Instytut Informatyki UWr

Wstęp do informatyki

Lista 11

Zadania na tej liście dotyczą drzew binarnych tworzonych z użyciem następujących definicji typów:

Język C:

```
typedef struct node *pnode;
typedef struct node{
        int val;
        pnode left;
        pnode right;} snode;
```

Język Python:

```
class TreeItem:
   def __init__(self,value):
      self.val = value
      self.left = None
      self.right = None
```

W rozwiązaniach zadań w języku C można korzystać z funkcji

```
pnode utworz(int wart)
```

tworzącej drzewo z jednym węzłem z wartością `val` równą `wart` (p. notatka do wykładu).

1. [1] Do drzewa BST (na początku pustego) wstawiane są elementy 1, 2, 3, 4, 5, 6, 7. Podaj kolejność wstawiania elementów, przy której drzewo będzie miało największą (najmniejszą) możliwą wysokość. Odpowiedź uzasadnij i uogólnij na przypadek ciągu liczb 1,2, …,$2^k - 1$ dla dowolnego naturalnego $k>1$.
2. [1] Napisz funkcję, która dla parametru `t` wskazującego na korzeń drzewa binarnego zwraca jako wartość liczbę elementów w drzewie o korzeniu `t`.
3. [1] Napisz funkcję, która dla parametru `t` zwraca jako wartość *wysokość* drzewa o korzeniu `t`
4. [0.5] Napisz funkcję, która dla parametru `t` opisującego drzewo BST wypisuje (w porządku niemalejącym) wszystkie elementy dodatnie znajdujące się w drzewie o korzeniu `t`.
5. [2] Napisz funkcję, która dla danego drzewa binarnego sprawdza czy jest ono drzewem BST.
6. [1] Napisz funkcję, która łączy dwa drzewa BST w jedno drzewo przy założeniu, że wszystkie wartości w pierwszym drzewie są mniejsze od najmniejszej wartości w drugim drzewie. Czas działania Twojej funkcji powinien być O(h), gdzie h to wysokość pierwszego drzewa.
7. [0.5] Opisz efekt działania poniższych funkcji dla dowolnego drzewa binarnego `t` i dla drzewa BST:

```
wypisz(pnode t)
{ if (t!=NULL){
      printf("%d\n", t->val);
      wypisz(t->left);
      wypisz(t->right);
  }
}
```

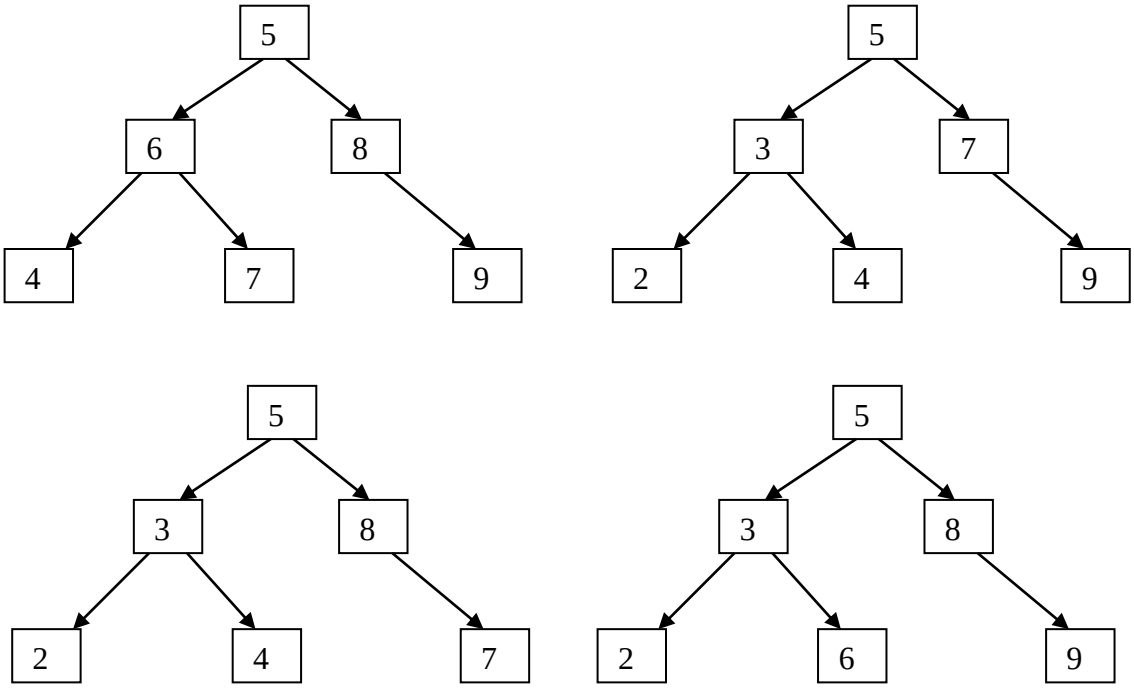
```
def wypisz(t):
  if (t!=None)
      print t.val
      wypisz(t.left)
      wypisz(t.right)
```

```
wypisz(pnode t)
{ if (t!=NULL){
      wypisz(t->left);
      wypisz(t->right);
      printf("%d\n", t->val);
  }
}
```

```
def wypisz(t):
  if (t!=None)
      wypisz(t.left)
      wypisz(t.right)
      print t.val
```

Zadania dodatkowe, nieobowiązkowe (nie wliczają się do puli punktów do zdobycia na ćwiczeniach, punktacja została podana tylko jako informacja o trudności zadań wg wykładowcy)

8. [0] Które z poniższych drzew są drzewami BST:



9. [2] Napisz funkcję, która dla parametru `t` opisującego drzewo BST wypisuje (w porządku rosnącym) wszystkie elementy dodatnie znajdujące się w drzewie o korzeniu `t`. Czas działania Twojej funkcji powinien być O(h+m), gdzie h to wysokość drzewa `t`, a m to liczba elementów dodatnich w drzewie.
10. [1] Napisz funkcję wyszukującą element o podanym kluczu w drzewie BST bez użycia rekurencji.
11. [1] Napisz funkcję wstawiającą element o podanym kluczu do drzewa BST bez użycia rekurencji.
12. [1] Napisz funkcję usuwającą element o podanym kluczu z drzewa BST bez użycia rekurencji.
13. [1] Dla funkcji wyszukującej element w drzewie BST utwórz wersję, w której operacja ta będzie przyspieszona przez użycie wartownika.
    Wskazówka: wszystkie wskaźniki NULL zastąp wskaźnikami do jednego węzła, w którym umieszczasz szukany klucz przed przystąpieniem do wyszukiwania.