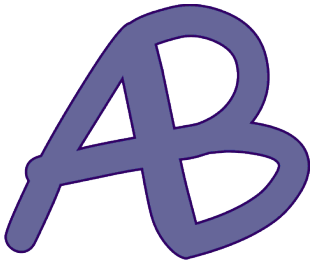


Wykład II

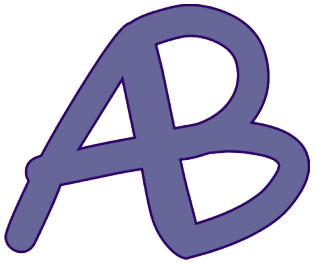
Algorytmy – metody prezentacji i zapisu

C++

Wybór wielokrotny
Pętle

www.bartoszewski.uniwersytetradom.pl

Część 1V



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.

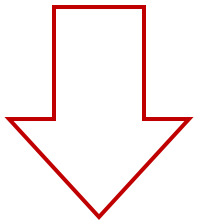
Muhammad ibn Musa al-Chorezmi zasłużył się stworzeniem kilku dzieł z dziedziny matematyki, w których opisał dużą ilość reguł matematycznych (w tym dodawania, odejmowania, mnożenia i dzielenia zwykłych liczb dziesiętnych). Opis tych procedur był na tyle precyzyjny i formalny, jak na tamte czasy, że właśnie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

Cechy algorytmu

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

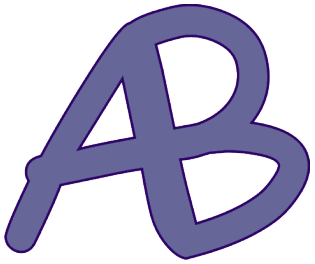
Należy jednak podkreślić, że nie każda metoda czy schemat jest algorytmem.

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

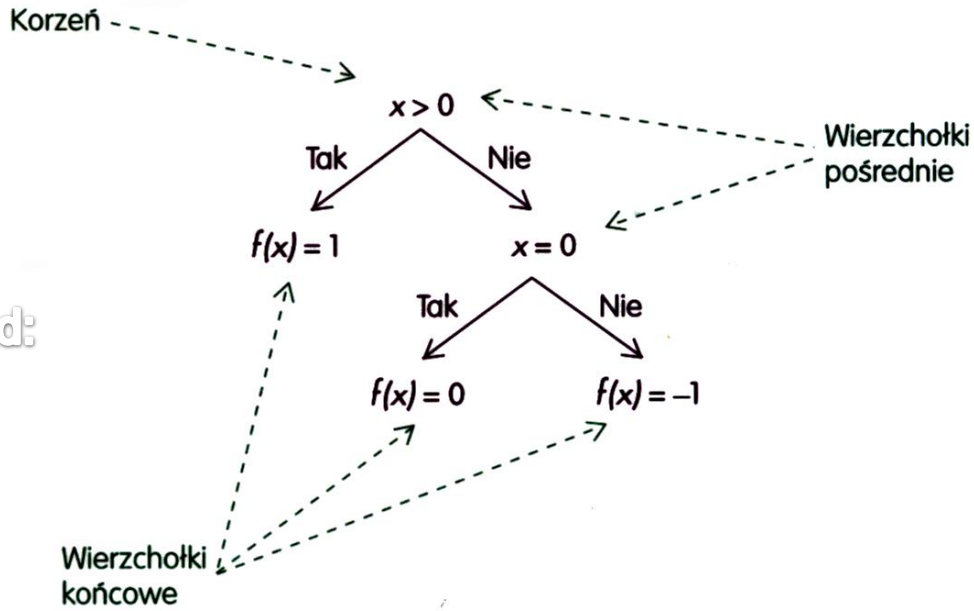
1. Opis słowny za pomocą ograniczonego podzbioru języka naturalnego

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:



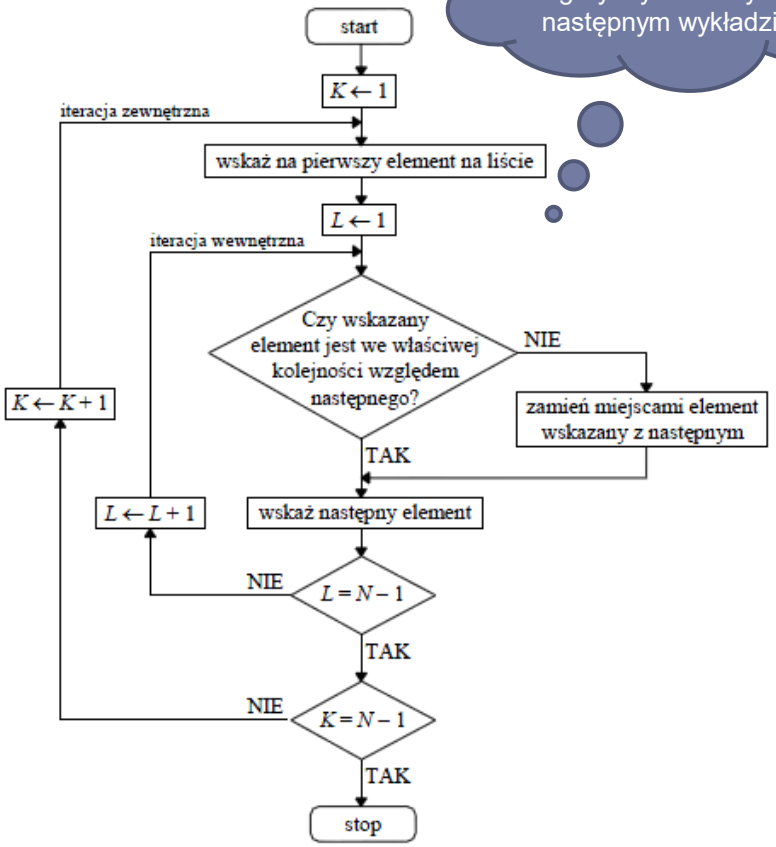
Metody zapisu algorytmu

3. Schematy blokowe.

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

Przykład:

Schemat blokowy sortowania bąbelkowego:



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:

Przykład:

```
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
```

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

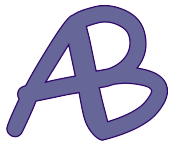
Zapis algorytmów – zmienne i operatory

Zmienna to w programowaniu element programu, który może mieć przypisaną pewną wartość (wartość może być różna w różnych momentach wykonania programu). Zmienna jest uchwytem 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 czyn 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:

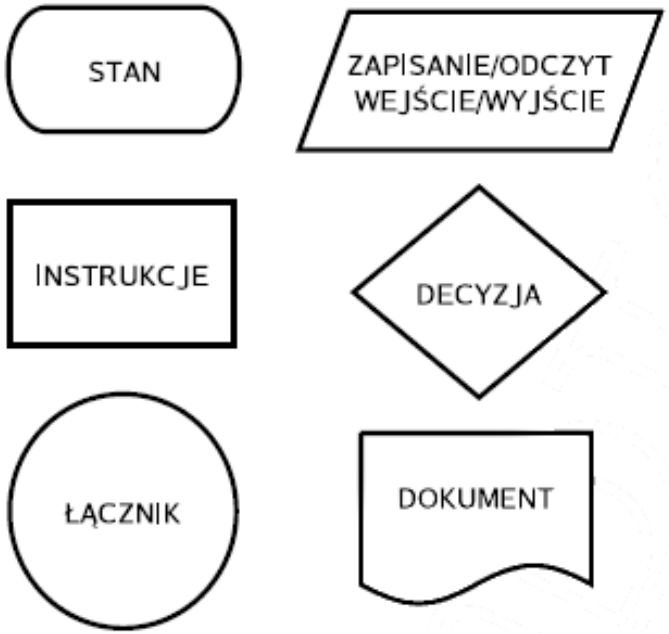
  == - pytanie czy równe

  = - operator przypisania

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

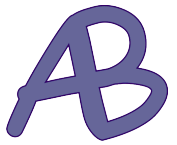
x = x + 1 - w języku programowania - operacja podstawienia
(wartość zapisaną w zmiennej x zwiększamy o 1)

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.

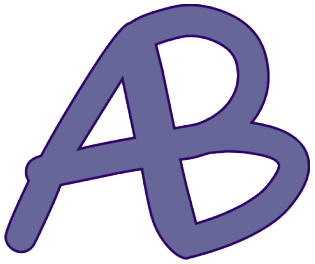
Zapis algorytmów – schemat blokowy



Schemat blokowy tworzony jest według następujących reguł:

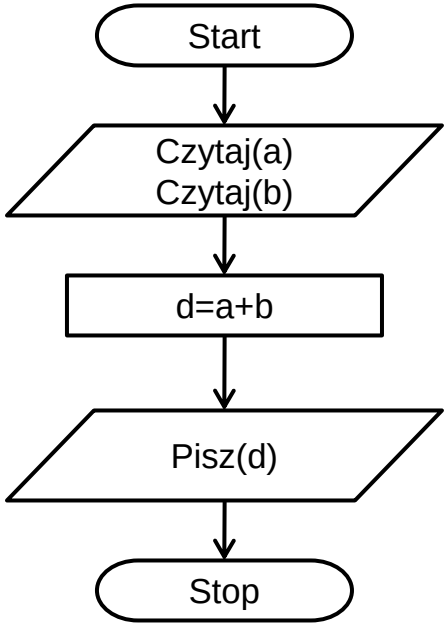
1. Schemat blokowy składa się z bloków połączonych zorientowanymi liniami;
2. Bloki obrazują ciąg operacji;
3. Zawsze wykonywane są albo wszystkie instrukcje w bloku albo żadna;
4. Dalsze operacje nie zależą od poprzednich wariantów, chyba że zależności te zostały przekazane za pomocą danych;
5. Kolejność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

