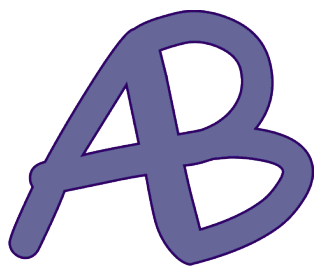


Wykład I

Cyfrowa reprezentacja informacji

Algorytmy – metody prezentacji i zapisu



Dlaczego system binarny?

Pojęcie bitu

Bit – jednostka informacji wystarczająca do zakomunikowania jednego z dwu równo prawdopodobnych zdarzeń.



1



0



Przyczyny zastosowania systemu binarnego

kb	Mb	Gb	Tb
kilobit	megabit	gigabit	terabit

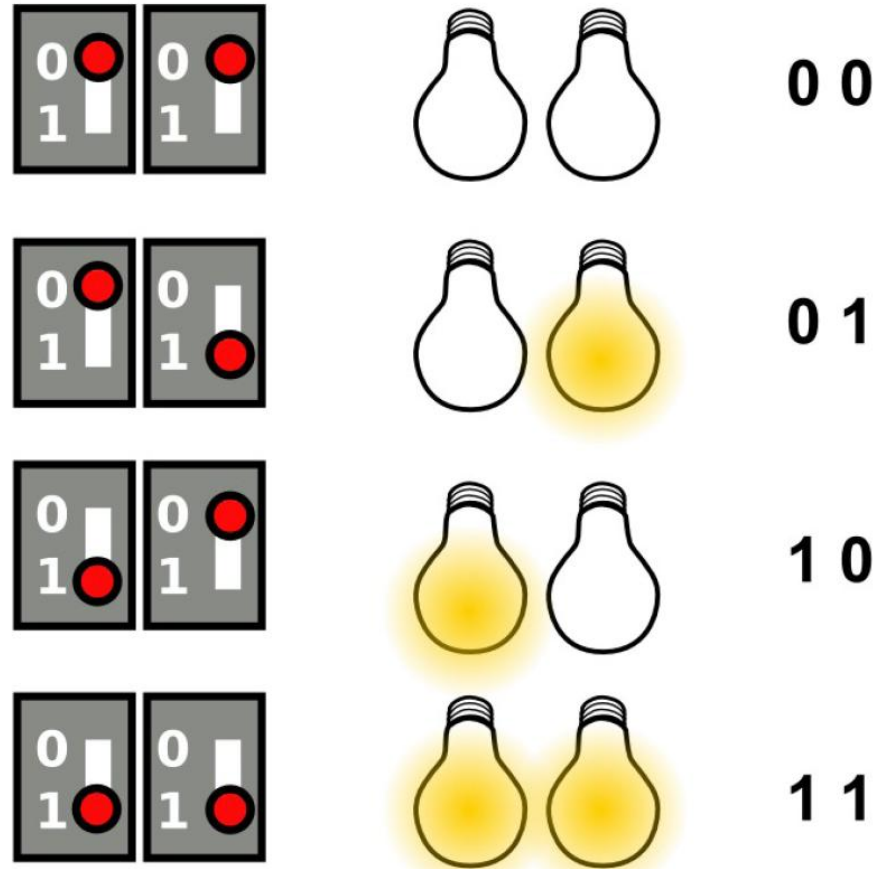
1 bajt = 8 bitów (*ang. byte*)

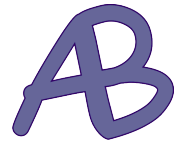
kB	MB	GB	TB
kilobajt	megabajt	gigabajt	terabajt

Przyczyny zastosowania systemu binarnego

Przyczyny zastosowania systemu binarnego w technologii cyfrowej to:

- łatwość implementacji elektrycznej i elektronicznej,
- odporność na zakłócenia,
- możliwość interpretacji cyfr {0, 1} jako wartości logicznych (algebra Boole'a).



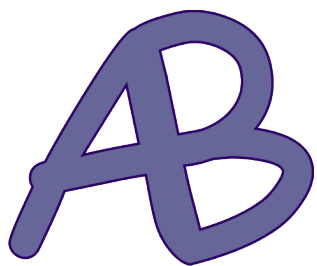


Przyczyny zastosowania systemu binarnego

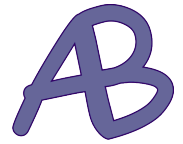
Ciekawostka:

Jedynym znanym komputerem zbudowanym z elementów 3-stanowych był eksperymentalny radziecki Sietuń (1959).

Element reprezentujący jednostkę informacji stanowiła para rdzeni magnetycznych, z których każdy mógł być namagnesowany w jednym z dwóch kierunków; czwarty -niewykorzystany stan -służył do celów kontrolnych.



Pozycyjne systemy liczbowe

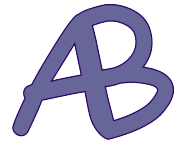


System dziesiętny

0 0 0
0 0 1
0 0 2
...
...
...
9 9 7
9 9 8
9 9 9

Ile różnych liczb można zapisać w systemie dziesiętnym za pomocą 3 cyfr?

System dziesiętny



0 0 0
0 0 1
0 0 2
...
...
...
9 9 7
9 9 8
9 9 9

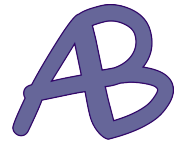
Tysiąc – od 0 do 999

W dowolnym systemie liczbowym można przedstawić

P^n

P - podstawa systemu
n - ilość cyfr

liczb/kombinacji.



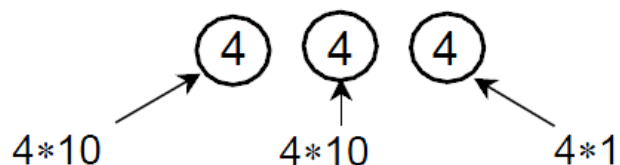
Przykładowa liczba 907 w systemie dziesiętnym powstaje wg poniższego schematu:

10^2 10^1 10^0

$$\mathbf{9 \quad 0 \quad 7} = \mathbf{9*100+0*10+7*1=907}_{10}$$

System o dowolnej podstawie

System pozycyjno–wagowy: na przykład liczba 444



Wagi systemu dziesiętnego: 1, 10, 100, 1000,

$$L = C_{n-1} \cdot P^{n-1} + C_{n-2} \cdot P^{n-2} + \dots + C_1 \cdot P^1 + C_0 \cdot P^0$$

C – elementy zbioru cyfr dostępnych w danym systemie,

$$C \in \{0, \dots, P-1\},$$

P – podstawa systemu, $P = 2, 4, 8, 10, 16$ (60 – Babilon, czas),

n – liczba całkowita.

