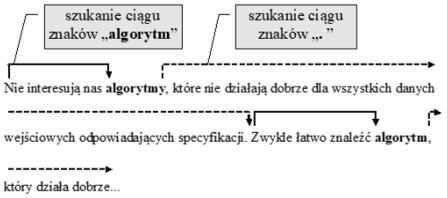
Podprogramy, czyli procedury

Przykład zastosowania:

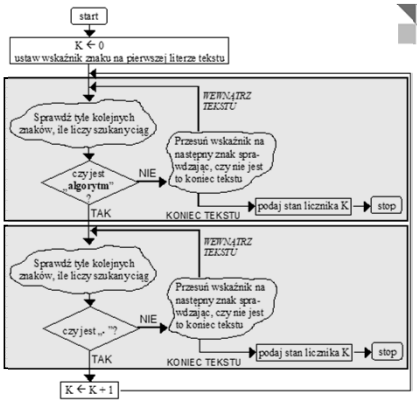
dane wejściowe – poprawnie zredagowany tekst w języku polskim

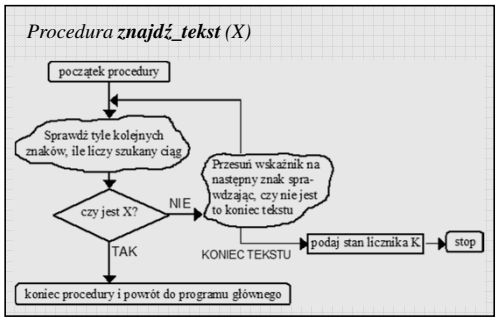
wynik - liczba zdań zawierających słowo „algorytm”

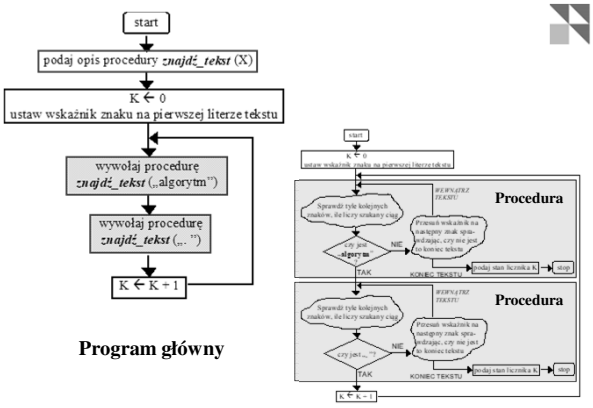


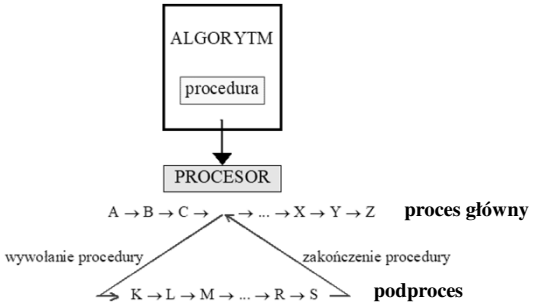
Zarys schematu blokowego:

K – licznik zdań zawierających ciąg znaków „algorytm”









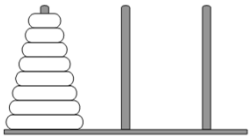
Zalety stosowania procedur przy budowie algorytmów:

- Zwartość opisu algorytmu
- Czytelność struktury programu
- Możliwość zastosowania „analitycznej” metody budowania algorytmu, czyli rozłożenia całego programu na podprogramy opracowywane oddzielnie
- Możliwość zastosowania „syntetycznej” metody budowania algorytmu, czyli składanie programu z wcześniej dopracowanych podprogramów

REKURENCJA

Wywołanie procedury wewnątrz niej samej

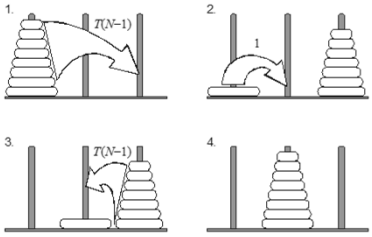
Problem „wież Hanoi" dla N krążków:

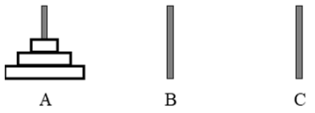


dane wejściowe - trzy kołki i *N* krążków o różnych średnicach

wynik - sekwencja ruchów przenosząca krążki z pierwszego kołka na kołek trzeci zgodnie z zadanymi regułami

