laboratorium: zadanie 3 *17 grudnia 2006 r.*

ALGORYTMY I STRUKTURY DANYCH

DRZEWO AVL

Instytut Informatyki Uniwersytetu Wrocławskiego *Paweł Rzechonek*

Zadanie:

Zaimplementuj słownik z operacjami *insert delete* i *search* w postaci drzewa AVL. W drzewach tych mają być przechowywane liczby całkowite. Klucze w pamiętanym zbiorze dynamicznym nie mogą się powtarzać. Program powinien interpretować i realizować następujące rozkazy:

- Sprawdzenie czy element jest w drzewie (przykładowo 17):
  `search 17`
  W odpowiedzi procedura zwraca wartość 1 gdy element został znaleziony, albo 0 gdy nie było go w drzewie.
- Wstawienie elementu do drzewa (przykładowo 33):
  `insert 33`
  W odpowiedzi procedura zwraca wartość 1 gdy element został wstawiony, albo 0 gdy elementu nie wstawiono (w drzewie był już taki element).
- Usunięcie elementu z drzewa (przykładowo 65):
  `delete 65`
  W odpowiedzi procedura zwraca wartość 1, gdy element został usunięty, albo 0 gdy elementu nie usunięto (w drzewie takiego elementu nie było).
- Sprawdzenie liczby elementów w drzewie:
  `size`
  W odpowiedzi procedura zwraca liczbę wszystkich elementów w drzewie.
- Wypisanie wszystkich elementów zapamiętanych w drzewie w porządku *inorder*, *preorder* lub *postorder*:
  `print-inorder`
  `print-preorder`
  `print-postorder`
  W odpowiedzi procedura przegląda wszystkie węzły w drzewie w odpowiedniej kolejności i wypisuje zapamiętane w nich wartości w jednej linii (po wypisaniu każdej wartości jest drukowana pojedyncza spacja).

Dane:

W pierwszym wierszu z danymi jest podana liczba n ($1 \leq n \leq 1000000$) oznaczająca liczbę instrukcji wykonywanych na początkowo pustym drzewie AVL, a w kolejnych n wierszach zapisane są rozkazy zgodnie ze składnią opisaną wcześniej.

Wyniki:

W wyniku należy wypisać w n wierszach wszystkie n odpowiedzi generowane przez program (każda odpowiedź w osobnym wierszu).

Uwaga:

Aby uniknąć niejednoznaczności w wynikach przy usuwaniu elementu (operacja *delete*) zakładamy że:

- jeśli usuwana wartość leży w liściu, to odcinamy ten liść;
- jeśli usuwana wartość leży w węźle wewnętrznym, który posiada jednego syna, to przenosimy element z syna do tego węzła i odcinamy syna;
- jeśli usuwana wartość leży w węźle wewnętrznym, który posiada dwóch synów, to z lewego poddrzewa usuwamy element maksymalny i zastępujemy nim dotychczasową wartość w tym węźle.

Wstawienie nowego elementu do drzewa (operacja *insert*) jest jednoznaczne i polega zawsze na utworzeniu nowego liścia na końcu ścieżki poszukiwań.

