Politechnika Wrocławska

Wstęp do programowania

INP001213Wcl

rok akademicki 2018/19

semestr zimowy

Wykład 7

Karol Tarnowski

karol.tarnowski@pwr.edu.pl

A-1 p. 411B

Politechnika Wrocławska

Plan prezentacji

- Algorytmy rekurencyjne
  - schemat Hornera
  - algorytm Eukildesa
  - obliczanie potęgi
  - wieże Hanoi
- Iteracja a rekurencja
  - ciąg Fibonacciego
  - symbol Newtona
  - Silnia
- Rekurencja ogonowa

Na podstawie:

- M. M. Sysło, *Algorytmy*

Politechnika Wrocławska

Rekurencja

Schemat Hornera

$$w_n(x) = a_0x^n + a_1x^{n-1} + \ldots + a_{n-1}x^1 + a_n$$

$$w_n(x) = (a_0x^{n-1} + a_1x^{n-2} + \ldots + a_{n-1})x + a_n$$

$$w_n(x) = w_{n-1}(x)x + a_n$$

$$w_{n-1}(x) = a_0x^{n-1} + a_1x^{n-2} + \ldots + a_{n-1}$$

Rekurencja

Schemat Hornera

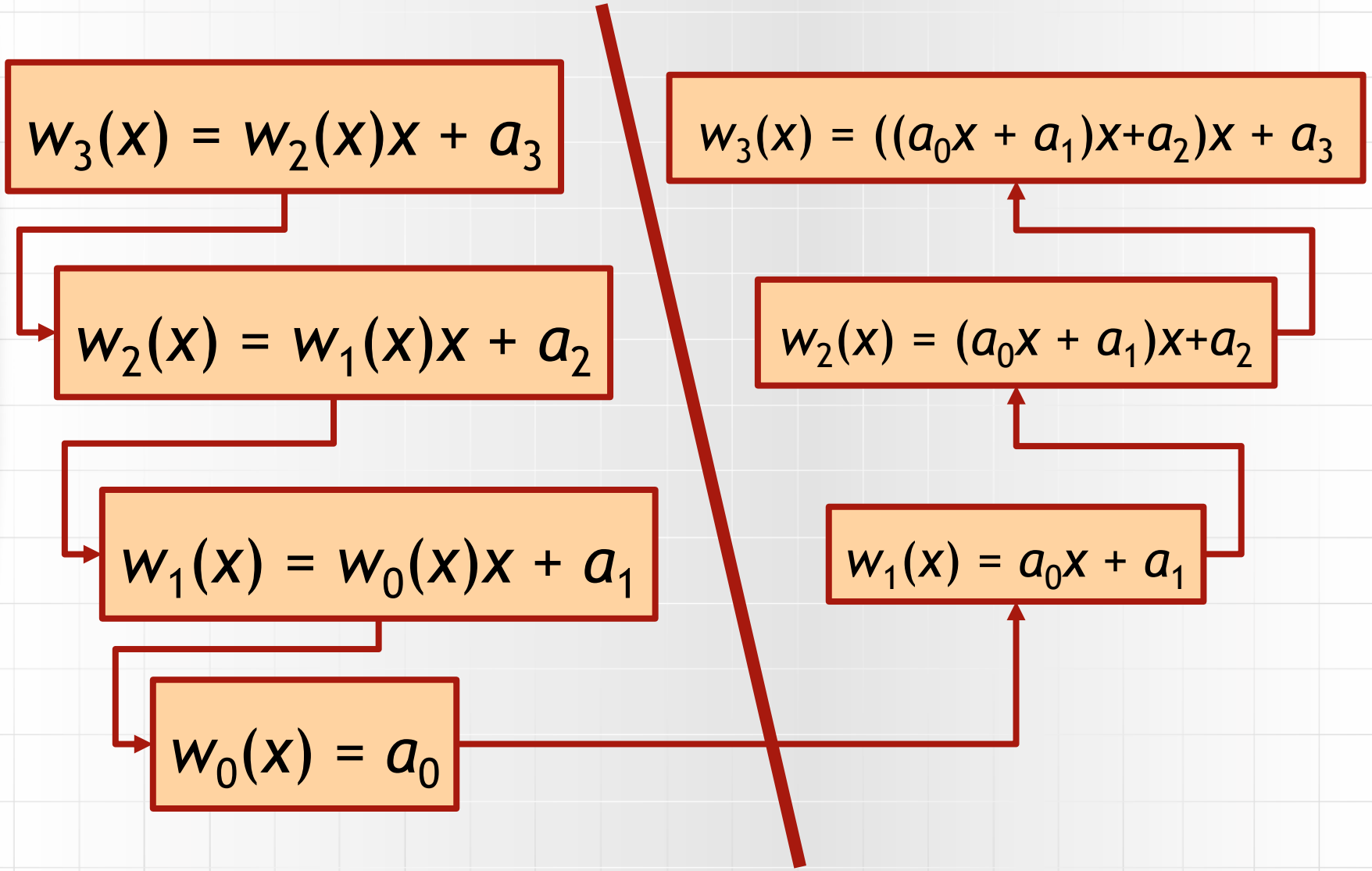
$$w_n(x) = w_{n\text{-}1}(x)x + a_n$$

$$w_0(x) = a_0$$

$$w_n(x) = \begin{cases} a_0, & \text{dla } n = 0, \\ w_{n-1}(x)x + a_n, & \text{dla } n \geq 1. \end{cases}$$

