartosc %d.\n",--licznik);
    }

    return 0;
}
```

Politechnika Wrocławska

Pętla do...while

- Ogólna postać

```
do
{
   blok instrukcji C;
}
while(warunek);
```

Politechnika Wrocławska

Pętla do...while

Start here | licznik.c | while.c | **do_while.c**

```c
/*do_while.c*/
/*Karol Tarnowski*/
/*Wstęp do programowania*/
/*na podstawie: G. Perry, D. Miller,
Język C Programowanie dla początkujących, Helion, 2014*/

#include <stdio.h>

main(){
    int licznik = 0;

    do{
        printf("Licznik ma wartosc %d.\n",++licznik);
    }while(licznik < 4);

    do{
        printf("Licznik ma wartosc %d.\n",--licznik);
    }while(licznik > 1);

    return 0;
}
```

Politechnika Wrocławska

Pętla do...while

Start here | licznik.c | while.c | do_while.c | **do_while_scanf.c**

```
1  /*do_while_scanf.c*/
2  /*Wstęp do programowania*/
3  /*Program ilustruje wykorzystanie
4    pętli do-while do sprawdzenia
5    poprawności podawanych danych.*/
6
7  #include <stdio.h>
8
9  main(){
10    float a;
11
12    do{
13       printf("Podaj liczbe dodatnia: ");
14       scanf(" %f",&a);
15    }while(a <= 0);
16
17    printf("Liczba %g jest dodatnia, brawo!\n",a);
18
19    return 0;
20 }
```

Politechnika Wrocławska

Pętla `for`

- Ogólna postać

```
for(poczatek; test; krok)
{
   blok instrukcji C;
}
```

Politechnika Wrocławska

Pętla for

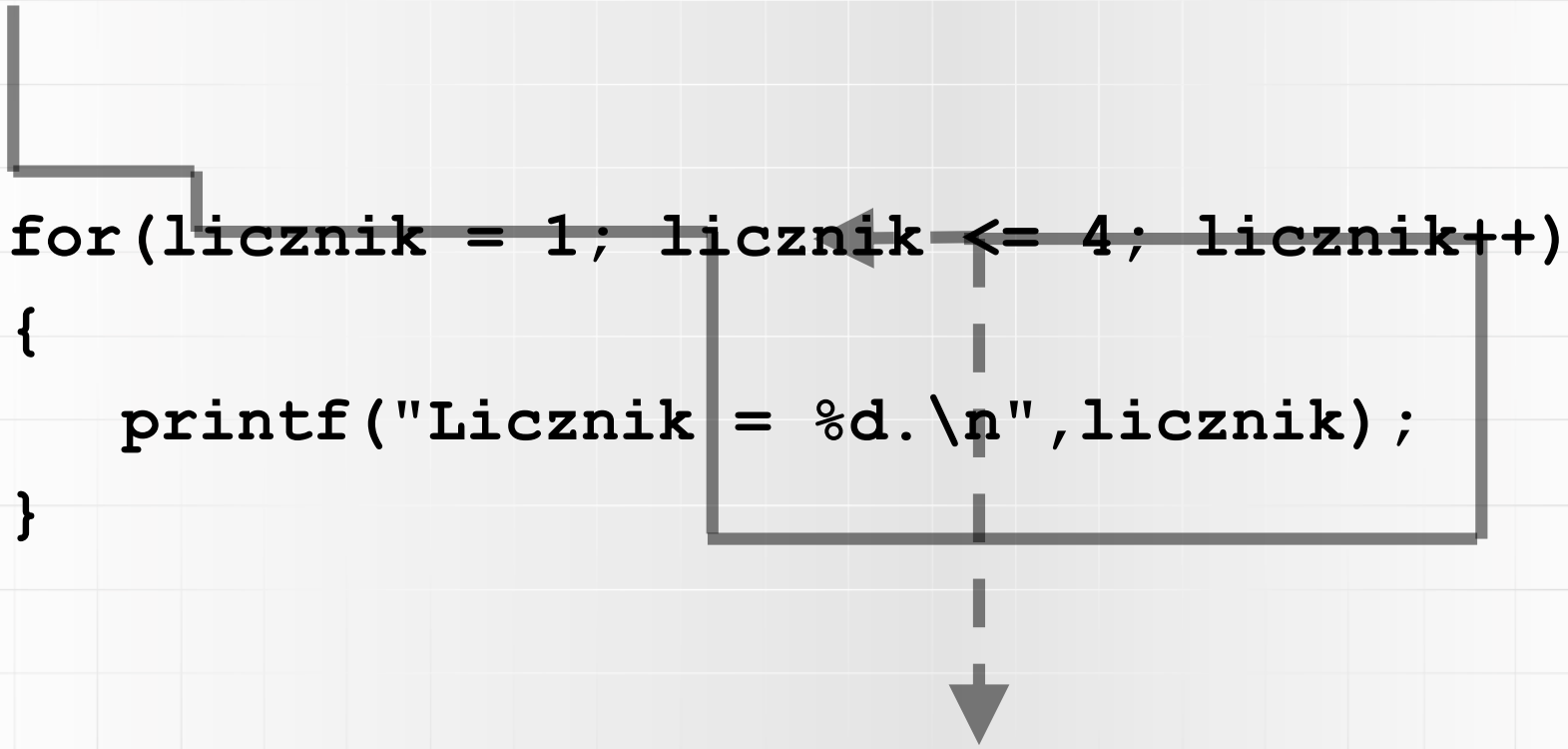
Start here | for.c