System o dowolnej podstawie

Przykłady:

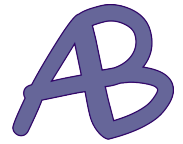
$$P = 2 \rightarrow C \in \{0,1\}$$

$$P = 4 \rightarrow C \in \{0,1,2,3\}$$

$$P = 8 \rightarrow C \in \{0,1,2,3,4,5,6,7\}$$

$$P = 10 \rightarrow C \in \{0,1,2,3,4,5,6,7,8,9\}$$

$$P = 16 \rightarrow C \in \underbrace{\{0,1,2,3,4,5,6,7,8,9\}}_{10 \text{ cyfr}} \underbrace{\{A,B,C,D,E,F\}}_{\text{uzupełnienie}}$$



System o dowolnej podstawie

ZAPIS liczby **1011** w różnych systemach ($n = 4$):

$$1011_{(2)} = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 8 + 0 + 2 + 1 = 11$$

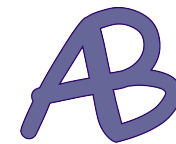
$$1011_{(4)} = 1 \cdot 4^3 + 0 \cdot 4^2 + 1 \cdot 4^1 + 1 \cdot 4^0 = 64 + 0 + 4 + 1 = 69$$

$$1011_{(8)} = 1 \cdot 8^3 + 0 \cdot 8^2 + 1 \cdot 8^1 + 1 \cdot 8^0 = 512 + 0 + 8 + 1 = 521$$

$$1011_{(10)} = 1 \cdot 10^3 + 0 \cdot 10^2 + 1 \cdot 10^1 + 1 \cdot 10^0 = 1000 + 0 + 10 + 1 = 1011$$

$$1011_{(16)} = 1 \cdot 16^3 + 0 \cdot 16^2 + 1 \cdot 16^1 + 1 \cdot 16^0 = 4096 + 0 + 16 + 1 = 4113$$

Systemy niepozycyjne



Zupełnie inna sytuacja występuje w zapisie liczby w systemie rzymskim.

Kolejne liczby od 1;:::; 9 mają postać:

I; II; III; IV; V; V I; V II; V III; IX

Widać, że w takim zapisie pozycja cyfry (o ile w ogóle można mówić w tym wypadku o cyfrze), nie jest związana z wyznaczaniem jej wartości, lecz istotna jest postać całej liczby.

*Taki system zapisu nazywamy **addytywnym systemem liczbowym**.*

System dwójkowy (binarny)

Korzystając z definicji pozycyjnego systemu liczbowego otrzymujemy, że podstawą systemu dwójkowego jest liczba 2, oraz cyframi tego systemu są elementy zbioru $\{0, 1\}$.

Zapiszmy przykładową liczbę w tym systemie

$$x = 10111010_{(2)}$$

otrzymujemy:

$$\begin{aligned} x = & 1 \cdot 2^9 + 0 \cdot 2^8 + 1 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + \\ & + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 \end{aligned}$$

Zastępując teraz potęgi liczby 2 odpowiednimi wartościami, otrzymujemy

$$\begin{aligned} x = & 1 \cdot 512 + 0 \cdot 256 + 1 \cdot 128 + 1 \cdot 64 + 1 \cdot 32 + \\ & + 1 \cdot 16 + 0 \cdot 8 + 1 \cdot 4 + 1 \cdot 2 + 0 \cdot 1 = 758_{(10)} \end{aligned}$$

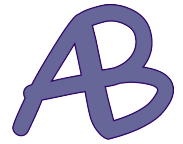
System dwójkowy (binarny)

Zapis binarny prosty pozwala za pomocą n cyfr zapisywać liczby z zakresu:

$$0 \leq L_{10} \leq 2^n - 1$$

$$\text{Dla } n = 8: \quad 0 \leq L_{10} \leq 2^8 - 1 = 256 - 1 = 255$$

$$\begin{aligned} 11111111_{(2)} &= 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = \\ &= 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255_{(10)} \end{aligned}$$



System dwójkowy (binarny)

ZALETY:

- prostota
- łatwa realizacja techniczna (elektronika)
- możliwość interpretacji cyfr $\{0, 1\}$ jako wartości logicznych (**algebra Boole'a**)

WADY:

- długość zapisu
- przyzwyczajenie

Ważniejsze potęgi dwójki

$$2^0 = 1$$

$$2^1 = 2$$

$$2^2 = 4$$

$$2^3 = 8$$

$$2^4 = 16$$

$$2^5 = 32$$

$$2^6 = 64$$

$$2^7 = 128$$

$$2^8 = 256 = 1 \text{ bajt}$$

$$2^{16} = 65.536$$

$$2^{24} = 16.777.216$$

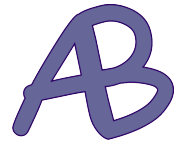
1 bajt = 8 bitów

10101111

$$2^{10} \text{ bajtów} = 1 \text{ kB} \quad (1024)$$

$$2^{20} \text{ bajtów} = 1 \text{ MB} \quad (1024 * 1024)$$

$$2^{30} \text{ bajtów} = 1 \text{ GB} \quad (1024 * 1024 * 1024)$$



Ważniejsze potęgi dwójki

Kolor 24 / 32 bitowy, dźwięk 16 bitowy


$$2^{24} = 2^{10} * 2^{10} * 2^4 = 1024 * 1024 * 16 \approx 16\,000\,000$$

$$2^{32} \approx 1\,000\,000\,000 * 4$$

System dwójkowy - konwersja

KONWERSJA LICZBY DZIESIĘTNEJ DO DWÓJKOWE.

$$(147)_{10} = (?)_2$$

	Reszta:	
$147 : 2 = 73$	$C_0 = 1$	$n = 8$ 
$73 : 2 = 36$	$C_1 = 1$	
$36 : 2 = 18$	$C_2 = 0$	
$18 : 2 = 9$	$C_3 = 0$	
$9 : 2 = 4$	$C_4 = 1$	
$4 : 2 = 2$	$C_5 = 0$	
$2 : 2 = 1$	$C_6 = 0$	
$1 : 2 = 0$	$C_7 = 1$	

$$(147)_{10} = (10010011)_2$$

$$\begin{aligned}
 10010011 &= 1 \cdot 2^7 + 1 \cdot 2^4 + 1 \cdot 2^1 + 1 \cdot 2^0 = \\
 &= 128 + 16 + 2 + 1 = 147
 \end{aligned}$$

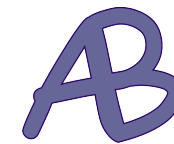
System dwójkowy - arytmetyka

Dodawanie w systemie dwójkowym

$$\begin{array}{r} 11 \\ 00111000 \\ + 00010001 \\ \hline 01001001 \end{array} \qquad \begin{array}{r} 01000001 \\ + 00010100 \\ \hline 01010101 \end{array}$$

Dodawanie jest realizowane podobnie jak dla systemu dziesiętnego, należy jedynie pamiętać, że

$$1_{(2)} + 1_{(2)} = 10_{(2)}$$



Odejmowanie w systemie dwójkowym

00111001	00101101
- 00001101	- 00010001
-----	-----
00101100	00011100

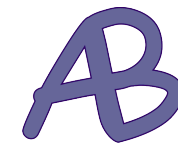
W przypadku odejmowania 0 - 1 w systemie dwójkowym, musimy dokonać zapożyczenia 1 na następnej pozycji liczby.