Rekurencja

Schemat Hornera



Politechnika Wrocławska

Rekurencja

- wywołania rekurencyjne
- obliczenia właściwe

- poziom rekurencji

Politechnika Wrocławska

Algorytm Euklidesa

zakładając, że $m \leq n$

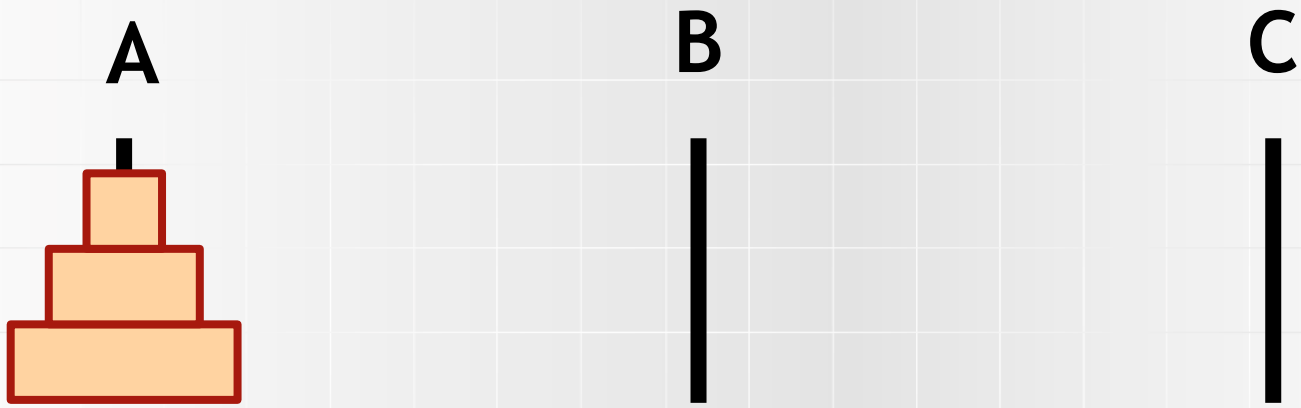
$$\mathrm{NWD}(m,n) = \begin{cases} n, & \text{dla } m = 0, \\ \mathrm{NWD}(n \bmod m, m), & \text{dla } m > 0. \end{cases}$$

Politechnika Wrocławska

Obliczanie potęgi

$$x^m = \begin{cases} x, & \text{dla } m = 1 \\ \left(x^{m/2}\right)^2, & m - \text{parzyste} \\ \left(x^{(m-1)/2}\right)^2 x, & m - \text{nieparzyste} \end{cases}$$

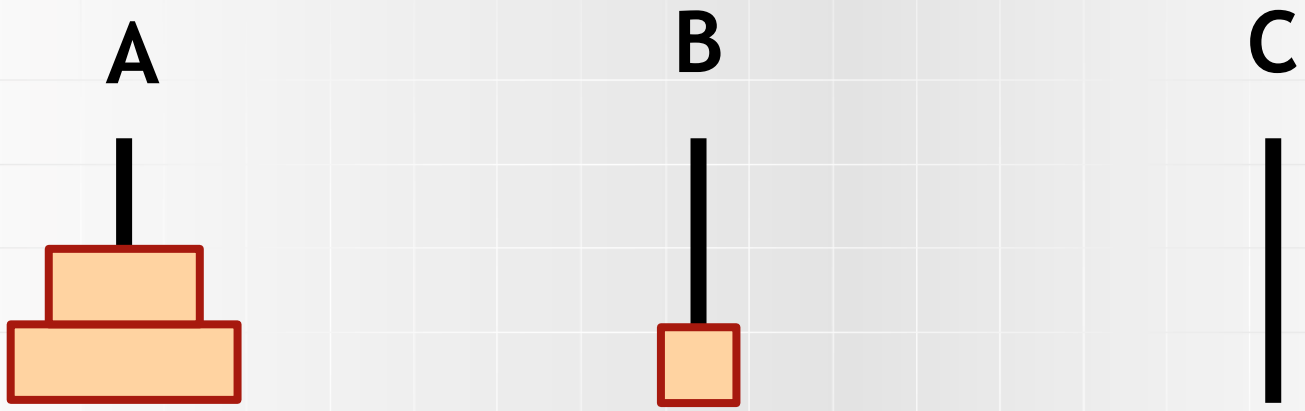
Wieże Hanoi



Przenieś wszystkie krążki z A na B (można użyć pomocniczo C), przy czym

- krążki wolno przenosić pojedynczo,
- nie można kłaść krążków większych na mniejszych.

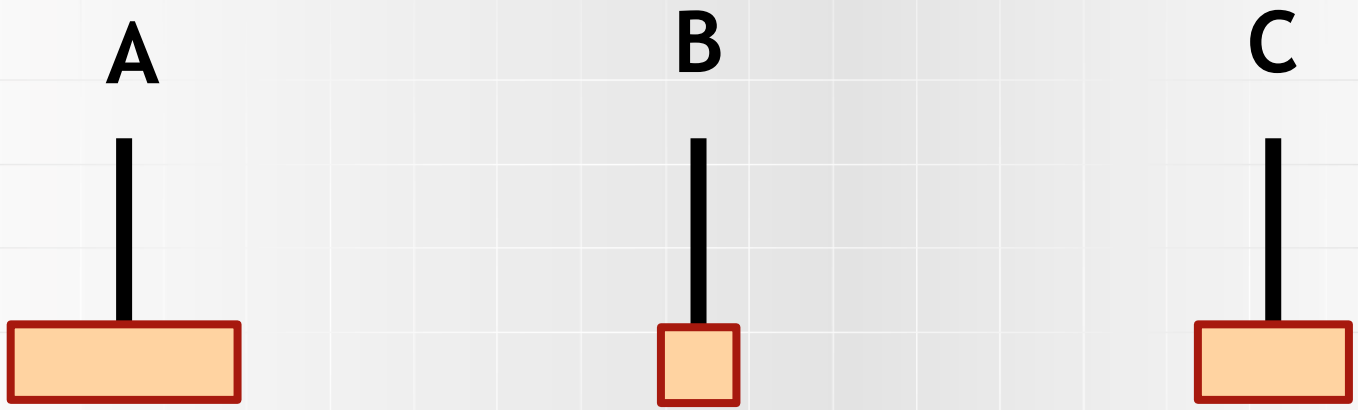
Wieże Hanoi



Przenieś wszystkie krążki z A na B (można użyć pomocniczo C), przy czym

- krążki wolno przenosić pojedynczo,
- nie można kłaść krążków większych na mniejszych.