Część VI

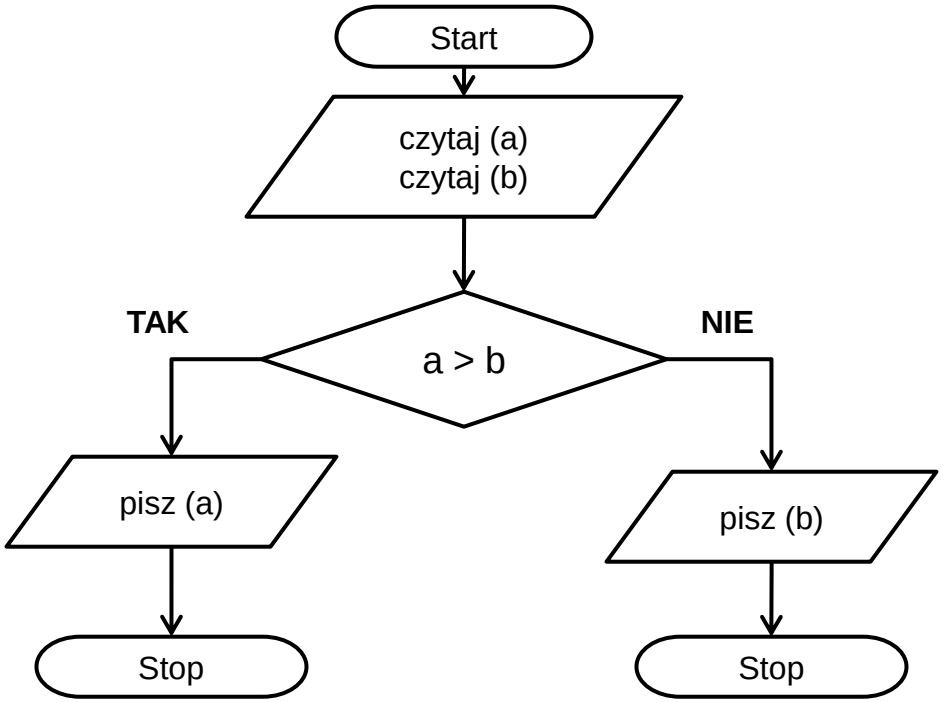


Rodzaje algorytmów

Algorytmy liniowe



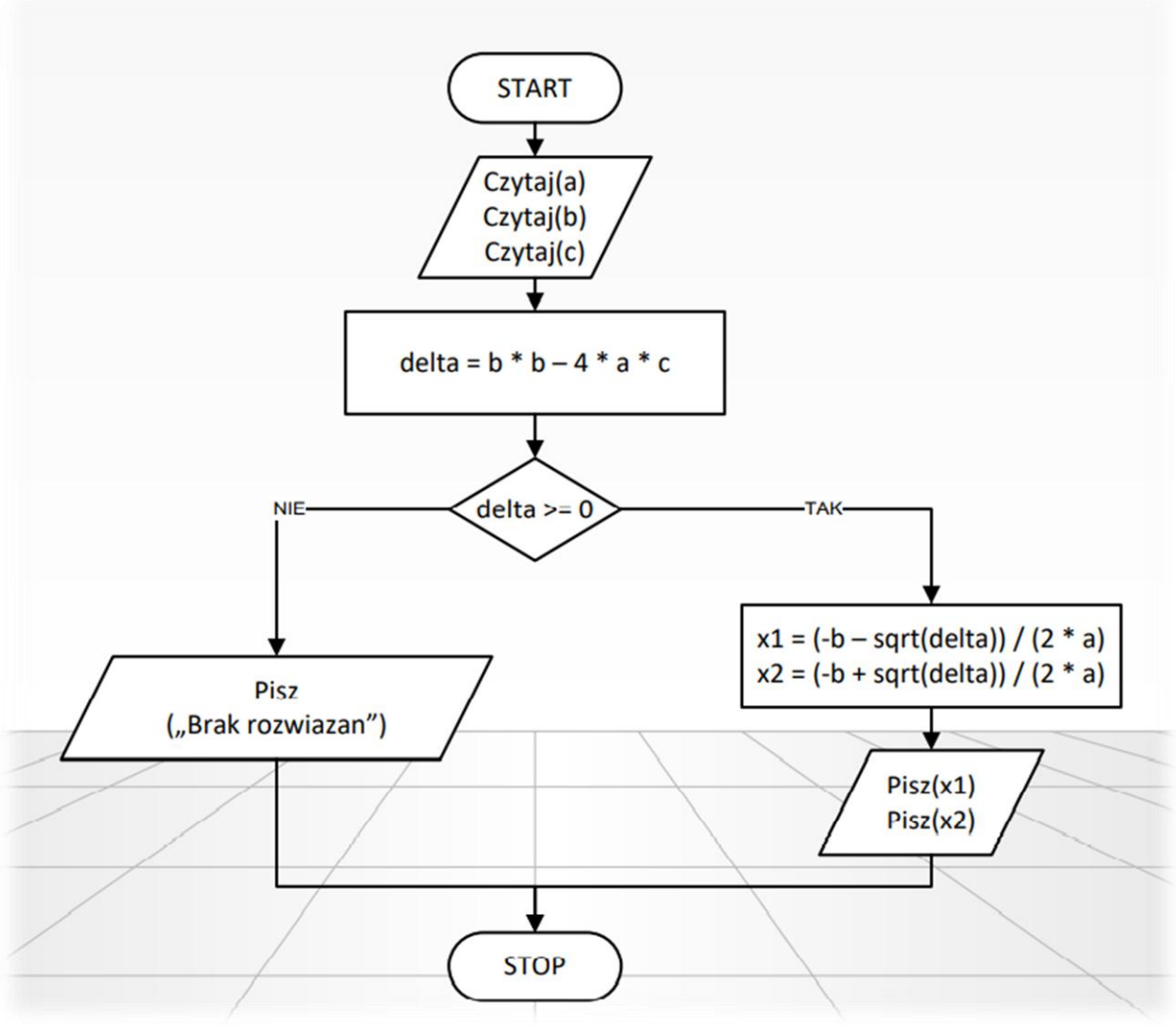
