

## Wykład 1

- Informacje o przedmiocie
- Zakres przedmiotu
- Literatura
- Pojęcia podstawowe

7.10.2024

## Wstęp do informatyki

Rok akademicki: 2024/2025

Liczba godzin: Semestr 1, wyk. 28 godz., ćw. 28 godz.

Wykład: Marek Gajęcki

Ćwiczenia: Marek Gajęcki, Andrei Karatkevich, Łukasz Janeczko

Cel wykładu: Celem przedmiotu jest zapoznanie studentów z podstawowymi pojęciami informatyki, programowaniem w języku proceduralnym oraz wprowadzenie do podstawowych algorytmów i struktur danych.

Slajdy i inne materiały: <https://upel.agh.edu.pl>

### Wyznaczanie oceny końcowej:

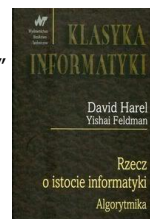
1. W semestrze odbędą się 3 kolokwia i na ich podstawie zostanie wystawiona ocena z ćwiczeń.
2. Warunkiem otrzymania oceny końcowej jest otrzymanie pozytywnych ocen z ćwiczeń i egzaminu.
3. Ocenę końcową obliczamy ze wzoru  $OK = \max(\text{round}(SR), 3.0)$  gdzie SR jest średnią arytmetyczną z ocen zaliczenia ćwiczeń i egzaminów uzyskanych we wszystkich terminach.
4. Jeżeli pozytywną ocenę z ćwiczeń i egzaminu uzyskano w pierwszym terminie oraz ocena końcowa jest mniejsza niż 5.0 to ocena końcowa jest podnoszona o 0.5

## Zakres przedmiotu

- Pojęcia podstawowe
- Mechanizmy języka strukturalnego
- Skalarne typy danych w językach programowania
- Strukturalne typy danych w językach programowania
- Procedury i funkcje - przekazywanie parametrów
- Rekurencja, przykłady stosowania
- Struktury odsyłaczowe
- Przykłady struktur danych
- Pojęcie złożoności obliczeniowej
- Przykłady algorytmów
- Reprezentacja liczb w komputerze
- Pojęcia architektury komputera, paradygmatów programowania

## Literatura

- D. Harel „Rzecz o istocie informatyki - algorytmika”
- J.G. Brookshear „Informatyka w ogólnym zarysie”
- J. Mieścicki „Wstęp do informatyki nie tylko dla informatyków”
- N. Wirth „Algorytmy + struktury danych = programy”
- J. Bentley „Perełki oprogramowania”
- J. Bentley „Więcej perełek oprogramowania”
- T.H. Cormen „Wprowadzenie do algorytmów”
- Mark Lutz „Python. Wprowadzenie”



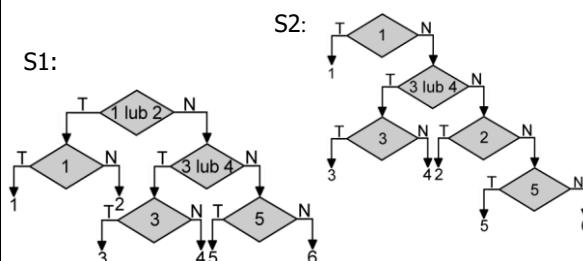
## Informatyka (Computer Science)

**Informatyka** to nauka o przetwarzaniu informacji.

Aktualnie obejmuje wiele zagadnień, między innymi:

- algorytmika, struktury danych, języki programowania
- mikroprocesory, architektury komputerów
- bazy danych, systemy operacyjne, sieci komputerowe
- kompilatory, kryptografia, metody numeryczne
- inżynieria oprogramowania, projektowanie systemów
- grafika, sztuczna inteligencja, aplikacje internetowe
- złożoność obliczeniowa
- ...