Wieże Hanoi

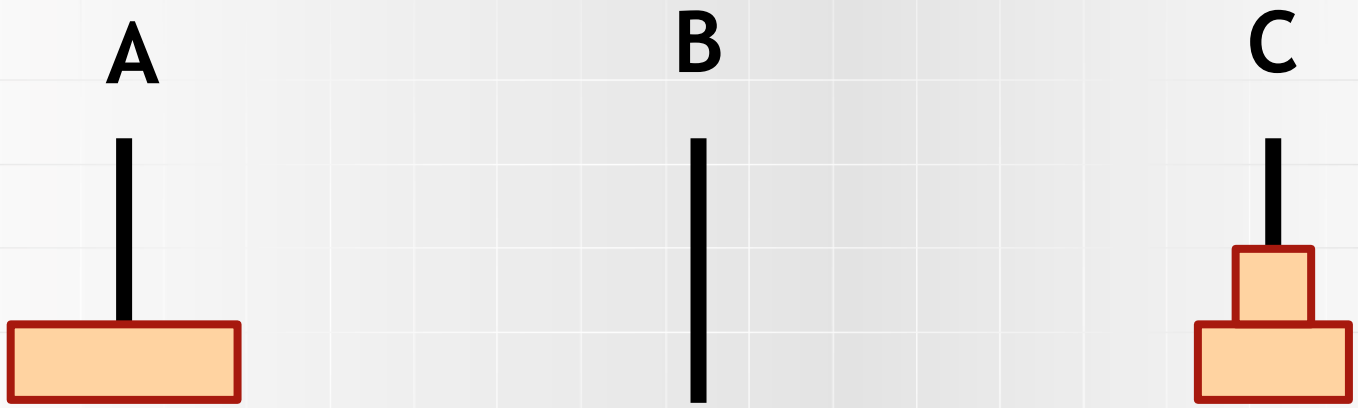


Przenieś wszystkie krążki z A na B (można użyć pomocniczo C), przy czym

- krążki wolno przenosić pojedynczo,
- nie można kłaść krążków większych na mniejszych.

Politechnika Wrocławska

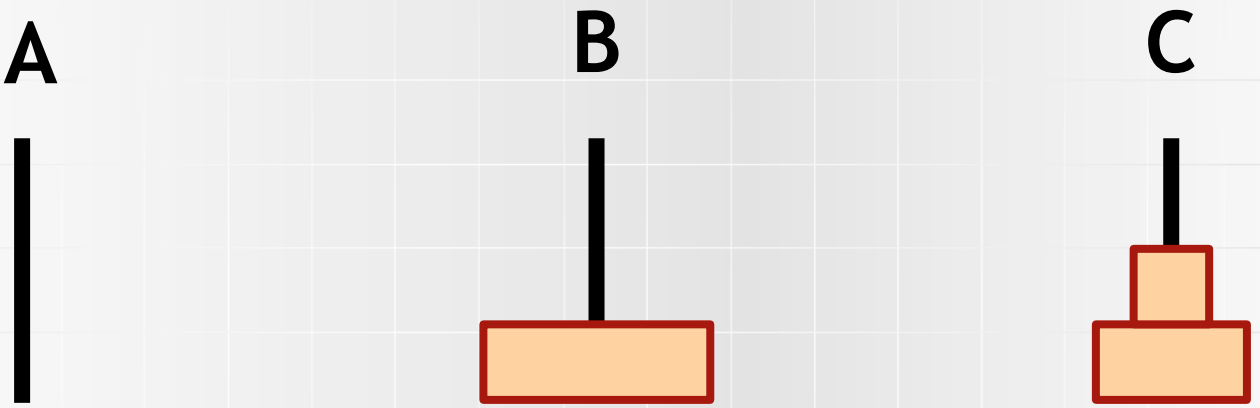
Wieże Hanoi



Przenieś wszystkie krążki z A na B (można użyć pomocniczo C), przy czym

- krążki wolno przenosić pojedynczo,
- nie można kłaść krążków większych na mniejszych.

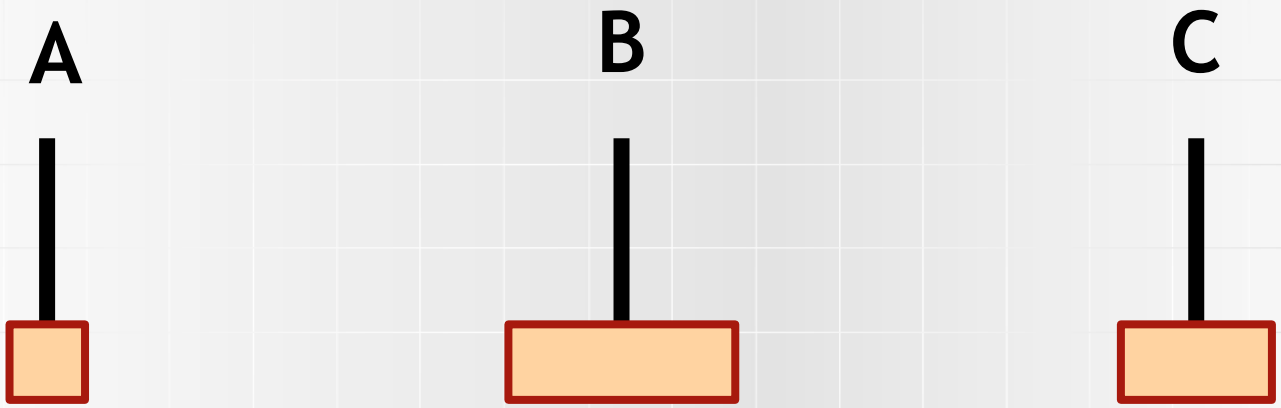
Wieże Hanoi



Przenieś wszystkie krążki z A na B (można użyć pomocniczo C), przy czym

- krążki wolno przenosić pojedynczo,
- nie można kłaść krążków większych na mniejszych.