Algorytmy rozgałęzione



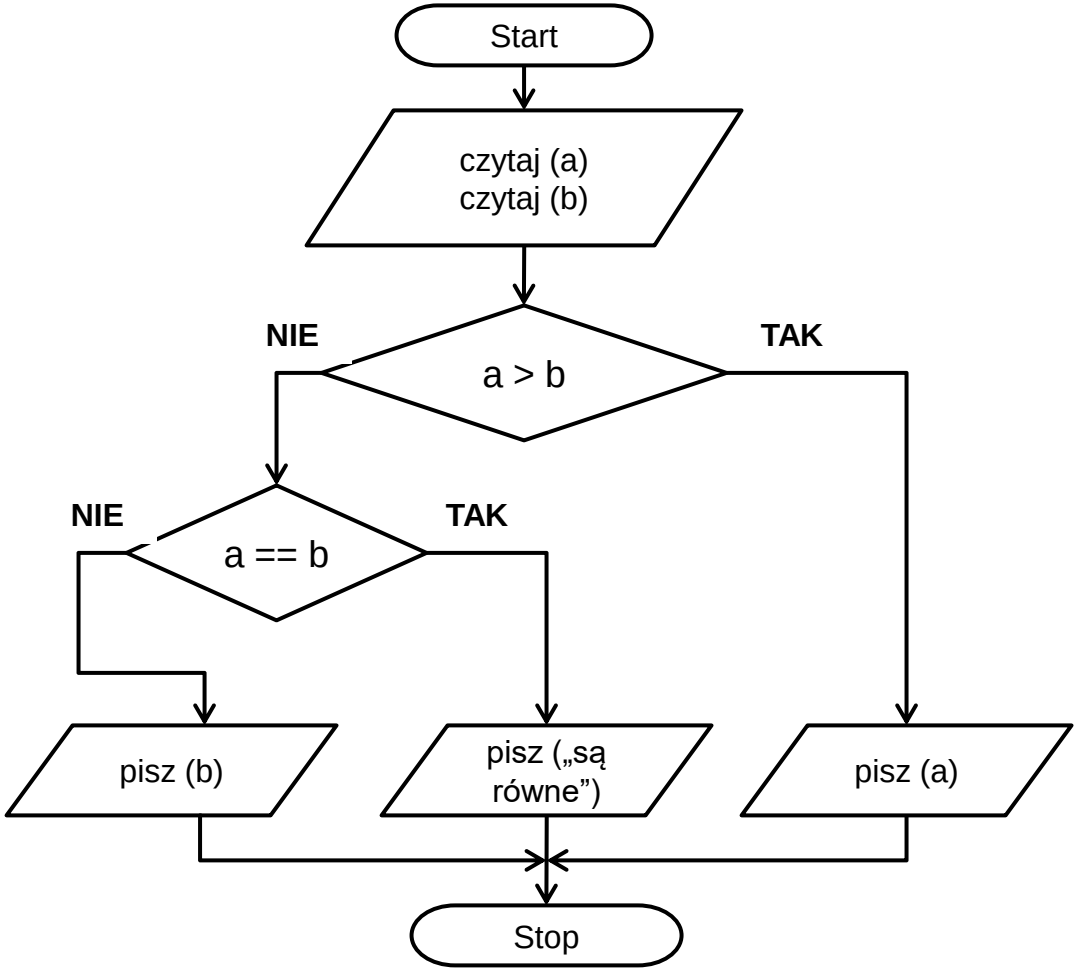
Algorytmy rozgałęzione – przykład

Przykład:

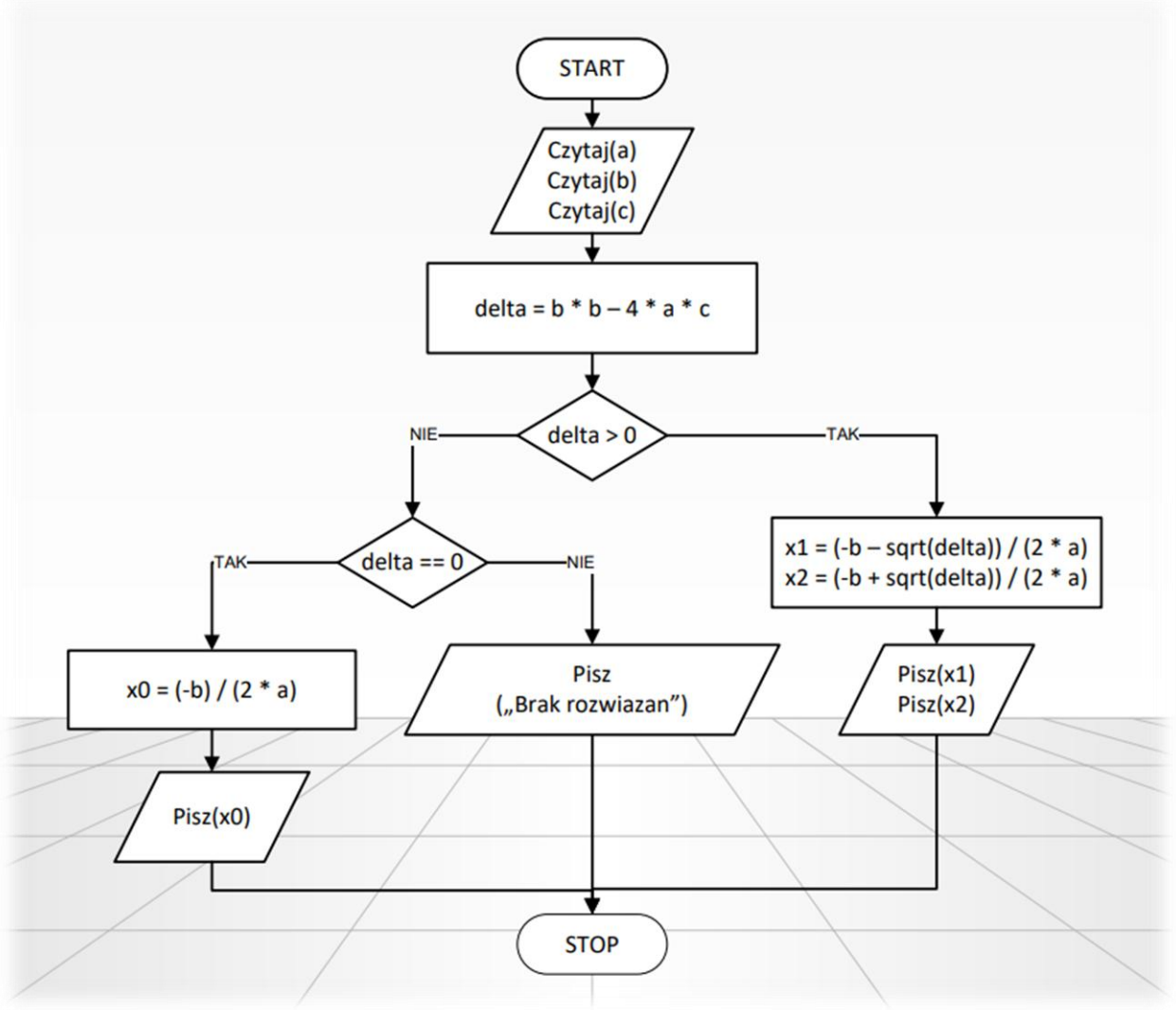
$ax^2 + bx + c = 0$



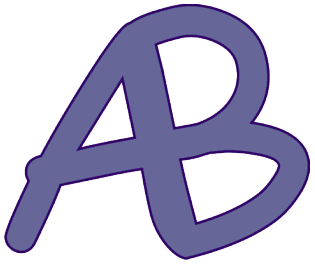
Algorytmy rozgałęzione wielokrotnie



Algorytmy rozgałęzione wielokrotnie



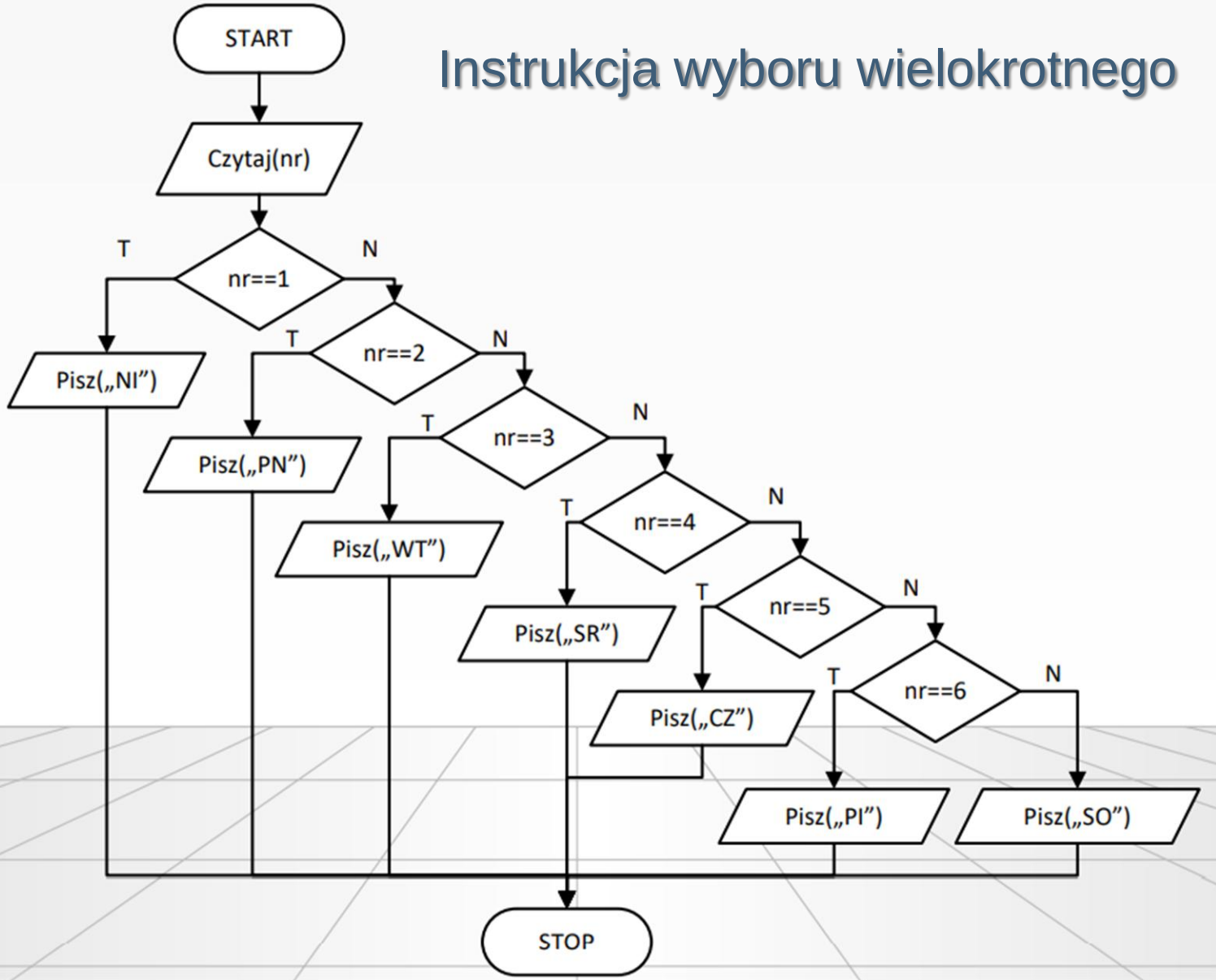
$ax^2 + bx + c = 0$
II wersja



C++

Wybór wielokrotny

Instrukcja wyboru wielokrotnego



Wyboru wielokrotnego za pomocą if

```
1   #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int nr;
6       cout << "Podaj numer dnia tygodnia";
7       cin >> nr;
8       if (nr == 1)
9           cout << "Niedziela";
10      else if (nr == 2)
11          cout << "Poniedziałek";
12      else if (nr == 3)
13          cout << "Wtorek";
14      else if (nr == 4)
15          cout << "Sroda";
16      else if (nr == 5)
17          cout << "Czwartek";
18      else if (nr == 6)
19          cout << "Piatek";
20      else
21          cout << "Sobota";
22      return 0;
23  }
```

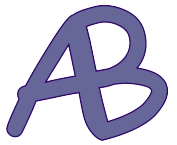
```
20      else if (nr==7)
21          cout << "Sobota";
22          else cout <<"To nie jest prawidłowy numer";
23      return 0;
24  }
```

Wersja zapewniająca kontrole poprawności danych

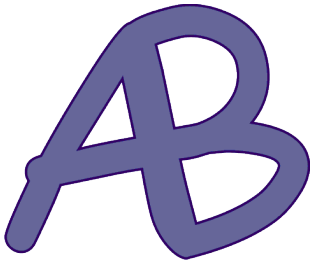
Instrukcja wyboru wielokrotnego switch

```
switch (zmienna)
{
  case wartosc_1:  instrukcja_1;  break;
  case wartosc_2:  instrukcja_2;  break;
  case wartosc_3:  instrukcja_3;  break;
  default: instrukcja_defaltowa;
}
```