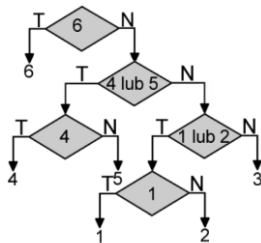
## Identyfikacja elementów zbioru



$$E(S1) = 1/6 \cdot 2 + 1/6 \cdot 2 + 1/6 \cdot 3 + 1/6 \cdot 3 + 1/6 \cdot 3 + 1/6 \cdot 3 = 2.66$$
$$E(S2) = 1/6 \cdot 1 + 1/6 \cdot 3 + 1/6 \cdot 3 + 1/6 \cdot 3 + 1/6 \cdot 4 + 1/6 \cdot 4 = 3$$

## Identyfikacja elementów zbioru

S3:



$$P(6)=0.95$$

$$P(1-5)=0.01$$

$$E(S1)=0.01 \cdot 2 + 0.01 \cdot 2 + 0.01 \cdot 3 + 0.01 \cdot 3 + 0.01 \cdot 3 + 0.95 \cdot 3 = 2.98$$

$$E(S3)=0.01 \cdot 4 + 0.01 \cdot 4 + 0.01 \cdot 3 + 0.01 \cdot 3 + 0.01 \cdot 3 + 0.95 \cdot 1 = 1.12$$

## Teoria informacji

Ilość informacji zawarta w danym zbiorze jest miarą stopnia trudności rozpoznania elementów tego zbioru.



Claude Shannon

Ilością informacji zawartej w zbiorze  $X=\{x_1, x_2, \dots, x_N\}$ , nazywamy liczbę:

$$H(X) = - \sum_{i=1}^N p_i \cdot \log_2(p_i)$$

gdzie:

$p_i (p_i > 0, \sum p_i = 1)$  jest prawdopodobieństwem wystąpienia elementu  $x_i$

**Przykłady:**

$\{0,1\}$   $-\log_2 \frac{1}{2} = 1$  1 bit  
 $\{0,9\}$   $-\log_2 0.9 \approx 0.152$  3.32 bita  
 zbiór liter języka ang. 4.7 bita

$$\left\{ \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4} \right\} \Rightarrow -\left( \frac{1}{4} \log_2 \frac{1}{4} + \frac{1}{4} \log_2 \frac{1}{4} + \frac{1}{4} \log_2 \frac{1}{4} + \frac{1}{4} \log_2 \frac{1}{4} \right) = 2$$

## Kompresja danych

Ciąg danych

AABACADABA

1) Kodowanie proste

A - 00  
 B - 01  
 C - 10  
 D - 11

2) Kodowanie Huffmana

A - 0 0.6  
 B - 10 0.2  
 C - 110 0.1  
 D - 111 0.1

Po zakodowaniu

1) 00 00 01 00 10 00 11 00 01 00 20 bitów  
 2) 0 0 10 0 110 0 111 0 10 0 16 bitów

Ilość informacji

$$H(X) = 0.6 \cdot \log(0.6) + 0.2 \cdot \log(0.2) + 0.1 \cdot \log(0.1) + 0.1 \cdot \log(0.1) = 15.7$$

## Podstawowe pojęcia

**Zadanie algorytmiczne** – polega na określeniu:

- wszystkich poprawnych danych wejściowych
- oczekiwanych wyników jako funkcji danych wejściowych

**Algorytm** - specyfikacja ciągu elementarnych operacji, które przekształcają dane wejściowe na wyniki.

Algorytm można przedstawić w postaci:

- werbalnej (*opis słowny*)
- symbolicznej (*schemat blokowy*)
- programu

## Przykład zapisu algorytmu

**Problem:** równanie kwadratowe

**Dane:** współczynniki **a, b, c**

**Wyjście:** pierwiastki **x1, x2** albo informacja o ich braku

**Postać werbalna** algorytmu:

Mając dane współczynniki **a, b, c**:

oblicz  $d = b^2 - 4 \cdot a \cdot c$

Jeżeli **d** jest nieujemne:

oblicz  $p = \sqrt{d}$

oblicz  $x1 = (-b-p)/(2 \cdot a)$

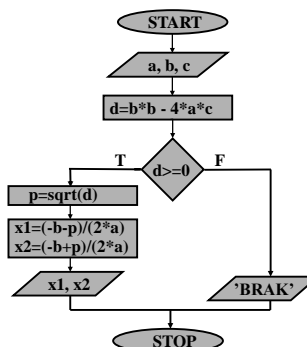
oblicz  $x2 = (-b+p)/(2 \cdot a)$

wypisz wartości **x1, x2**

Jeżeli **d** jest ujemne:

wypisz "BRAK PIERWIASTKÓW"