Wieże Hanoi

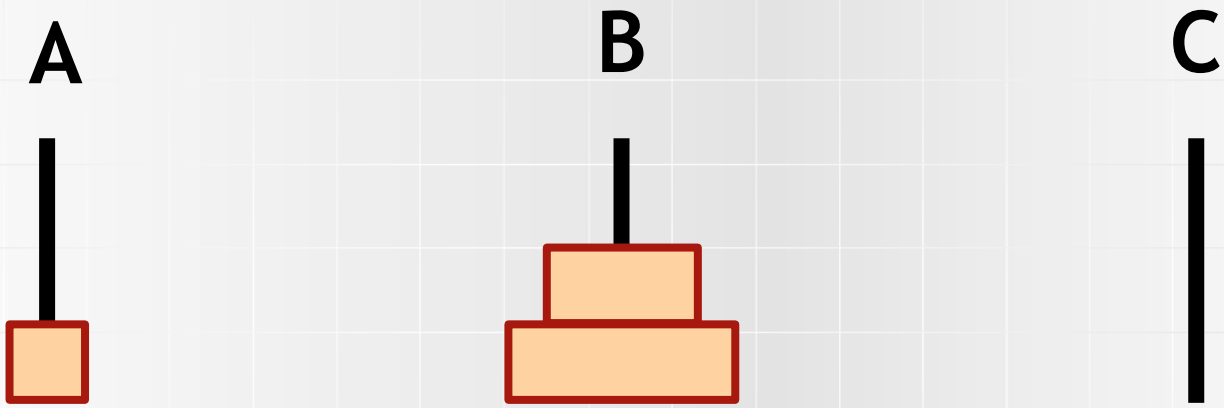


Przenieś wszystkie krążki z A na B (można użyć pomocniczo C), przy czym

- krążki wolno przenosić pojedynczo,
- nie można kłaść krążków większych na mniejszych.

Politechnika Wrocławska

Wieże Hanoi

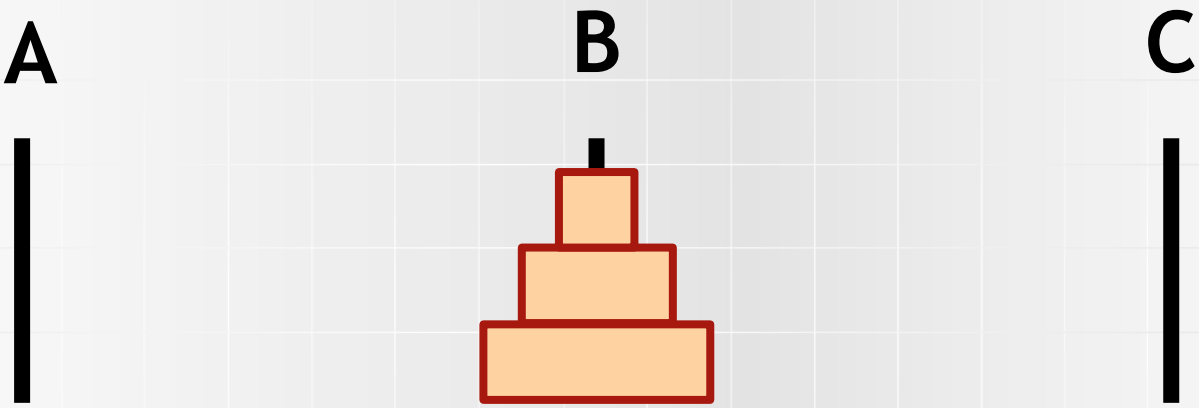


Przenieś wszystkie krążki z A na B (można użyć pomocniczo C), przy czym

- krążki wolno przenosić pojedynczo,
- nie można kłaść krążków większych na mniejszych.

Politechnika Wrocławska

Wieże Hanoi

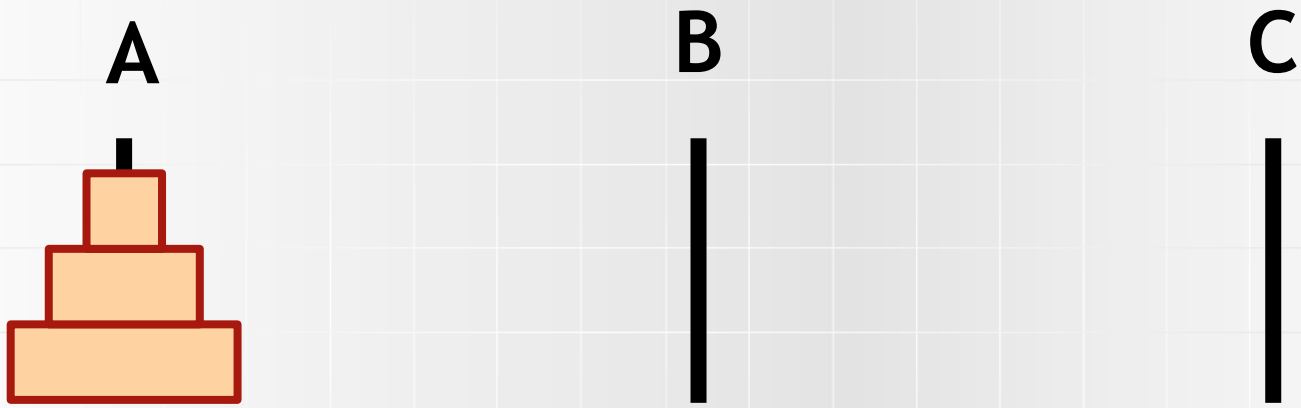


Przenieś wszystkie krążki z A na B (można użyć pomocniczo C), przy czym

- krążki wolno przenosić pojedynczo,
- nie można kłaść krążków większych na mniejszych.