Instrukcja wyboru wielokrotnego switch



```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5    int d;
6    cout << "Podaj nr dnien tygodnia:\t";
7    cin >> d;
8    switch (d)
9    {
10     case 1: cout<< endl << "Niedziela";        break;
11     case 2: cout<< endl << "Poniedzialek";     break;
12     case 3: cout<< endl << "Wtorek";           break;
13     case 4: cout<< endl << "Sroda";            break;
14     case 5: cout<< endl << "Czwartek";         break;
15     case 6: cout<< endl << "Piatek";           break;
16     case 7: cout<< endl << "Sobota";           break;
17     default: cout <<endl
18       << "to juz nie w tym tygodniu";
19   }
20   return 0;
21 }
```



C++

Iteracja

Pętla for

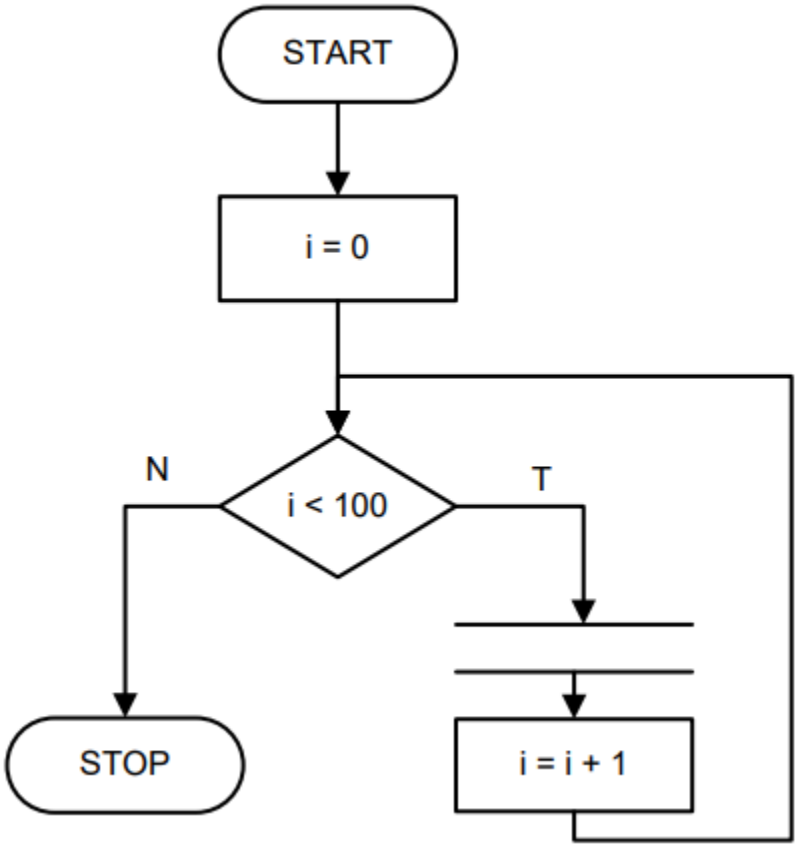
```
for ( instrukcja_ini ; wyrazenie_warunkowe ; instrukcja_krok )
        tresc_petli ;
```

- instrukcja_ini - instrukcja wykonywana zanim pętla zostanie po raz pierwszy uruchomiona
- wyrazenie_warunkowe – wyrażenie obliczane przed każdym obiegiem pętli. Jeżeli jest ono różne od zera, to pętla będzie dalej wykonywana
- instrukcja_krok – instrukcja wykonywana po zakończeniu każdego obiegu pętli

Pętla for

Pętla wykonywana 100 razy

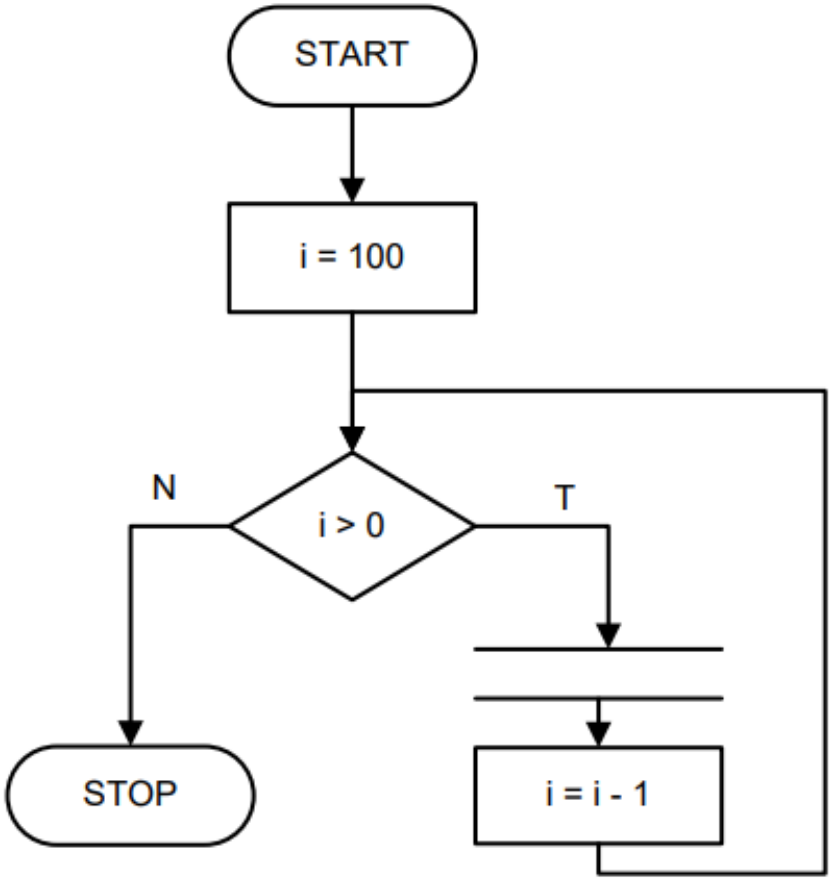
```
for (int i=0; i<100; i++)
{
    // Instrukcje do wykonania
}
```