System dwójkowy - arytmetyka

Mnożenie w systemie dwójkowym

$\begin{array}{r} 1111 \\ \times 101 \\ \hline 1111 \\ 0000 \\ + 1111 \\ \hline 1001011 \end{array}$	$\begin{array}{r} 10001 \\ \times 11 \\ \hline 10001 \\ + 10001 \\ \hline 110011 \end{array}$
--	---

Mnożenie jest wykonywane analogicznie jak mnożenie w systemie dziesiętnym.

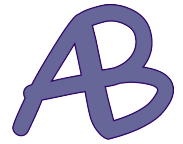


Dzielenie w systemie dwójkowym

$$\begin{array}{r}
 110 \\
 \text{-----} \\
 10010 : 11 = 00000110 \\
 - 11 \\
 \text{-----} \\
 1 \\
 11 \\
 - 11 \\
 \text{-----} \\
 0
 \end{array}$$

$$\begin{array}{r}
 1011 \\
 \text{-----} \\
 1111001 : 1011 = 1011 \\
 - 1011 \\
 \text{-----} \\
 10000 \\
 - 1011 \\
 \text{-----} \\
 1011 \\
 - 1011 \\
 \text{-----} \\
 0
 \end{array}$$

Dzielenie podobnie jak mnożenie wykonujemy tak samo jak w przypadku dzielenia w systemie dziesiętnym.



System szesnastkowy (hexadecymalny)

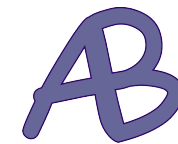
Duże liczby binarne są nieczytelne.

0101001010010010000111100101010010101010110

Celem wprowadzenia systemu szesnastkowego jest skrócenie zapisu bez przeliczania na system dziesiętny.

Każde 4 bity da się przedstawić za pomocą 1 cyfry szesnastkowej – bez żadnego przeliczania.

hex	bin	dec
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15



System szesnastkowy (hexadecymalny)

Przykład:

01010010100100100001111001010100101010101100

0101 0010 1001 0010 0001 1110 0101 0100 1010
1010 1100

0101 0010 1001 0010 0001 1110 0101 0100 1010
5 2 8 2 1 E 5 4 A
1010 1100
A C

52821E54AAC

hex	bin	dec
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

System szesnastkowy (hexadecymalny)

System szesnastkowy podlega tym samym zasadą co inne systemy wagowo – pozycyjne .

16^5 16^4 16^3 16^2 16^1 16^0

4 F D 3 0 D

$$= 4 * 16^5 + 15 * 16^4 + 13 * 16^3 +$$

$$+ 3 * 16^2 + 0 * 16^1 + 13 * 16^0$$

hex	bin	dec
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

Kodowanie liczb ujemnych

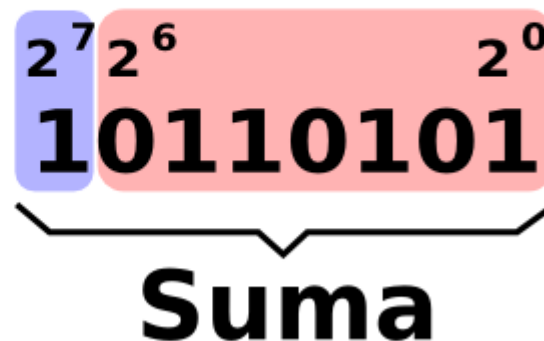


część ujemna

część dodatnia

Kod U2

(Uzupełnień do dwóch)



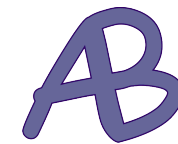
Najmniejsza liczba

$$10000000 = -128 + 0 = -128$$

Największa liczba

$$01111111 = -0 + 127 = 127$$

Kodowanie liczb ujemnych



Problem: wygenerować w KU2 liczbę -5 (przeciwna do +5)

1. Zapisać liczbę (+5);
2. Zamienić wszystkie 1/0 i 0/1;
3. Dodać 1.

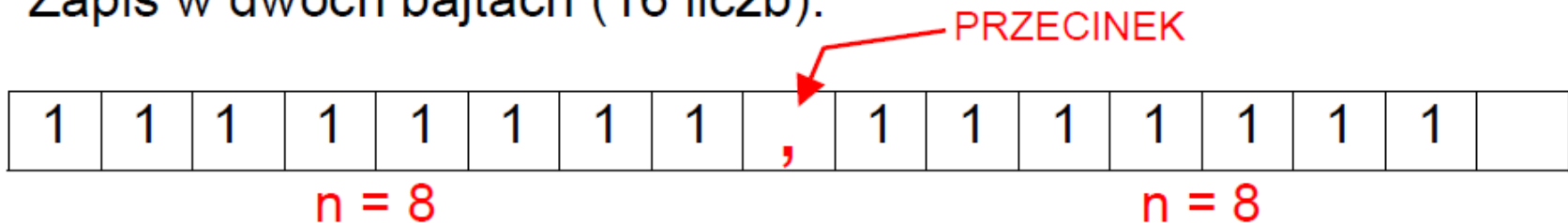
$$\begin{array}{r} \text{00000101} = -0 + 5 = 5 \quad (1) \\ \text{11111010} \quad (2) \\ + \quad \quad \quad 1 \quad (3) \\ \hline \end{array}$$

$$\text{11111011} = -128 + 64 + 32 + 16 + 8 + 2 + 1 = -5$$

Liczby rzeczywiste – zapis stałoprzecinkowy

Liczby rzeczywiste – część całkowita + część ułamkowa

Zapis w dwóch bajtach (16 liczb):



$$2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 255$$

$$2^{-1} + 2^{-2} + 2^{-3} + 2^{-4} + 2^{-5} + 2^{-6} + 2^{-7} + 2^{-8} = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} + \frac{1}{128} + \frac{1}{256} = \frac{255}{256}$$

Powyższy zapis ma same wady:

1. nie można zapisać liczb większych od 255,
2. przy małych liczbach pozostaje dużo wolnego miejsca,
3. MARNOWANIE PAMIĘCI KOMPUTEROWEJ.