Wieże Hanoi

Rozwiązanie iteracyjne

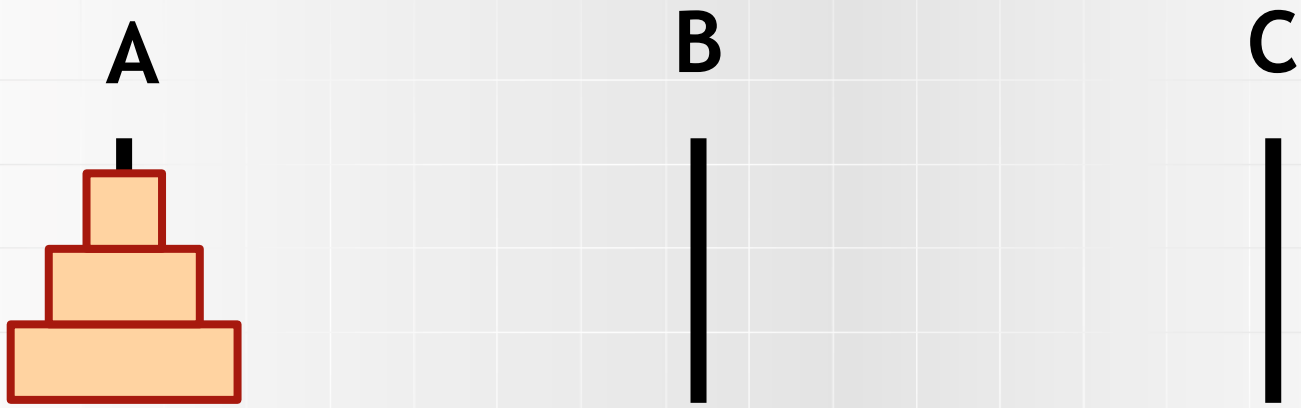


Spostrzeżenia:

- najmniejszy krążek zawsze jest na górze,
- na dwóch pozostałych palikach są krążki różnych rozmiarów (tylko jeden można przenieść).

Wieże Hanoi

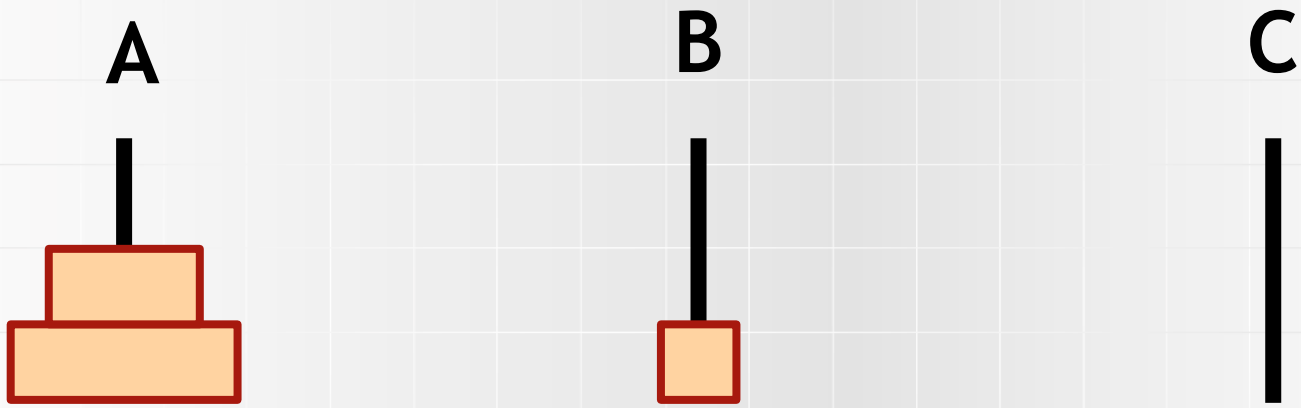
Rozwiązanie iteracyjne



1. Przenieś najmniejszy krążek na następny palik. Jeśli wszystkie krążki są przeniesione to zakończy.
2. Wykonaj jedyne możliwe przeniesienie krążka, który nie jest najmniejszym krążkiem, i idź do kroku 1.

Wieże Hanoi

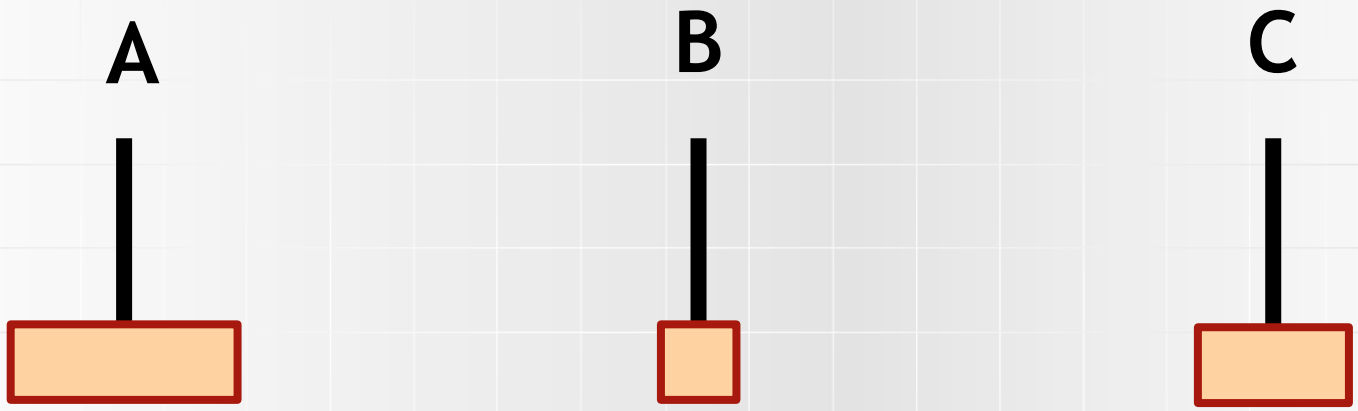
Rozwiązanie iteracyjne



1. Przenieś najmniejszy krążek na następny palik. Jeśli wszystkie krążki są przeniesione to zakończy.
2. Wykonaj jedyne możliwe przeniesienie krążka, który nie jest najmniejszym krążkiem, i idź do kroku 1.

