Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής & Υπολογιστών
https://courses.softlab.ntua.gr/progtech/

Προγραμματιστικές Τεχνικές

Προσοχή!

Λύνετε αυτή την άσκηση μόνο αν το υπόλοιπο της διαίρεσης του αριθμού μητρώου σας δια του 4 είναι ίσο με 1.

```
if (AM % 4 != 1)
  break;
```

Το προγραμματιστικό πρόβλημα που περιγράφεται παρακάτω αποτελεί το θέμα της τρίτης και τελευταίας εργαστηριακής εξέτασης. Υποβάλλετε εντός τριών (3) ημερών τις λύσεις σας στο γνωστό αυτόματο σύστημα ελέγχου — αν δεν έχετε κωδικό (`pi20bXYZ` ή `pt21aXYZ`), φροντίστε να αποκτήσετε.

Άσκηση Γ Δέντρα AVL με διπλότυπα κλειδιά

Προθεσμία υποβολής στον grader: 4/6/2021

Στο σύνδεσμο `https://git.softlab.ntua.gr/pub/avl-tree` μπορείτε να βρείτε μία σχετικά απλή υλοποίηση δέντρων AVL. Στο αρχείο `avltree.hpp` ορίζεται ένα class template για τα δέντρα AVL και υλοποιούνται οι εξής βασικές πράξεις: κατασκευή και καταστροφή δέντρων, προσθήκη, αναζήτηση και αφαίρεση, ενδοδιατεταγμένη (in-order) διάσχιση μέσω iterator.

Ο σκοπός αυτής της άσκησης είναι να τροποποιήσετε το αρχείο `avltree.hpp` και να προσθέσετε στην κλάση `avltree<T>` τη δυνατότητα να προστίθεται σε ένα δέντρο το ίδιο κλειδί περισσότερες από μία φορές. Συγκεκριμένα, ζητείται να προσθέσετε μία μέθοδο με την παρακάτω επικεφαλίδα:

```
1 public:
2   int count(const T &x);
```

η οποία θα επιστρέφει το πλήθος των εμφανίσεων του κλειδιού x στο δέντρο. Αν το x δεν εμφανίζεται στο δέντρο, η μέθοδος `count` πρέπει να επιστρέφει μηδέν. Επίσης, η μέθοδος `size` θα πρέπει να επιστρέφει το συνολικό πλήθος των στοιχείων του δέντρου, συμπεριλαμβανομένων των διπλοτύπων, και οι iterators θα πρέπει να "σταματούν" στα διπλότυπα στοιχεία πολλές φορές, καθώς διασχίζουν το δέντρο.

Προφανώς, μπορείτε να ορίσετε όσες επιπλέον (private) βοηθητικές μεθόδους χρειαστείτε, δεν πρέπει όμως να τροποποιήσετε τα υπάρχοντα περιεχόμενα του αρχείου, εκτός όσων προτείνονται παρακάτω.

Για να υλοποιήσετε την ύπαρξη διπλοτύπων, μπορείτε να προσθέσετε στους κόμβους του δέντρου ένα ακόμα πεδίο με τιμή ακέραιο αριθμό: την πολλαπλότητα του κλειδιού. Θα χρειαστεί να τροποποιήσετε κατάλληλα τις μεθόδους `insert` και `remove`, έτσι ώστε να αυξάνονται και να μειώνονται κατάλληλα οι πολλαπλότητες των κόμβων. Για να υλοποιήσετε σωστά τη διάσχιση μέσω iterators, μπορείτε να προσθέσετε στην κλάση `TreeIteratorImpl` έναν ακέραιο αριθμό που να παριστάνει πόσα "στιγμιότυπα" του κόμβου στον οποίο δείχνει το `ptr` έχουμε ήδη διασχίσει.

Μπορείτε να δοκιμάσετε την υλοποίησή σας με προγράμματα όπως το `example.cpp` που δίνεται στο repository της υλοποίησης των δέντρων AVL και με τα test cases που θα βρείτε εκεί. Μπορείτε να τροποποιήσετε την εντολή '`l`' έτσι ώστε να τυπώνει επίσης το πλήθος των κόμβων με τιμή ίση με το δοθέν κλειδί, δηλαδή το αποτέλεσμα της `count`. Ίσως χρειαστεί να φτιάξετε και δικά σας test cases που να εισάγουν διπλότυπα κλειδιά.

Προσοχή!

- Το αρχείο που θα ανεβάσετε στον grader θα πρέπει να είναι το τροποποιημένο `avltree.hpp`. Μην πειράξετε οτιδήποτε εκτός από αυτά που ζητάει η άσκηση!
- Φροντίστε να βάλετε στην πρώτη γραμμή ένα **σχόλιο με το ονοματεπώνυμο και τον αριθμό μητρώου σας!**