```c
/*for.c*/
/*Wstęp do programowania*/
/*na podstawie: G. Perry, D. Miller,
Język C Programowanie dla początkujących, Helion, 2014*/

#include <stdio.h>

main(){
    int licznik;

    for(licznik = 1; licznik <= 4; licznik++){
        printf("Licznik ma wartosc %d.\n",licznik);
    }

    for(licznik = 3; licznik >= 1; licznik--){
        printf("Licznik ma wartosc %d.\n",licznik);
    }

    return 0;
}
```

Politechnika Wrocławska

Pętla for

```
for(licznik = 1; licznik <= 4; licznik++)
{
    printf("Licznik = %d.\n",licznik);
}
```



Pętla for

Porównanie z pętlą while

```
licznik = 1;
while(licznik <= 4)
{
    printf("Licznik = %d.\n",licznik);
    licznik++;
}
```

Politechnika Wrocławska

Przerywanie pętli

- Do przerywania pętli można wykorzystać instrukcję **`break`**
- Po wykonaniu instrukcji **`break`** program przechodzi do wykonania kodu znajdującego się za pętlą
- Najczęściej instrukcja **`break`** występuje w połączeniu z instrukcją **`if`**

Politechnika Wrocławska

Instrukcja break

Start here | for_break.c

```
1  /*break.c*/
2  /*Wstęp do programowania*/
3  /*Program ilustruje użycie instrukcji break*/
4  #include <stdio.h>
5
6  main(){
7      printf("Zgadnij moja liczbe!\n");
8
9      int x = 17, y, n;
10
11     for(n = 5; n>0; n--){
12         printf("\nLiczba pozostalych prob: %d.\n",n);
13         printf("Podaj liczbe calkowita: ");
14         scanf("%d",&y);
15         if( y == x ){
16             printf("Brawo! %d to moja liczba.\n",x);
17             break;
18         }
19     }
20     return 0;
21 }
22
```

Politechnika Wrocławska

Instrukcja break

Start here | for_break.c

```
1  /*break.c*/
2  /*Wstęp do programowania*/
3  /*Program ilustruje użycie
4  #include <stdio.h>
5
6  main(){
7      printf("Zgadnij moja li
8
9      int x = 17, y, n;
10
11     for(n = 5; n>0; n--){
12         printf("\nLiczba poz
13         printf("Podaj liczbe
14         scanf("%d",&y);
15         if( y == x ){
16             printf("Brawo! %d
17             break;
18         }
19     }
20     return 0;
21 }
22
```

```
Zgadnij moja liczbe!

Liczba pozostalych prob: 5.
Podaj liczbe calkowita: 2

Liczba pozostalych prob: 4.
Podaj liczbe calkowita: 3

Liczba pozostalych prob: 3.
Podaj liczbe calkowita: 5

Liczba pozostalych prob: 2.
Podaj liczbe calkowita: 7

Liczba pozostalych prob: 1.
Podaj liczbe calkowita: 11
```

Instrukcja break

Start here | for_break.c

```c
/*break.c*/
/*Wstęp do programowania*/
/*Program ilustruje użycie
#include <stdio.h>

main(){
   printf("Zgadnij moja li

   int x = 17, y, n;

   for(n = 5; n>0; n--){
      printf("\nLiczba pozostalych prob: %d.\n",n);
      printf("Podaj liczbe calkowita: ");
      scanf("%d",&y);
      if( y == x ){
         printf("Brawo! %d to moja liczba.\n",x);
         break;
      }
   }
   return 0;
}
```

```
Zgadnij moja liczbe!