Wieże Hanoi

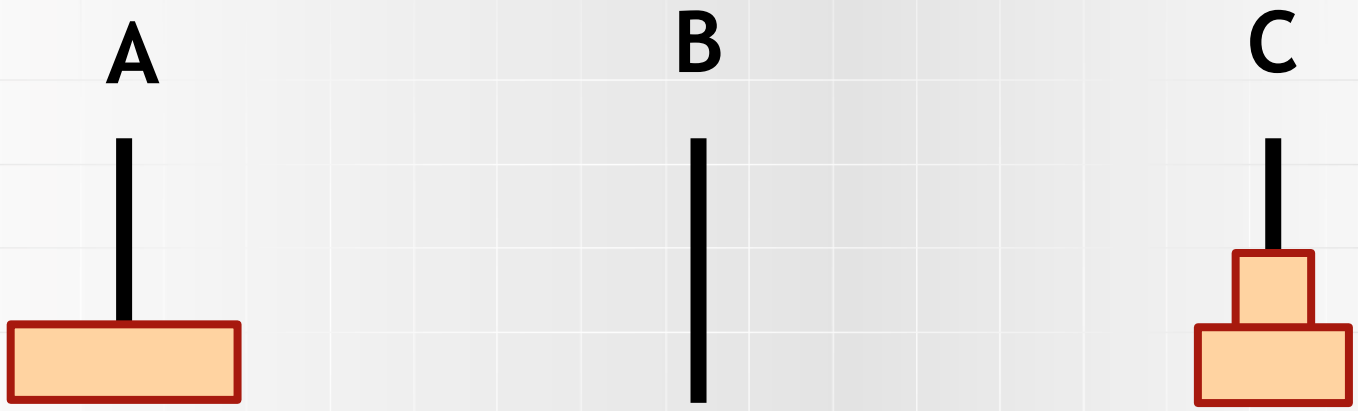
Rozwiązanie iteracyjne



1. Przenieś najmniejszy krążek na następny palik. Jeśli wszystkie krążki są przeniesione to zakończy.
2. Wykonaj jedyne możliwe przeniesienie krążka, który nie jest najmniejszym krążkiem, i idź do kroku 1.

Wieże Hanoi

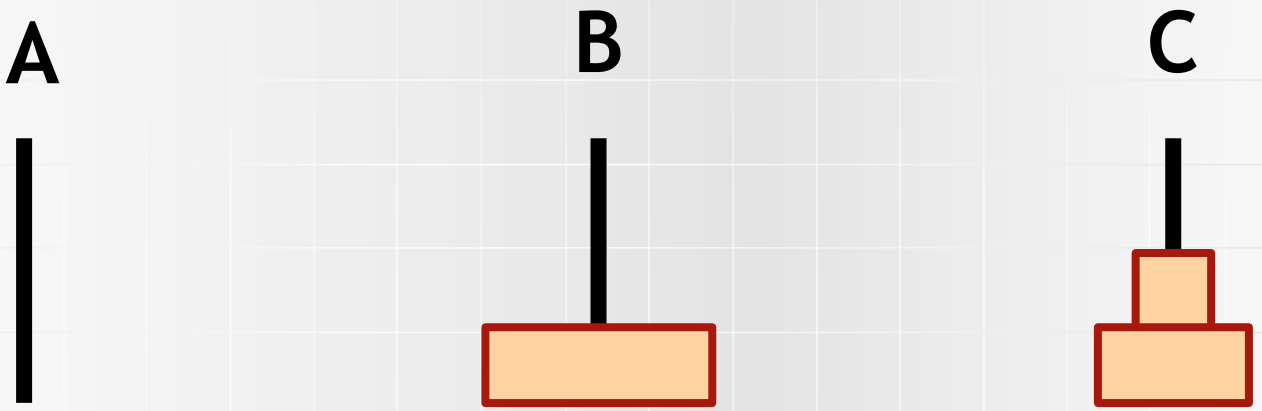
Rozwiązanie iteracyjne



1. Przenieś najmniejszy krążek na następny palik. Jeśli wszystkie krążki są przeniesione to zakończy.
2. Wykonaj jedyne możliwe przeniesienie krążka, który nie jest najmniejszym krążkiem, i idź do kroku 1.

Wieże Hanoi

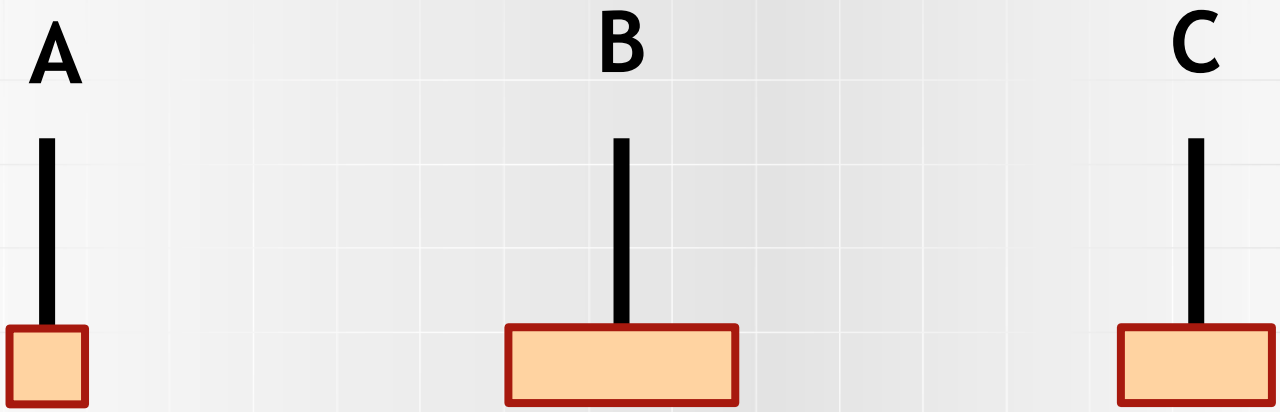
Rozwiązanie iteracyjne



1. Przenieś najmniejszy krążek na następny palik. Jeśli wszystkie krążki są przeniesione to zakończy.
2. Wykonaj jedyne możliwe przeniesienie krążka, który nie jest najmniejszym krążkiem, i idź do kroku 1.