Pętla for

Można tez odliczać do tyłu

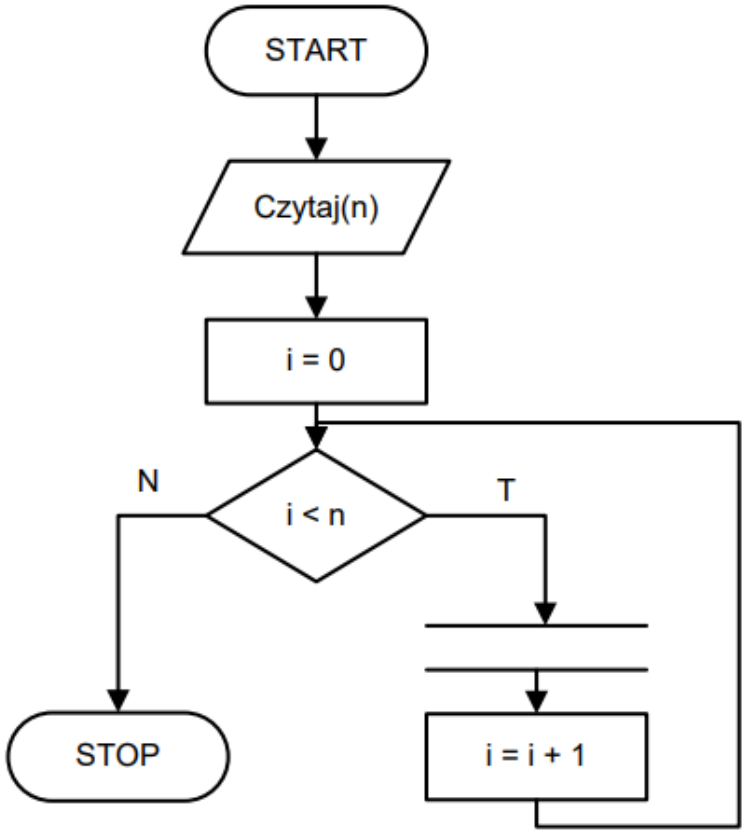
```
for (int i=100; i>0; i--)
{
    // Instrukcje do wykonania
}
```



Pętla for

Pętla wykonywana „n” razy
(n wczytane z klawiatury)

```
int n;
cin>>n;
for (int i=0; i<n; i++)
{
    // Instrukcje do wykonania
}
```



Pętla for - przykład

```
1   #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int n;
6       cout << "n=";
7       cin >> n;
8       long mianownik = 1;
9       double suma = 0;
10      for (int i = 1; i <= n; i++)
11      {
12          mianownik = mianownik * 2;
13          suma = suma + (1.0 / mianownik);
14          cout << mianownik << " ";
15      }
16      cout << "x" << n << "=" << suma;
17      return 0;
18  }
```

$$x_n = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots + \frac{1}{2^n}$$

Literatura:

W prezentacji wykorzystano przykłady i fragmenty:

- Gręboszl J. : ***Symfonia C++, Programowanie w języku C++ orientowane obiektowo***, Wydawnictwo Edition 2000.
- Jakubczyk K.: *Turbo Pascal i Borland C++ Przykłady*, Helion.

Warto zajrzeć także do:

- Sokół R. : ***Microsoft Visual Studio 2012 Programowanie w C i C++,*** Helion.
- Kerninghan B. W., Ritchie D. M.: ***język ANSI C***, Wydawnictwo Naukowo Techniczne.

Dla bardziej zaawansowanych:

- Gręboszl J. : ***Pasja C++,*** Wydawnictwo Edition 2000.
- Meyers S.: ***język C++ bardziej efektywnie***, Wydawnictwo Naukowo Techniczne