Liczba pozostalych prob: 5.
Podaj liczbe calkowita: 13

Liczba pozostalych prob: 4.
Podaj liczbe calkowita: 17
Brawo! 17 to moja liczba.
```

Przerywanie bieżącego wykonania pętli

- Do przerywania pętli można wykorzystać instrukcję **`continue`**
- Po wykonaniu instrukcji **`continue`** program przechodzi do kolejnego wykonania kodu pętli

Politechnika Wrocławska

Instrukcja continue

Start here | while_continue.c

```
1  /*while_continue.c*/
2  /*Wstęp do programowania*/
3  /*Program ilustruje użycie instrukcji continue*/
4  #include <stdio.h>
5
6  main(){
7      printf("Program oblicza srednia pieciu ocen.\n");
8
9      float ocena, suma = 0;
10     int licznik = 0;
11
```

Politechnika Wrocławska

Instrukcja continue

Start here | while_continue.c

```
12      while(licznik < 5){
13          printf("Podaj ocene: ");
14          scanf("%f",&ocena);
15          if(ocena != 2. && \
16              ocena != 3. && ocena != 3.5 && \
17              ocena != 4. && ocena != 4.5 && \
18              ocena != 5. && ocena != 5.5 )
19          {
20              printf("Podano nieprawidlowa ocene. \
21  Sprobuj jeszcze raz.\n");
22              continue;
23          }
24          suma += ocena;
25          licznik++;
26      }
27
28      printf("Srednia ocen to %g.",suma/licznik);
29
30      return 0;
31  }
```

Politechnika Wrocławska

Instrukcja switch

- Ogólna postać

```
switch(wyrazenie){
    case(wyrazenie1):
        {blok instrukcji C}
    case(wyrazenie2):
        {blok instrukcji C}
    //...
    default:
        {blok instrukcji C}
}
```

Politechnika Wrocławska

Instrukcja switch

Start here | switch.c

```
1  /*switch.c*/
2  /*Wstęp do programowania*/
3  /*Program ilustruje użycie instrukcji switch*/
4  #include <stdio.h>
5
6  main(){
7      char choice;
8
9      printf("Mozesz wybrac jedna opcje.\n");
10     printf("1. Pierwsza.\n");
11     printf("2. Druga.\n");
12     printf("3. Trzecia.\n");
13     printf("4. Czwarta. \n");
14     printf("Wybierz opcje:\n");
15     scanf("%d",&choice);
16
```

Politechnika Wrocławska

Instrukcja switch

Start here | switch.c

```
17      switch(choice){
18          case(1):
19              printf("Wybrales opcje pierwsza.\n");
20              break;
21          case(2):
22              printf("Wybrales opcje druga.\n");
23              break;
24          case(3):
25              printf("Wybrales opcje trzecia.\n");
26              break;
27          case(4):
28              printf("Wybrales opcje czwarta.\n");
29              break;
30          default:
31              printf("Nie ma takiej opcji.\n");
32              break;
33      }
34
35      return 0;
36  }
```

Instrukcja switch

Start here | switch.c

```
17      switch(choice){
18          case(1):
19              printf("Wybrales opcje pierwsza.\n");
20              break;
21  Mozesz wybrac jedna opcje.
22  1. Pierwsza.
23  2. Druga.
24  3. Trzecia.
25  4. Czwarta.
26  Wybierz opcje:
27  3
28  Wybrales opcje trzecia.
29
30
31
32              break;
33      }
34
35      return 0;
36  }
```

Politechnika Wrocławska

Instrukcja switch

Start here | switch.c

```
17      switch(choice){
18          case(1):
19              printf("Wybrales opcje pierwsza.\n");
20              break;
21  Mozesz wybrac jedna opcje.
22  1. Pierwsza.
23  2. Druga.
24  3. Trzecia.
25  4. Czwarta.
26  Wybierz opcje:
27  5
28  Nie ma takiej opcji.
29
30
31
32              break;
33      }
34
35      return 0;
36  }
```

Politechnika Wrocławska

Instrukcja switch

Start here | switch_bez_break.c

```
17      switch(choice){
18          case(1):
19              printf("Wybrales opcje pierwsza.\n");
20          case(2):
21              printf("Wybrales opcje druga.\n");
22          case(3):
23              printf("Wybrales opcje trzecia.\n");
24          case(4):
25              printf("Wybrales opcje czwarta.\n");
26          default:
27              printf("Nie ma takiej opcji.\n");
28      }
29
30      return 0;
31  }
```

Politechnika Wrocławska

Instrukcja switch

Start here | switch_bez_break.c

```
17    switch(choice){
18  Mozesz wybrac jedna opcje.
19  1. Pierwsza.
20  2. Druga.
21  3. Trzecia.
22  4. Czwarta.
23  Wybierz opcje:
24  2
25  Wybrales opcje druga.
26  Wybrales opcje trzecia.
27  Wybrales opcje czwarta.
28  Nie ma takiej opcji.
29
30    return 0;
31  }
```

Politechnika Wrocławska

Podsumowanie

- Arytmetyka komputerowa
- Typy danych w języku C

- Algorytmy iteracyjne
- Instrukcje sterujące języka C