

Algoryty i Struktury Danych 2012

Lista 4

1. Do pustego drzewa czerwono-czarnego wstaw kolejno litery swojego imienia i nazwiska. Następnie usuń je w tej samej kolejności w jakiej były wstawiane.
2. Do pustego B-drzewa wstaw kolejno litery swojego imienia i nazwiska. Następnie usuń je w tej samej kolejności w jakiej były wstawiane.
3. Jak jest minimalna a jaka maksymalna liczba kluczy w drzewie czerwono-czarnym o czarnej wysokości h_b .
4. Jak jest minimalna, a jaka maksymalna liczba kluczy w B-drzewie mającym h poziomów, przy ustalonej wartości parametru t (patrz Cormen).
5. Zastosuj twierdzenie o rekursji uniwersalnej do każdego z poznanych algorytmów, do jakiego da się ono zastosować.
6. Udowodnij, że każda operacja na kopcu złącznym (dwumianowym), zawierającym n kluczy, ma złożoność $O(\log n)$.
7. Napisz program, który znajdzie i wyświetli w jakiej kolejności należy poruszać się skoczkiem po szachownicy, by każde pole odwiedzić dokładnie raz. Pole startowe oraz rozmiar szachownicy powinny być parametrem podawanym z linii komend lub w trybie interakcyjnym.
Zadanie teoretyczne: Dla jakich rozmiarów szachownicy nie istnieje rozwiązanie?
8. Napisz funkcję `void perm(int t[], int n)`, która wypisze wszystkie permutacje elementów podanej tablicy.
9. Napisz funkcję `bool przejście(bool t[8][8])`, które sprawdzi, czy można przejść z pola `t[0][0]` do `t[7][7]` poruszając się tylko w poziomie i pionie po polach zerowych.
Wskazówka: (a) możesz użyć klasy `UnionFind`. (b) możesz nie używać klasy `UnionFind`.
10. Zmodyfikuj klasę `UnionFind`, dodając tablicę `rank`, która zawiera oszacowanie maksymalnej wysokości drzewa i -tego drzewa. Przy operacji `union` należy podpinąć drzewo niższe pod wyższe. Jeśli są równej wysokości, to podpinamy dowolnie, ale `rank` korzenia należy zwiększyć o 1.
http://en.wikipedia.org/wiki/Disjoint-set_data_structure
Sprawdź czy modyfikacja powoduje poprawę efektywności dla dużych n .