Liczby rzeczywiste – zapis stałoprzecinkowy

Błąd przy zapisie liczb:

	Liczba dziesiętna	Część całkowita									Część ułamkowa										
1.	128	1	0	0	0	0	0	0	0	,	0	0	0	0	0	0	0	0	0	1	
2.	1	0	0	0	0	0	0	0	1	,	0	0	0	0	0	0	0	0	0	1	
3.	1/256	0	0	0	0	0	0	0	0	,	0	0	0	0	0	0	0	0	1	1	

„Obcięcie” liczb na dziewiątym miejscu – błąd bezwzględny

$$2^{-9} = \frac{1}{2^9} = \frac{1}{512} = 0,001953125 \text{ - wartość tracona z powodu braku miejsca.}$$

BŁĄD WZGLĘDNY:

Liczba 1: $\cong 0,0015\%$

Liczba 2: $\cong 0,1945\%$

Liczba 3: $\cong 50\%$!

Powyższy sposób zapisu powoduje,
że obliczenia są niewiarygodne
(obliczenia naukowe, ekonomiczne, multimedialne).

Liczby zmiennopozycyjne



5 973 600 000 000 000 000 000 000 kg
(Masa Ziemi)

$$5,9736 \cdot 10^{24}$$

5,9736 E+24

mantysa (precyzja)

cecha (wykładnik)

Notacja naukowa pozwala na kodowanie
bardzo dużych / małych liczb

Liczby zmiennopozycyjne



5,625₁₀

101,101

0,101101*2³

Mantysa znormalizowana dla liczb binarnych należy do przedziału $<\frac{1}{2}, 1)$.

W praktyce oznacza to, że przecinek należy ustawić w taki sposób, aby liczba miała postać:

0,1xxxxxx...

Dzięki normalizacji zapis staje się jednoznaczny.

Liczby zmiennopozycyjne



znak cecha /
 wykładnik mantysa /
 precyzja

1**10010****1010110101**

Standard IEEE 754

pojedyncza precyzja	1	8	23	(32 bity)
---------------------	---	---	----	-----------

podwójna precyzja	1	11	52	(64 bity)
-------------------	---	----	----	-----------

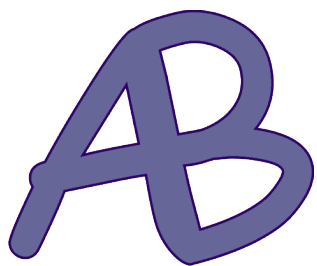


Przykład

1000110110110101

1 **00011** **0110110101**

$$\begin{aligned} & - 0,110\ 110101 * 2^3 = \\ & = 110,110101 = 6,828125_{10} \end{aligned}$$



Reprezentacja danych w komputerze

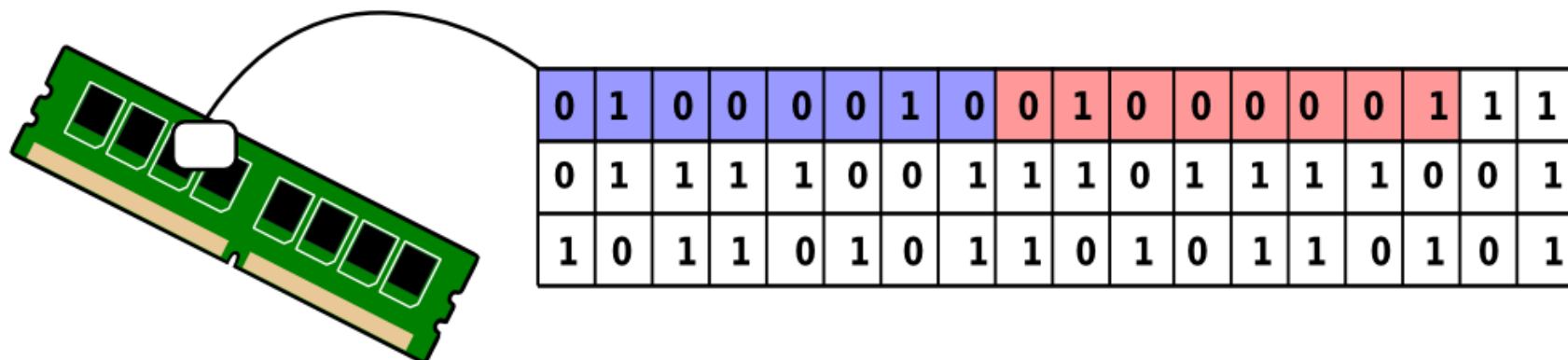
Kod ASCII

American Standard Code for Information Interchange

Kod przypisujący 7-bitowe (128 kombinacji) ciągi do znaków.

A 01000001
B 01000010

7 bitów



Kod ASCII



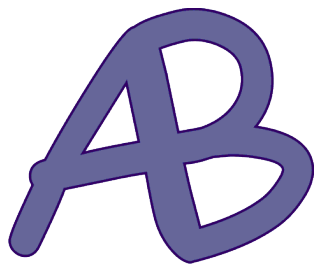
Regionalne strony kodowe

A	01000001
B	01000010

└──────────┘
7 bitów

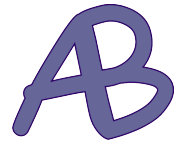
128 kombinacji wystarcza do zakodowania wszystkich liter i cyfr oraz kilkudziesięciu znaków drukowalnych (+ - =...) i niedrukowalnych znaków sterujących (np. nowy wiersz).

Rozbudowanie kodu do 8 bitów pozwala na przypisanie znaków narodowych (ąęäö...). Przykładowo Europa Centralna używa dla swoich alfabetów rozszerzenia iso-8859-2, a Europa Zachodnia iso-8895-1.



Pojęcie algorytmu

Trochę historii



Pierwsze opisy, które później nazwano algorytmami, dotyczyły rozwiązań zadań matematycznych.

Pomiędzy 400 a 300 rokiem p.n.e. grecki matematyk i filozof **Euklides**, wymyślił pierwszy znany nam nietrywialny algorytm, czyli przepis na realizację zadania. Był to algorytm znajdowania największego wspólnego dzielnika dwóch dodatnich liczb całkowitych.



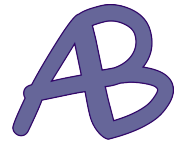
Trochę historii



Słowo **algorytm** pochodzi od nazwiska matematyka arabskiego, który żył na przełomie VIII i IX wieku naszej ery.

Muhammad ibn Musa al-Chorezmi zasłużył się stworzeniem kilku dzieł z dziedziny matematyki, w których opisał dużą ilość reguł matematycznych (w tym dodawania, odejmowania, mnożenia i dzielenia zwykłych liczb dziesiętnych). Opis tych procedur był na tyle precyzyjny i formalny, jak na tamte czasy, że właśnie od jego nazwiska pochodzi słowo algorytm.

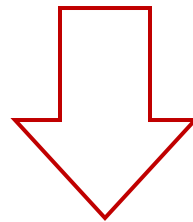




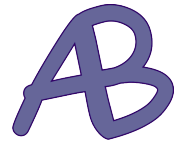
Intuicyjnie algorytm kojarzy się z metodą rozwiązywania zadania, przepisem postępowania czy też ze schematem działania.