Wieże Hanoi

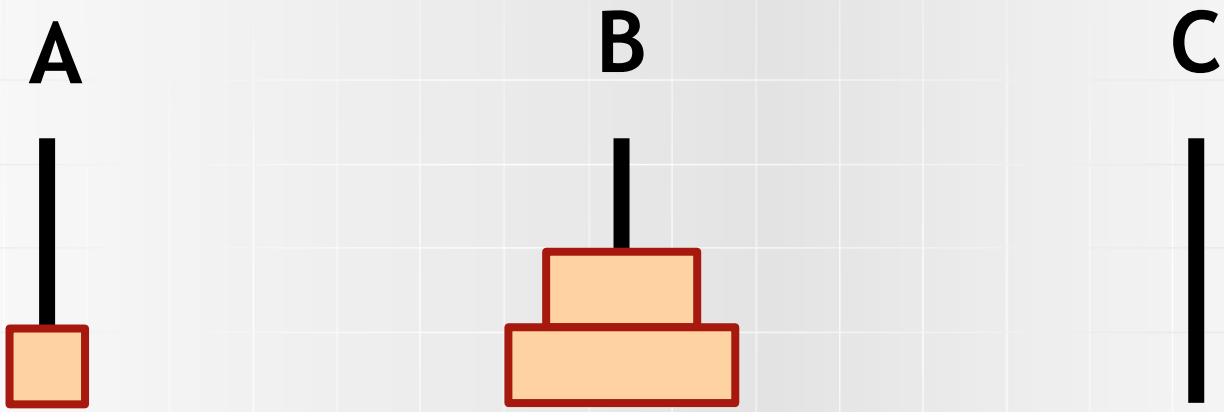
Rozwiązanie iteracyjne



1. Przenieś najmniejszy krążek na następny palik. Jeśli wszystkie krążki są przeniesione to zakończy.
2. Wykonaj jedyne możliwe przeniesienie krążka, który nie jest najmniejszym krążkiem, i idź do kroku 1.

Wieże Hanoi

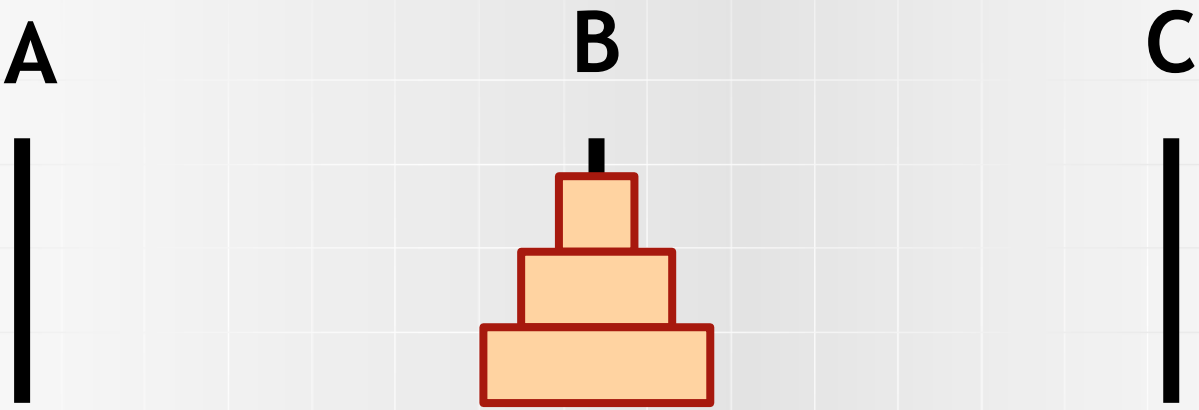
Rozwiązanie iteracyjne



1. Przenieś najmniejszy krążek na następny palik. Jeśli wszystkie krążki są przeniesione to zakończy.
2. Wykonaj jedyne możliwe przeniesienie krążka, który nie jest najmniejszym krążkiem, i idź do kroku 1.

Wieże Hanoi

Rozwiązanie iteracyjne



1. Przenieś najmniejszy krążek na następny palik. Jeśli wszystkie krążki są przeniesione to zakończy.
2. Wykonaj jedyne możliwe przeniesienie krążka, który nie jest najmniejszym krążkiem, i idź do kroku 1.

Wieże Hanoi

Liczba przeniesień krążków

$$h(n) = \begin{cases} 1, & \text{dla } n = 1, \\ 2h(n-1)+1, & \text{dla } n > 1. \end{cases}$$