## Zapis symboliczny a program



```
from math import sqrt

a = float(input("a="))
b = float(input("b="))
c = float(input("c="))
d = b*b - 4*a*c
if d >= 0:
    p = sqrt(d)
    x1 = (-b-p)/(2*a)
    x2 = (-b+p)/(2*a)
    print("x1=", x1)
    print("x2=", x2)
else:
    print("BRAK")
```

## Algorytm Euklidesa

Algorytm Euklidesa (około 300 r. p.n.e.)  
obliczający największy wspólny dzielnik.

**Dane:** liczby naturalne a,b

**Wynik:** NWD(a,b)



```
a = int(input("a="))
b = int(input("b="))
while a!=b:
    if a>b:
        a = a-b
    else:
        b = b-a
print(a)
```

a	b
24	30
24	6
18	6
12	6
6	6

## Algorytm Euklidesa

```
def nwd1(a,b):
    while b!=0:
        pom = b
        b = a%b
        a = pom
    return a
```

```
def nwd3(a,b):
    while True:
        if a>b: a = a%b
        if a==0: return b
        if b>a: b = b%a
        if b==0: return a
```

```
def nwd5(a,b):
    if b!=0:
        return nwd5(b,a%b)
    return a
```

```
def nwd2(a,b):
    while b!=0:
        (a,b)=(b,a%b)
    return a
```

```
def nwd4(a,b):
    while a*b!=0:
        if a>b: a = a%b
        else: b = b%a
    return a+b
```

```
def nwd6(a,b):
    while a!=b:
        if a>b:
            a = a-b
        else:
            b = b-a
    return a
```

## Przykład zadania

**Problem:** Rozkład liczby na czynniki pierwsze

Proszę napisać program, który dla wczytanej  
liczby naturalnej wypisuje jej rozkład na  
czynniki pierwsze.

**Przykład:**

120 : 2, 2, 2, 3, 5

```
120 | 2
60  | 2
30  | 2
15  | 3
5   | 5
1   |
```

## Rozkład na czynniki pierwsze

**Rozwiązanie pierwsze**

```
n = int(input("n="))
b = 2
while n>1:
    if n % b == 0:
        print(b)
        n = n // b
    else:
        b = b+1
    # end if
# end while
print("koniec")
```

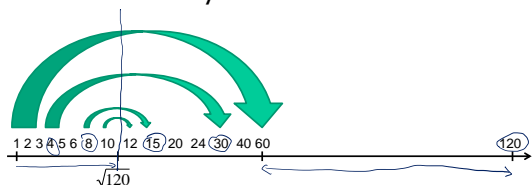
13 : 5 = 2  $\times$  3

13 // 5 = 2

13 / 5 = 2,6

## Położenie podzielników liczby

Podzielniki liczby 120



## Rozkład na czynniki pierwsze

**Rozwiązanie drugie**

```
from math import sqrt
n = int(input("n="))
b = 2
while b<=sqrt(n):
    if n % b == 0:
        print(b)
        n = n // b
    else:
        b = b+1
    # end if
# end while
if n>1: print(n)
print("koniec")
```

120 : 2 = 60  
60 : 2 = 30  
30 : 2 = 15  
15 : 3 = 5  
5 : 5 = 1

## Porównanie rozwiązań

Liczba cyfr Liczby pierwszej	Algorytm pierwszy	Algorytm drugi
6	0.07s	0.0s
7	0.58s	0.0s
8	7.75s	0.0s
9	1m7.2s	0.01s
10	9m43s	0.01s
11	1h23m	0.04s
12	12h27m	0.13s
13	4d9h	0.42s
14	37d12h	1.23s