Należy jednak podkreślić, że nie każda metoda czy schemat jest algorytmem.

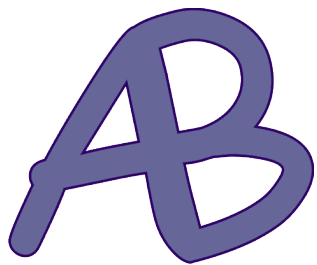
Algorytm powinien spełniać sześć warunków.



Cechy algorytmu



1. Musi posiadać określony **stan początkowy**, czyli operację od której zaczyna się jego realizacja.
2. Ilość operacji potrzebnych do zakończenia pracy musi być skończona - **warunek dyskretności (skończoności)**.
3. Musi dać się zastosować do rozwiązywania całej klasy zagadnień, a nie jednego konkretnego zadania - **warunek uniwersalności**.
4. Interpretacja poszczególnych etapów wykonania musi być jednoznaczna - **warunek jednoznaczności**.
5. Cel musi być osiągnięty w akceptowalnym czasie - **warunek efektywności**.
6. Musi posiadać **wyróżniony koniec**.



Notacja algorytmów

Metody zapisu algorytmu

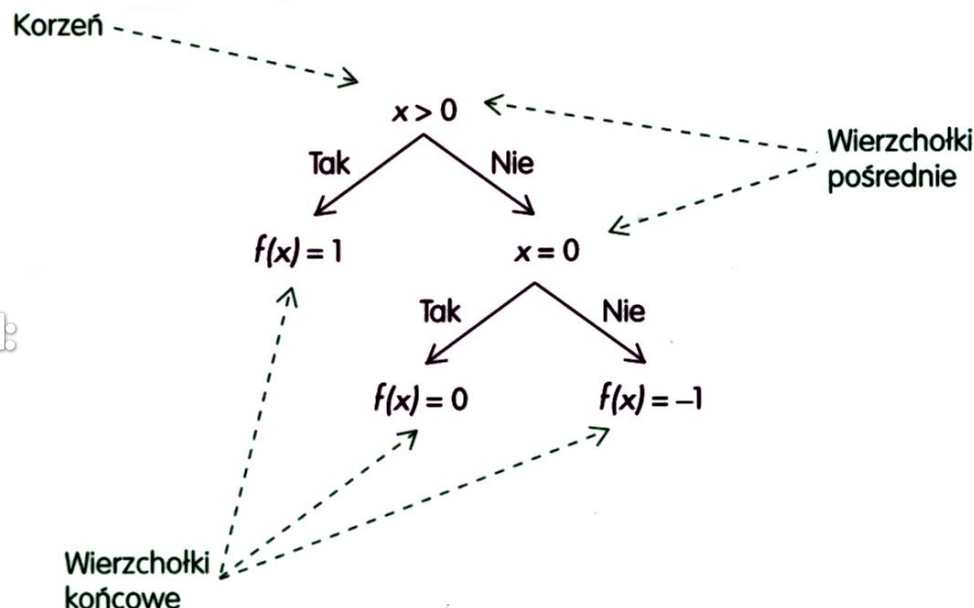
1. Opis słowny za pomocą ograniczonego podzbioru języka naturalnego

Przykład:

1. Dane są dwie liczby naturalne a i b .
2. Oblicz c jako resztę z dzielenia a przez b
3. Zastąp a przez b , zaś b przez c .
4. Jeżeli $b = 0$, to szukane NWD wynosi a , w przeciwnym wypadku wróć do punktu drugiego i kontynuuj.

2. Drzewo algorytmu

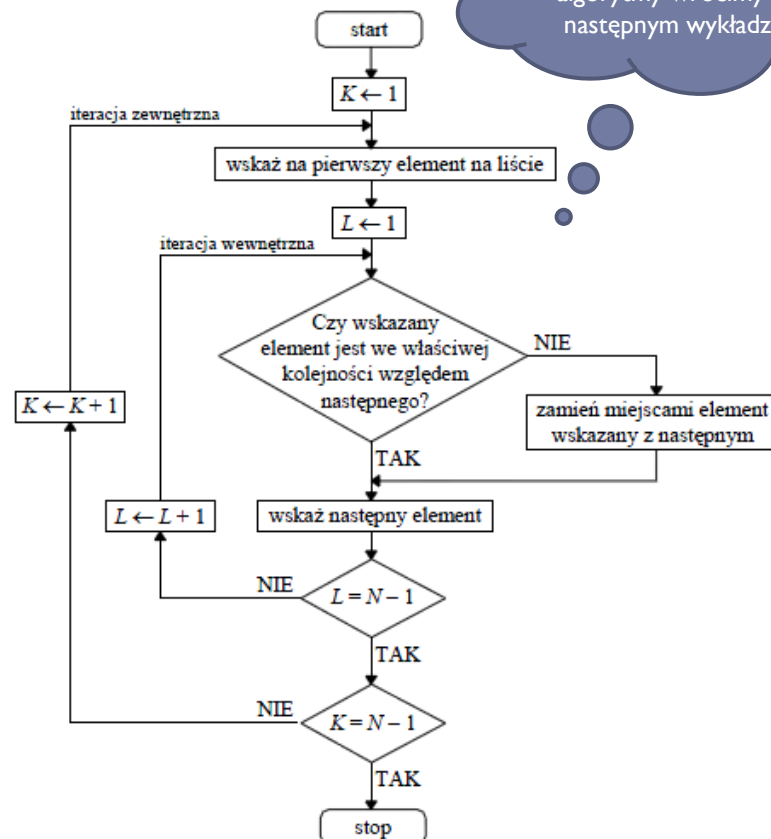
Przykład:



3. Schematy blokowe.

Schemat blokowy sortowania bąbelkowego:

Przykład:



Na razie potraktujmy to jako przykład metody zapisu - do tego algorytmu wrócimy na następnym wykładzie

Metody zapisu algorytmu

3. Pseudo-język.

Inną metodą przedstawienia algorytmu jest użycie zapisu za pomocą pseudo-języka programowania.

Zaletą tego podejścia jest bardzo łatwa implementacja algorytmu za pomocą konkretnie wybranego, istniejącego języka programowania.

Wadą jest mniejsza przejrzystość zapisu.

Algorytm Euklidesa w pseudokodzie:

NWD(liczba całkowita a, liczba całkowita b)

dopóki b różne od 0

c := reszta z dzielenia a przez b

a := b

b := c

zwróć a

Przykład:

Istnieją różne wersje pseudo-języka.
Najczęściej jest to PASCAL pozbawiony informacji dla kompilatora (i czasem przetłumaczony na polski)

Zapis algorytmów – zmienne i operatory

Zmienna to w programowaniu element programu, który może mieć przypisaną pewną wartość (wartość może być różna w różnych momentach wykonania programu). Zmienna jest uchwyttem do tej wartości.