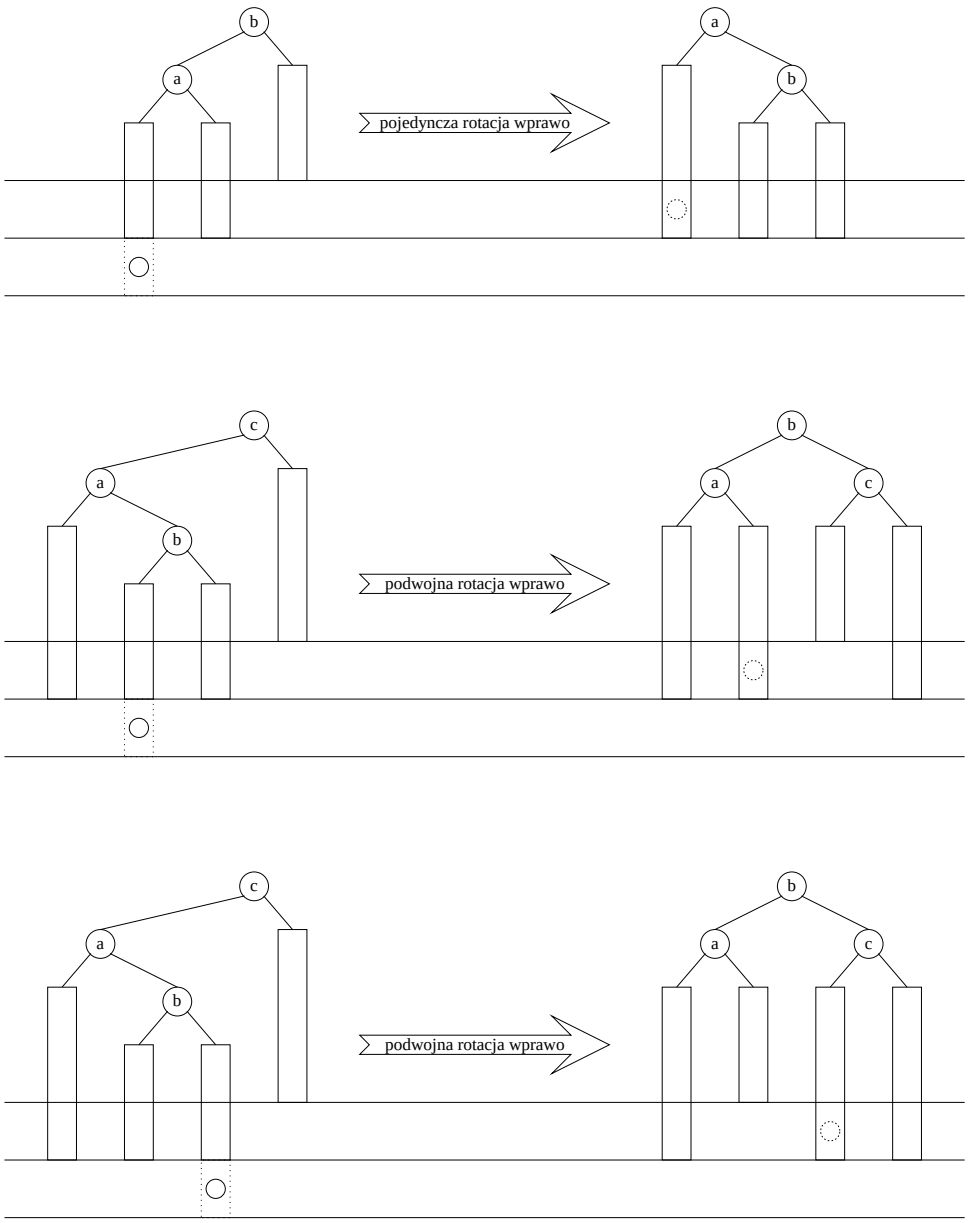
Przykład:

Przykładowe dane wejściowe mogą mieć postać:

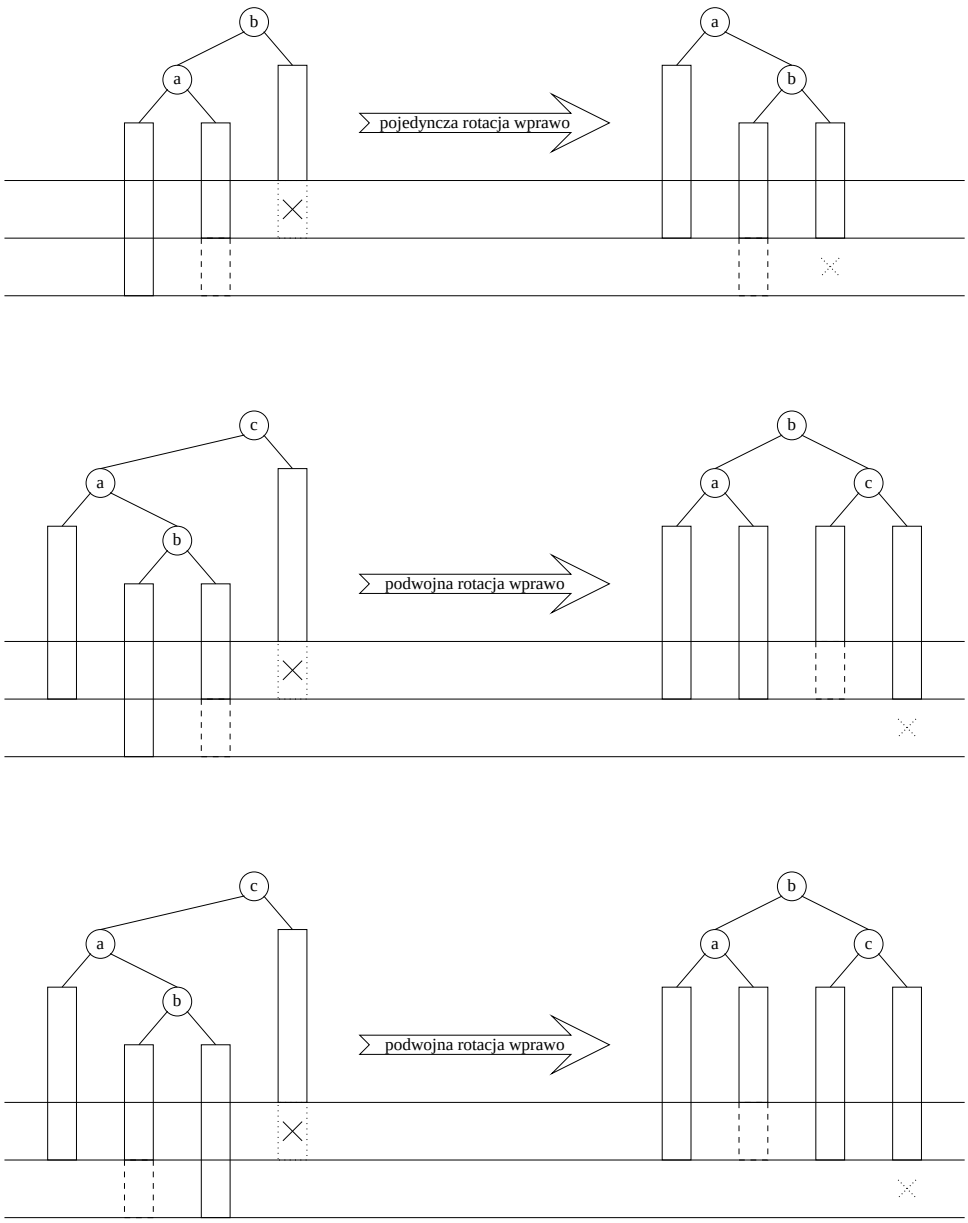
```
insert 5
search 4
insert 3
delete 2
print-preorder
insert 5
insert 2
print-postorder
insert 7
delete 2
print-inorder
search 5
size
```

Wówczas na wyjścu powinien pojawić się następujący wynik:

```
1
0
1
0
5 3
0
1
2 5 3
1
1
3 5 7
1
3
```



Rysunek 1: Na rysunkach przedstawiono wszystkie sytuacje, które wymagają wykonania rotacji (pojedynczej lub podwójnej) po wstawieniu elementu do drzewa AVL.



Rysunek 2: Na rysunkach przedstawiono wszystkie sytuacje, które wymagają wykonania rotacji (pojedynczej lub podwójnej) po usunięci elementu z drzewa AVL.