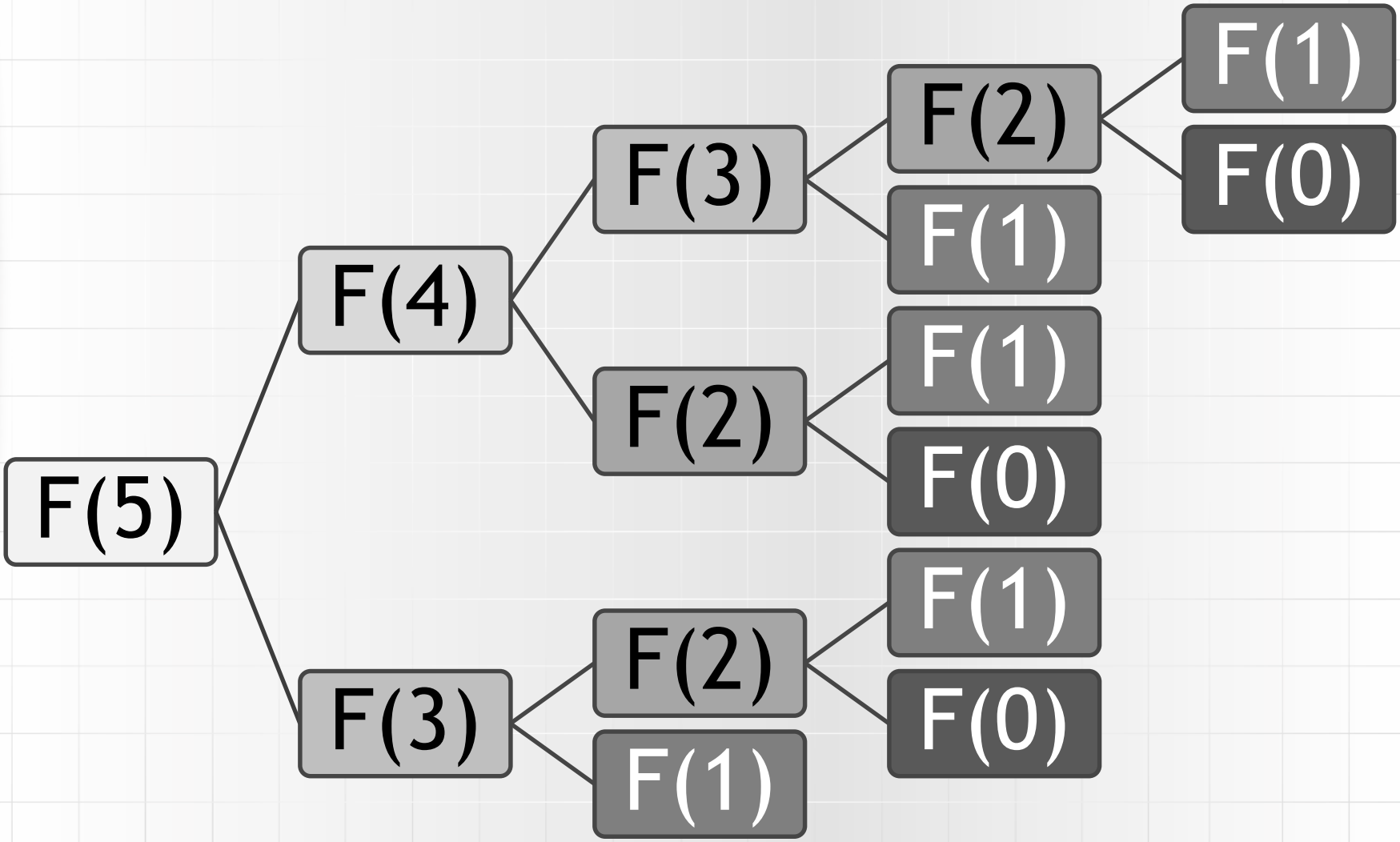
Politechnika Wrocławska

Ciąg Fibonacciego

$$F(n) = \begin{cases} 0 & \text{dla } n = 0, \\ 1 & \text{dla } n = 1, \\ F(n-1) + F(n-2) & \text{dla } n > 1. \end{cases}$$

Ciąg Fibonacciego

Wywołania rekurencyjne



Politechnika Wrocławska

Symbol Newtona

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \quad \text{dla } 0 \le k \le n$$

$$\binom{n}{k} = \begin{cases} 1 & \text{dla } k = 0 \text{ lub } k = n \\ \binom{n-1}{k-1} + \binom{n-1}{k} & \text{dla } 0 < k < n \end{cases}$$

$$\binom{n}{k} = \begin{cases} 1 & \text{dla } k = 0 \text{ lub } k = n \\ \binom{n-1}{k-1} \cdot \frac{n}{k} & \text{dla } 0 < k < n \end{cases}$$

Politechnika Wrocławska

Silnia

$$n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$$

$$n! = \begin{cases} 1 & \text{dla } n = 0, \\ n \cdot (n-1)! & \text{dla } n > 1. \end{cases}$$

Politechnika Wrocławska

Silnia

Start here | rekurencja.c

```
1  /*rekurencja.c*/
2  /*Program demonstruje działanie rekurencji
3   na przykładzie funkcji obliczającej
4   silnię. Pokazano także wykorzystanie
5   rekurencji ogonowej.*/
6
7  #include <stdio.h>
8
9  int silnia_rek(int n);
10 int silnia_ogon(int n);
11 int silnia_pom(int n, int a);
12
13 int main(){
14     int n = 5;
15     char format[] = "%d! = %d\n";
16     printf(format,n,silnia_rek(n));
17     printf(format,n,silnia_ogon(n));
18     return 0;
19 }
```

Politechnika Wrocławska

Silnia

Start here | rekurencja.c

```
1   /*rekurencja.c*/
2   /*Program demonstruje działanie rekurencji
6
7   #include <stdio.h>
8
9   int silnia_rek(int n);
10  int silnia_ogon(int n);
11  int silnia_pom(int n, int a);
12
13  int main(){
20
21  int silnia_rek(int n){
22      if(n==0)
23          return 1;
24      else
25          return n*silnia_rek(n-1);
26  }
```

Rekurencja ogonowa

Silnia

Start here | rekurencja.c

```
1   /*rekurencja.c*/
2   /*Program demonstruje działanie rekurencji
6
7   #include <stdio.h>
8
9   int silnia_rek(int n);
10  int silnia_ogon(int n);
11  int silnia_pom(int n, int a);
12
13  int main(){
20  int silnia_rek(int n){
26
27  int silnia_ogon(int n){
28      return silnia_pom(n,1);
29  }
30
31  int silnia_pom(int n, int a){
32      if(n==0)
33          return a;
34      else
35          return silnia_pom(n-1,a*n);
36  }
```

Rekurencja ogonowa

Silnia

```
silnia_ogon(5)
silnia_pom(5,1)
silnia_pom(4,5)
silnia_pom(3,20)
silnia_pom(2,60)
silnia_pom(1,120)
silnia_pom(0,120)
120
```

Politechnika Wrocławska

Podsumowanie

- Aby zrozumieć rekurencję, trzeba zrozumieć rekurencję ☺

- Obliczenia rekurencyjne składają się z dwóch faz:
  - wywołania rekurencyjne
  - powrót z wywołań obejmujący wykonywanie właściwych obliczeń (lub tylko przenoszenie wyników)
- Pamiętaj o warunku stopu
- Wywołania rekurencyjne mogą prowadzić do nadmiarowych obliczeń