ZADAĆA 9

Objektno orijentirano programiranje

Upute:

Zadaću je potrebno predati do 25. svibnja u 15:00 na Teamsu. Diskutiranje zadaće u okviru "study group" je dozvoljeno, ali predana zadaća mora biti samostalno riješena. Studenti koji predaju zadaću obavezni su prisustvovati vježbama, u suprotnome zadaća neće biti bodovana. Ako se kod ne prevodi (neovisno o tipu compiler-a), ili se događa greška prilikom izvršavanja koda, zadaća neće biti bodovana.

Zadatak 2. (100 bodova) Binary Search Tree

Definirajte klasu `BST` koja pohranjuje čvorove klase `Node` čija je deklaracija:

```
template <typename K, typename V>
struct Node {
    K key;
    V value;
    Node* parent, * left, * right;

    void print() const;
};
```

Deklaracija klase `BST` je:

```
template <typename K, typename V>
struct BST {
    Node<K, V>* root;

    BST();
    // copy konstruktor (pripaziti na shallow-copy)
    BST(const BST& bst);
    ~BST();

    // pretrazuj podstablo s korijenom x dok ne pronadeš čvor
    // vrijednoscu key (u suprotnom vrati nullptr)
    Node<K, V>* search(Node<K, V>* x, K key);

    // vrati pokazivač na čvor koji ima minimalnu vrijednost
    // kljuca u podstablu čiji je korijen x
```

```
    Node<K, V>* minimum(Node<K, V>* x);

    // vrati pokazivač na čvor koji ima maksimalnu vrijednost
    // kljuca u podstablu čiji je korijen x
    Node<K, V>* maximum(Node<K, V>* x);

    // vrati sljedbenika čvora x po vrijednosti key unutar stabla
    Node<K, V>* successor(Node<K, V>* x);

    // vrati prethodnika čvora x po vrijednosti key unutar stabla
    Node<K, V>* predecessor(Node<K, V>* x);

    // unesi novi čvor brinuvši se o definiciji binary search tree-a
    void insert(const K& key, const V& value);


    // zamijeni podstabla s korijenima u i v
    void transplant(Node<K, V>* u, Node<K, V>* v);

    // obriši čvor x brinuvši se o definiciji binary search tree-a
    void remove(Node<K, V>* x);

    // napravi inorder obilazak, vrijednosti redom pohrani
    // u vektor nodes
    void inorderWalk(Node<K, V>* x, vector<Node<K, V>*>& nodes) const;

    // copy pridruživanje (pripaziti na shallow-copy)
    BST& operator=(const BST& bst);

    void print() const;
};
```

Instancirajte objekt klase BST, dodajte mu nekoliko vrijednosti, ispišite ga, uklonite korijen te ponovno ispišite. Ispis treba biti:

```
----------------------------------------------------
Binary search tree with root at: 0x600062280
Nodes:
Node at: 0x600074720; parent at:0x600062280; left at: 0; right at: 0x600074780
Key: 15; value: Korisnik3
Node at: 0x600074780; parent at:0x600074720; left at: 0; right at: 0
Key: 17; value: Korisnik4
Node at: 0x600062280; parent at:0; left at: 0x600074720; right at: 0x6000622e0
Key: 20; value: Korisnik1
Node at: 0x6000622e0; parent at:0x600062280; left at: 0; right at: 0
```

Key: 25; value: Korisnik2

Binary search tree with root at: 0x6000622e0
Nodes:
Node at: 0x600074720; parent at:0x6000622e0; left at: 0; right at: 0x600074780
Key: 15; value: Korisnik3
Node at: 0x600074780; parent at:0x600074720; left at: 0; right at: 0
Key: 17; value: Korisnik4
Node at: 0x6000622e0; parent at:0; left at: 0x600074720; right at: 0
Key: 25; value: Korisnik2