Idea rozwiązania rekurencyjnego:





Oznaczmy przez X → Y przeniesienie szczytowego krążka z kołka X na Y

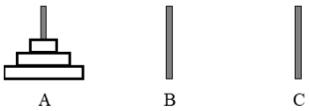
Wynik dla N = 3:

A → B, A → C, B → C, A → B, C → A, C → B, A → B

Rozwiązanie algorytmiczne:

Procedura **przenieś** (*N*) **krążków z** (X) **na** (Y) **używając** (Z)
1. jeśli *N* = 1, to wypisz ruch „X → Y",
2. w przeciwnym razie (tj. jeśli *N* > 1) wykonaj co następuje:
   2.1. Wywołaj procedurę
   przenieś (*N* – 1) **krążków z** (X) **na** (Z) **używając** (Y),
   2.2. Wypisz ruch „X → Y",
   2.1. Wywołaj procedurę
   przenieś (*N* – 1) **krążków z** (Z) **na** (Y) **używając** (X),
3. wróć do poziomu wywołania procedury.

Rozwiązanie dla trzech krążków:



Wywołaj procedurę

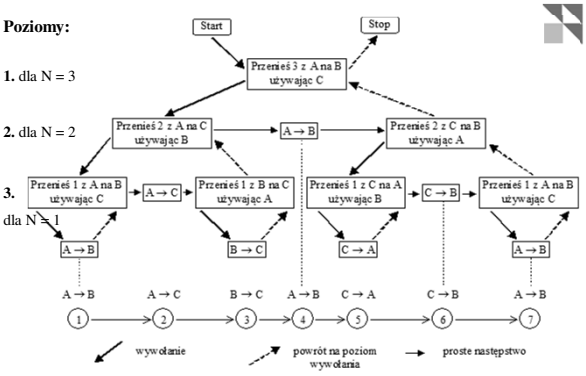
przenieś (3) **krążki z** (A) **na** (B) **używając** (C)

Poziomy:

1. dla N = 3

2. dla N = 2

3. dla N = 1



STRUKTURY DANYCH

Elementy zbioru danych wejściowych, które opisują konkretny przypadek problemu algorytmicznego (zadanie) muszą zostać zapisane w pamięci (operacyjnej) przed rozpoczęciem działania algorytmu

Podstawową formą zapisania pojedynczego elementu zbioru danych, np. ***a***, w pamięci jest utworzenie w niej **zmiennej** (pamięciowej), np. ***X***, i przypisanie tego elementu jako jej wartości: $X \leftarrow a$

Zmienna ma nadaną nazwę, np. ***X*** i określony typ.

Zmienna symbolizuje pewien obszar pamięci, w którym można wielokrotnie zapamiętywać i z którego można odczytywać pojedyncze elementy zbioru danych z zachowaniem zgodności typu elementu i zmiennej.

Przykładowe typy zmiennych (i danych):

- **liczbowy** (całkowity, rzeczywisty),
- **znakowy** (pojedynczy symbol z przyjętego alfabetu, ciąg symboli),
- **wskaźnikowy** (adres, pod którym można znaleźć w pamięci inną zmienną określonego typu)

Odczytanie elementu z pamięci symbolizowane jest przez odwołanie się do wartości zmiennej:

np. $X \leftarrow Y$ oznacza odczytanie z pamięci elementu będącego aktualną wartością zmiennej ***Y*** i przypisanie go zmiennej ***X***;

w obu obszarach pamięci symbolizowanych przez ***X*** i ***Y*** będzie po tej operacji przechowywany ten sam element zbioru danych

Zmienne pamięciowe mogą być łączone w zespoły nazywane (abstrakcyjnymi) **strukturami danych**

Jeśli w trakcie działania algorytmu struktura danych, utworzona przed rozpoczęciem jego działania, nie może być zmieniana w trakcie wykonywania algorytmu ani w zakresie liczby zmiennych, ani sposobu ich powiązania, to nazywamy ją **strukturą statyczną**

W przeciwnym przypadku, jeśli dysponujemy operacjami, za pomocą których możemy modyfikować strukturę (np. usuwać jej fragmenty, dodawać nowe lub zmieniać jej organizację) w trakcie działania algorytmu, to nazywamy ją **strukturą dynamiczną**