BUDOWA i ANALIZA ALGORYTMÓW

dr inż. Jarosław Sikorski

Wydział Informatyki WIT

e-mail: J.Sikorski@wit.edu.pl

Program wykładu:

1. Czym jest algorytm?
2. Podstawowe instrukcje sterujące
3. Struktura algorytmu – podprogramy
4. Algorytmy rekurencyjne
5. Podstawowe struktury danych
6. Przykładowe struktury statyczne
7. Dynamiczne struktury wskaźnikowe
8. Metody algorytmiczne
9. Formalna poprawności algorytmów
10. Analiza złożoności algorytmów
11. Klasy złożoności problemów algorytmicznych

Literatura:

- D.Harel *„Rzecz o istocie informatyki. Algorytmika”* WNT (2001)
- T.Cormen, C.Leiserson, R.Rivest *„Wprowadzenie do algorytmów”* WNT (2005)
- A.J.Aho, J.E.Hopcroft, J.D.Ullman *„Struktury danych i algorytmy”* PWN (1983)
- A.J.Aho, J.E.Hopcroft, J.D.Ullman *„Projektowanie i analiza algorytmów komputerowych”* Helion (2003)

Zadanie:

Wyznacz wartość sumy n podanych liczb: $a_1, a_2, ..., a_n$

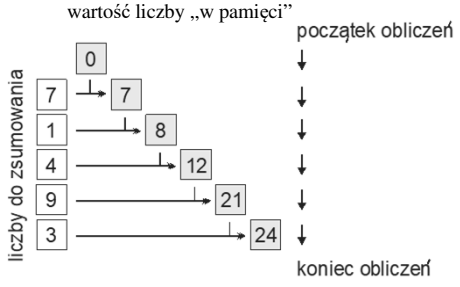
Rozwiązanie analityczne:

$$s = a_1 + a_2 + ... + a_n$$

Rozwiązanie algorytmiczne:

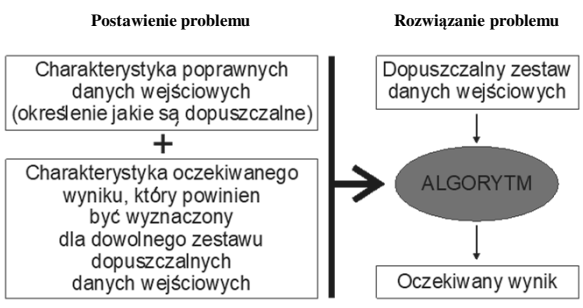
1. zapisz „w pamięci” liczbę 0,
2. odczytaj pierwszą liczbę,
3. wykonaj co następuje n razy:
   3.1. dodaj odczytaną liczbę do liczby „w pamięci” i zapisz wynik „w pamięci”,
   3.2. odczytaj następną liczbę,
4. jako wynik podaj liczbę „w pamięci”.

Przykład zastosowania algorytmu do zsumowania 5 liczb:



Ten sam algorytm może równie dobrze posłużyć do zsumowania 5 000 000 liczb

Co to jest problem algorytmiczny?



Złożoność i rozmiar rzeczywistych zadań stanowią wciąż wyzwanie dla algorytmiki

Dane wejściowe:	**poprawna sytuacja na szachownicy**
Oczekiwany wynik:	**najlepszy ruch białych**

Dane wejściowe:	**sieć dystrybucji, środki transportu i inne potrzebne zasoby**
Oczekiwany wynik:	**plan dostaw o najniższym koszcie**

ALGORYTM
Struktura
Instrukcje sterujące
Operacje podstawowe

↓

PROCESOR

A →B→C→…→X→Y→Z
Kolejno wykonywane operacje podstawowe

Postulat Długotrwałe procesy wyznaczania wyniku powinny być zapisane możliwie krótkimi algorytmami

Założenie Każda operacja podstawowa wykonywana jest w skończonym czasie

Co musimy uzgodnić przed rozpoczęciem budowy algorytmu?

Z czego go będziemy budować! **Czyli:**

- **trzeba uzgodnić zestaw operacji podstawowych**
- **trzeba uzgodnić zestaw instrukcji sterujących**
- **trzeba uzgodnić formę zapisania danych wejściowych**
- **trzeba uzgodnić formę zapisania wyniku**

Algorytm sumowania *n* liczb:

1. zapisz „w pamięci" liczbę 0,
2. odczytaj pierwszą liczbę,
3. wykonaj co następuje *n* razy:
   3.1. dodaj odczytaną liczbę do liczby „w pamięci" i zapisz wynik „w pamięci",
   3.2. odczytaj następną liczbę,
4. jako wynik podaj liczbę „w pamięci".