## RSA

$$P \stackrel{?}{\neq} NP$$

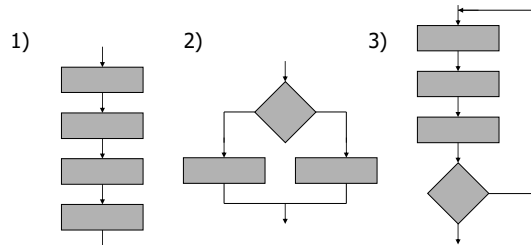
## Czy można jeszcze szybciej?

14 1,235 : 3600  
: 24  
: 365,25

50 1,235 · 10<sup>18</sup>

## Budowa algorytmów

- 1) Bezpośrednie następstwo
- 2) Wybór warunkowy
- 3) Iteracja warunkowa



## Podstawowe pojęcia

### Aspekty języka programowania:

- **Syntaktyka** (składnia) - zbiór reguł określający formalnie poprawne konstrukcje językowe
- **Semantyka** - opisuje znaczenie konstrukcji językowych, które są poprawne składniowo

## Notacja EBNF

### Elementy notacji EBNF:

- Symbole pomocnicze (nieterminalne)
- Symbole końcowe (terminalne)
- Produkcje
- Metasyμβole
  - < > - symbol pomocniczy
  - ::= - symbol produkcji
  - | - symbol alternatywy
  - { } - wystąpienie 0 lub 1 raz (EBNF)
  - [ ] - wystąpienie 0 lub więcej razy (EBNF)



John Backus      Peter Nau

EBNF de float

### Przykład

<cyfra dziesiętna> ::= 0|1|2|3|4|5|6|7|8|9  
 <liczba bez znaku> ::= <cyfra dziesiętna> {<cyfra dziesiętna>}  
 <liczba> ::= [+|-] <liczba bez znaku>

## Gramatyka języka

```

<pgm> ::= <pgmHeading> <pgmDeclarations> <codeBlock> ";"
<pgmHeading> ::= program <pgmIdentifier> ";"
<pgmDeclarations> ::= { <pgmDeclaration> }
<pgmDeclaration> ::= <varDeclaration> | <typeDeclaration> | <procDeclaration>
....
<codeBlock> ::= begin { <statement> } end
<statement> ::= <assignStatement> | <ifStatement> | <whileStatement> | <procCall>
<assignStatement> ::= <variable> "=" <expression> ";"
<ifStatement> ::= if <expression> then <codeBlock> [ else <codeBlock> ] ";"
<whileStatement> ::= while <expression> do <codeBlock> ";"
....
<identifier> ::= <letter> { <letter> | <digit> }
<longint> ::= <digit> { <digit> }
<relOperator> ::= "=" | "<" | ">" | "<=" | ">=" | ">="
<addOperator> ::= "+" | "-" | or
<multOperator> ::= "*" | div | and
<letter> ::= "A" | ... | "Z" | "a" | ... | "z"
<digit> ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"

```

## Pytania i zadania

- Ile informacji zawiera 10 znakowe słowo, którego każdy znak z jednakowym prawdopodobieństwem jest jedną z liter a, b, c ?
- Jak stworzyć kody Huffmana dla zbiorów 5,6,7,... elementów?
- Jak przyspieszyć działania programu rozkładu na czynniki pierwsze?
- Czym różni się notacja BNF od notacji EBNF?
- Zapisać w notacji EBNF składnię instrukcji warunkowej oraz pętli.
- Zapisać w notacji EBNF składnię wyrażenia arytmetycznego w którym mogą wystąpić zmienne a,b,c, operatory +, \* oraz nawiasy ( ).
- Proszę znaleźć najmniejszą liczbę pierwszą, której suma cyfr wynosi 101, a cyfrs są w porządku nierosnącym.