- ✓ W większości języków programowania (poza językami najwyższego poziomu) **zmienne musimy zadeklarować**, czyli poinformować kompilator, o tym że taka zmienna wystąpi i o tym jaki typ danych zamierzamy w niej przechowywać.
- ✓ Umożliwia to kompilatorowi zarezerwowanie odpowiedniego miejsca w pamięci operacyjnej i dobrane właściwych procedur obliczeniowych (na poziomie języka maszynowego).

Dziś spotkamy typy:

- REAL (liczba rzeczywista)
- INTEGER (całkowita)

Zapis algorytmów – zmienne i operatory

Operatory stosowane w pseudo-języku oraz w schematach blokowych:

+ **-** ***** **/**

- chyba nie wymagają komentarza

%

- operator reszty z dzielenia całkowitoliczbowego

sqr (..)

- kwadrat

sqrt (..)

- pierwiastek kwadratowy

==

- pytanie „czy jest równe”

!=

- pytanie „czy jest różne” (\neq)

> i **<**

- pytanie czy jest większe i czy jest mniejsze

>= i **<=**

- większe lub równe (\geq) i mniejsze lub równe (\leq)

=

- operator przypisania (podstawienia) (alternatywnie **:=**)

Zapis algorytmów – zmienne i operatory

Dwie ważne uwagi:

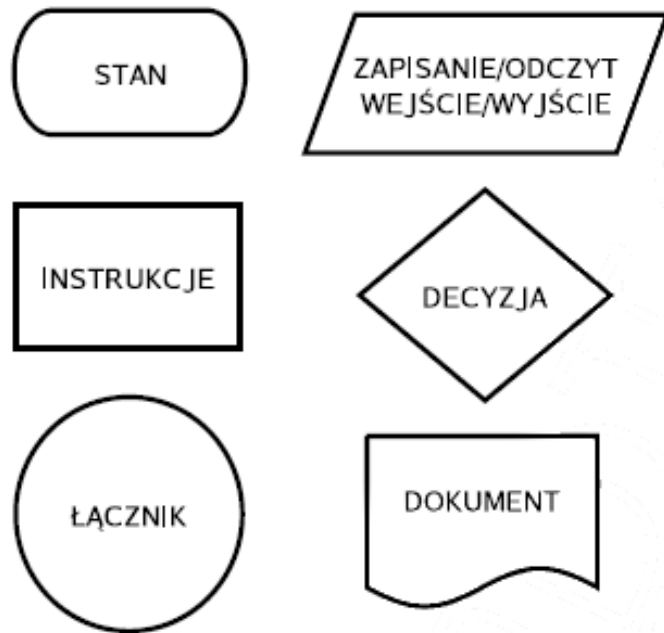
- ✓ Zmienna w programie komputerowym (i algorytmie) to nie to samo co zmienna w zadaniu matematycznym.
- ✓ Rozróżniaj operatory:

== - pytanie czy równe
= - operator przypisania

$x = x + 1$ - w matematyce jest to równie sprzeczne

$x = x + 1$ - w języku programowania - operacja podstawienia
(wartość zapisaną w zmiennej x zwiększamy o 1)

Zapis algorytmów – schemat blokowy

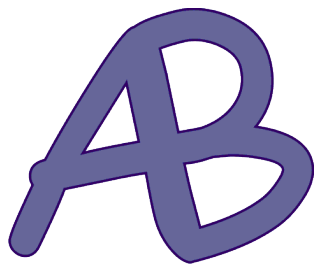


- ✓ **Stan** - Określa zwykle moment startu i końca.
- ✓ **Zapis/odczyt** - Wskazuje miejsce w których odbywa się zapis danych (lub ich odczyt).
- ✓ **Instrukcje** - Blok instrukcji, które mają być wykonane.
- ✓ **Decyzja** - Wyliczenie warunku logicznego znajdującego się wewnątrz symbolu i podjęcie na jego podstawie decyzji.
- ✓ **Łącznik** - Połączenie z inną częścią schematu blokowego, np. gdy nie mieści się on na jednej stronie.

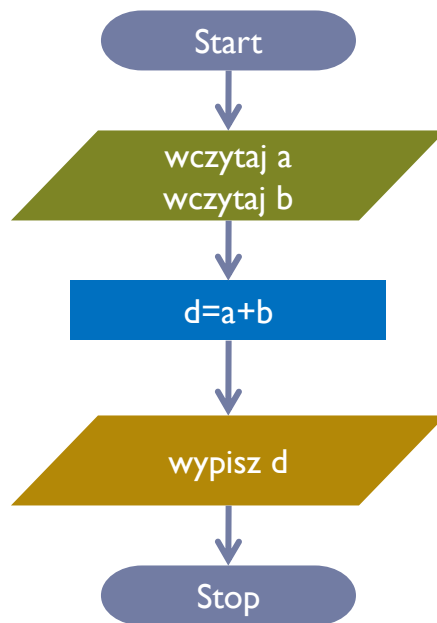
Zapis algorytmów – schemat blokowy

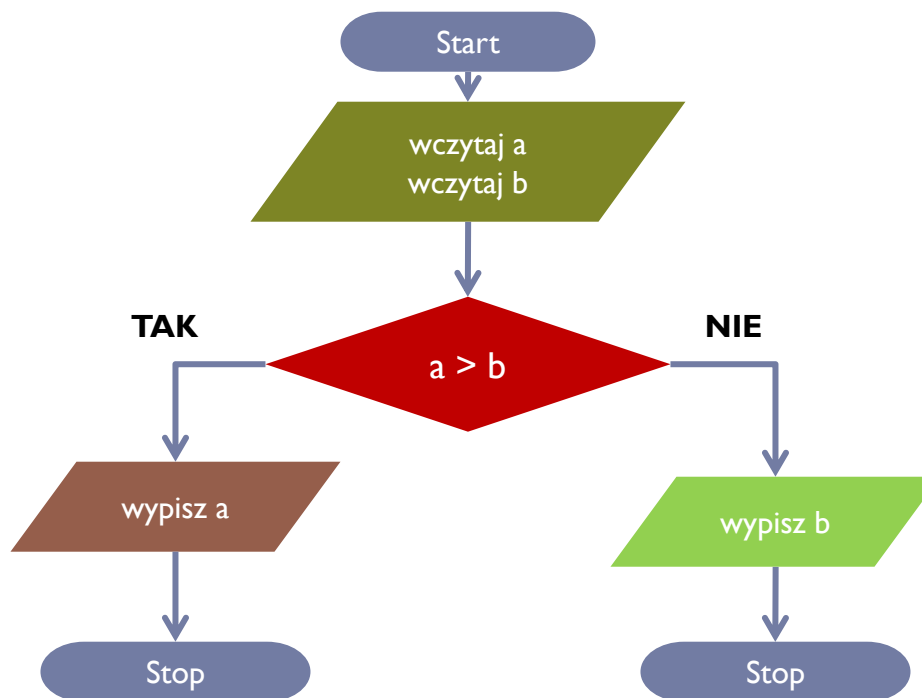
Schemat blokowy tworzony jest według następujących reguł:

1. Schemat blokowy składa się z bloków połączonych zorientowanymi liniami;
2. Bloki obrazują ciąg operacji;
3. Zawsze wykonywane są albo wszystkie instrukcje w bloku albo żadna;
4. Dalsze operacje nie zależą od poprzednich wariantów, chyba że zależności te zostały przekazane za pomocą danych;
5. Kolejność wykonania operacji jest ściśle określona przez zorientowane
6. Linie łączące poszczególne bloki;
7. Do każdego bloku może prowadzić co najwyżej jedna linia;
8. Linie mogą się łączyć ale nie mogą się rozdzielać (bez bloku decyzyjnego).