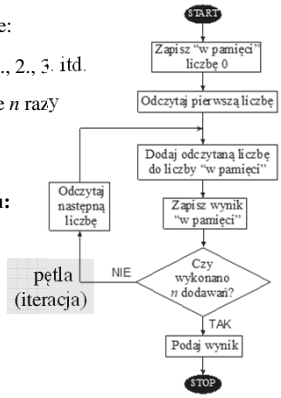
Operacje podstawowe:

- zapisanie liczby „w pamięci"
- odczytanie kolejnej liczby z podanych
- dodanie dwóch liczb do siebie ← operacja „obliczeniowa"
- odczytanie liczby „w pamięci

Użyte instrukcje sterujące:

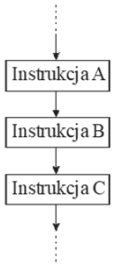
- następstwo operacji: 1., 2., 3. itd.
- powtarzaj co następuje *n* razy

Schemat blokowy algorytmu:

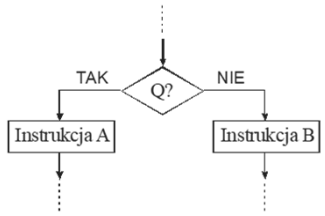


Zestaw podstawowych instrukcji sterujących

➢ **bezpośrednie następstwo** - *„wykonaj A, potem B, potem C, ..."*



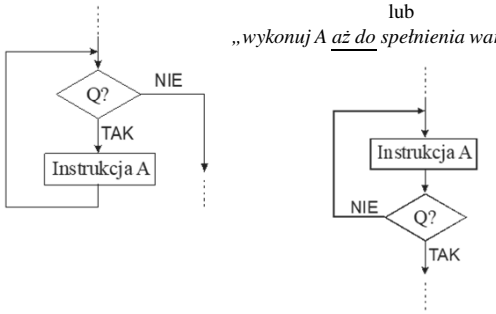
- **wybór warunkowy –**
  *„jeśli warunek Q jest spełniony, to wykonaj A, w przeciwnym przypadku wykonaj **B**”*



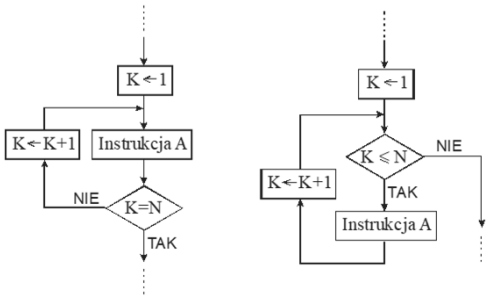
- **iteracja warunkowa –**
  „dopóki warunek Q jest spełniony, wykonuj A”

lub

„wykonuj A aż do spełnienia warunku Q”

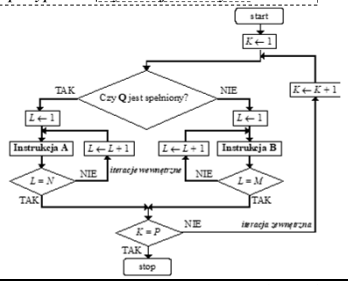


- **pętla (iteracja) ograniczona –**
  „wykonaj A dokładnie N razy”



Instrukcje sterujące mogą być zagnieżdżane jedne w drugich!

1. *wykonaj co następuje **p** razy:*
   1.1. *jeśli warunek **Q** jest spełniony, to wykonaj **n** razy **A**,*
   *w przeciwnym przypadku wykonaj **m** razy **B**”*



Algorytmiczny problem sortowania

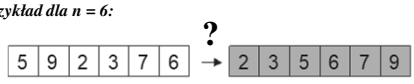
Dopuszczalne dane wejściowe:

lista *n* elementów, które można porównywać parami i określać właściwą kolejność występowania

Oczekiwany wynik:

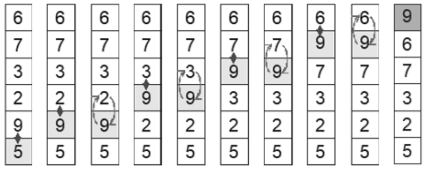
uporządkowana według zadanej kolejności lista tych samych *n* elementów

Na przykład dla n = 6:

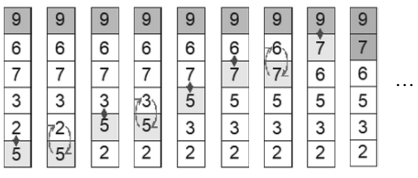


Sortowanie bąbelkowe na przykładzie:

1. przejście przez listę



2. przejście przez listę



Algorytm sortowania bąbelkowego:

1. wykonaj co następuje $n - 1$ razy:
   1.1. wskaż pierwszy element listy,
   1.2. wykonaj co następuje $n - 1$ razy:
      1.2.1. porównaj wskazany element listy z następnym
      1.2.2. jeśli te dwa elementy są w niewłaściwej kolejności, to zamień je miejscami,
      1.2.3. wskaż następny element z listy.