Rodzaje algorytmów

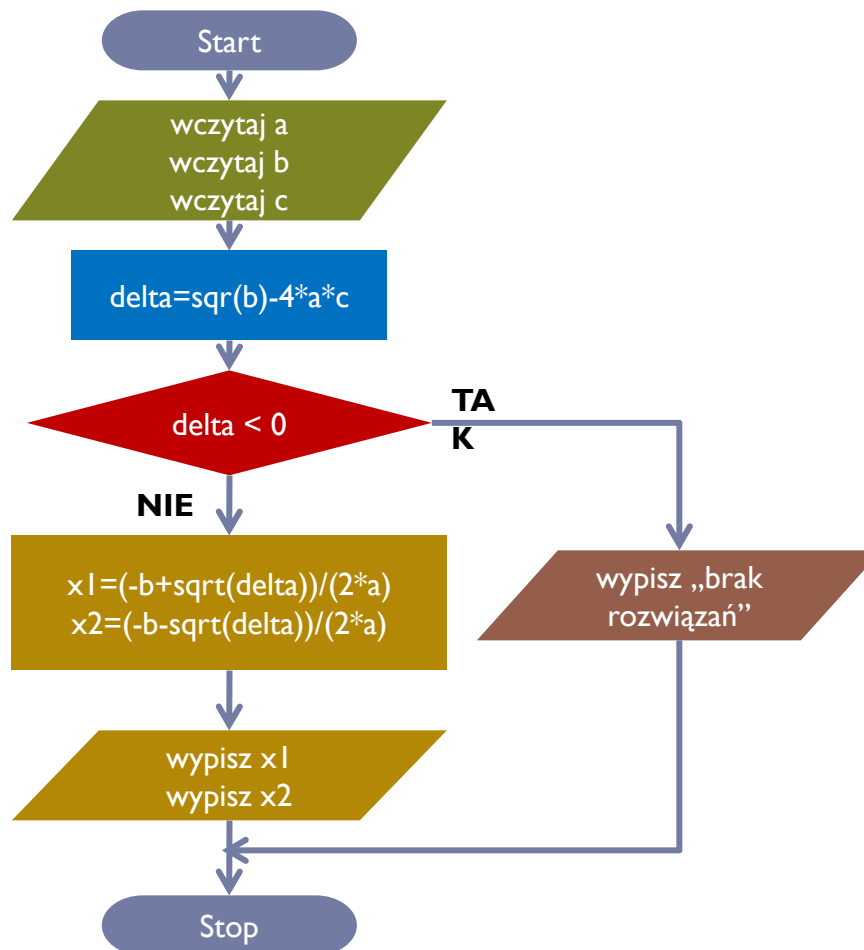




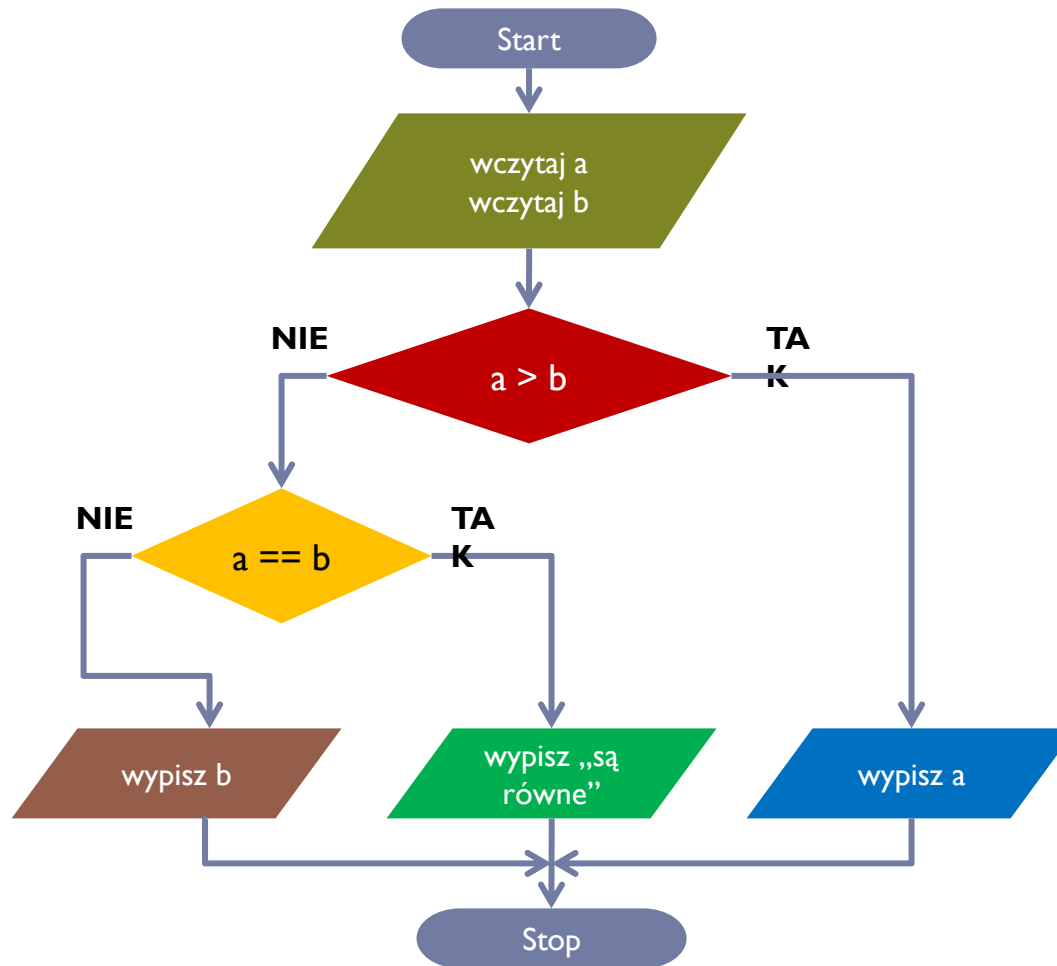
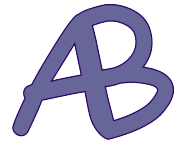
Algorytmy rozgałęzione – przykład

Przykład:

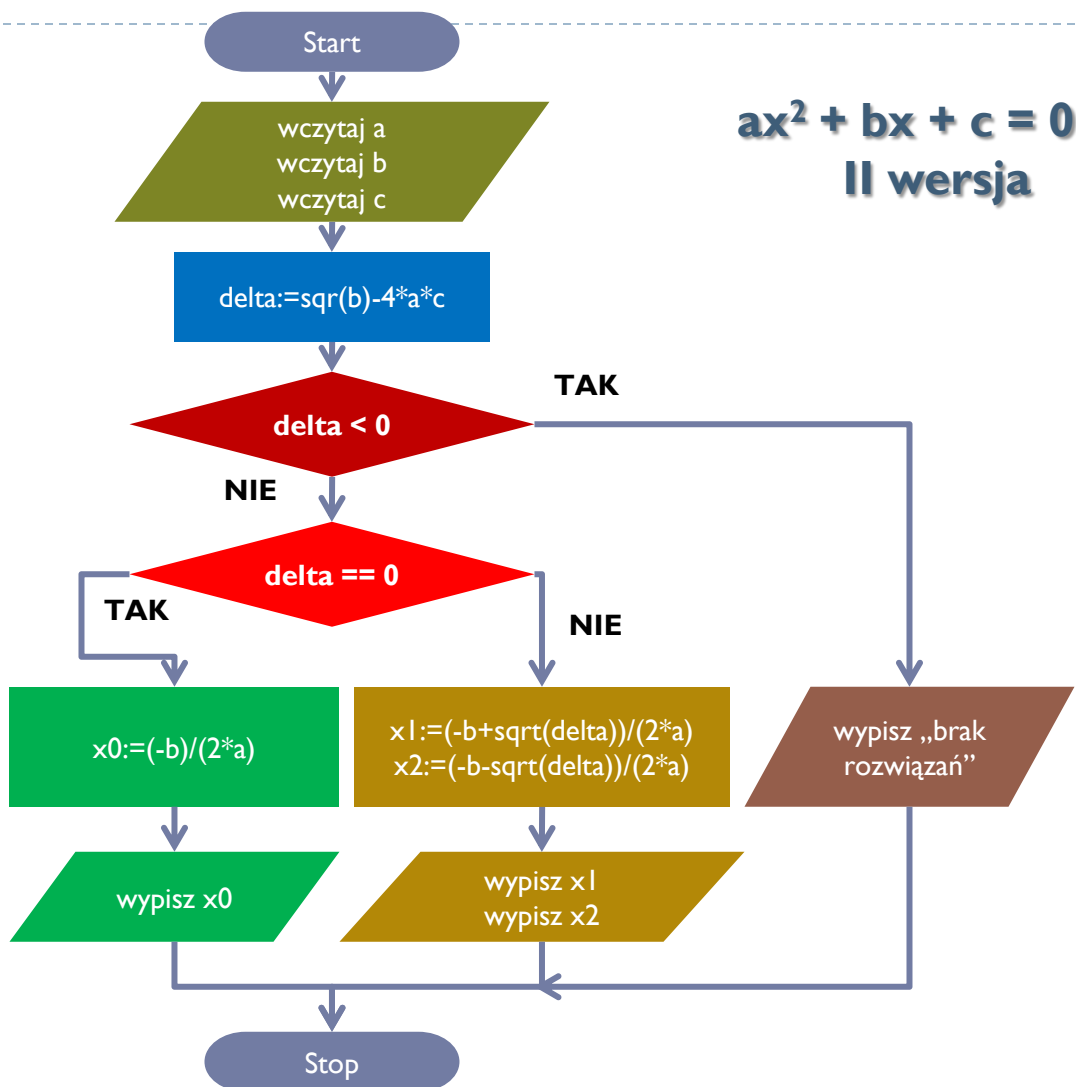
$$ax^2 + bx + c = 0$$



Algorytmy rozgałęzione wielokrotnie



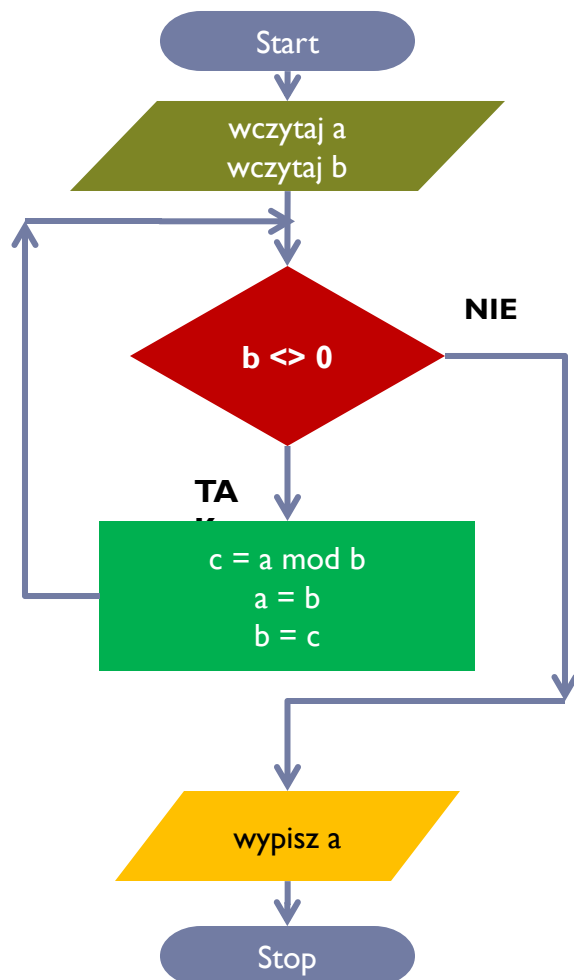
Algorytmy rozgałęzione wielokrotnie



Algorytmy iteracyjne



NWW – algorytm Euklidesa



Zapis w pseudokodzie

```
NWD(liczba całkowita a, liczba całkowita b)
dopóki b różne od 0
    c = reszta z dzielenia a przez b
    a = b
    b = c
zwróć a
```

Literatura:

W prezentacji wykorzystano fragmenty i zadania z książek i stron internetowych:

- Piotr Fulmański, Ścibór Sobieski, **Wstęp do informatyki, Podręcznik**, Wydawnictwo Uniwersytetu Łódzkiego, 2005
ISBN: 83-7171-844-6
- Maciej M. Sysło, **Algorytmy**, WSiP, Warszawa 2002, ISBN: 83-02-06659-1
- <http://pl.wikipedia.org>