Κ08 Δομές Δεδομένων και Τεχνικές Προγραμματισμού

Διδάσκων: Μανόλης Κουμπαράκης

Υπεύθυνος Εργαστηρίου: Μίλτος Κυριακάκος

Εργαστήριο 3: Δυαδικά Δένδρα Αναζήτησης

Στο εργαστήριο αυτό θα χρησιμοποιήσουμε τον κώδικα που παρουσιάστηκε στην ένατη ενότητα διαλέξεων (Δυαδικά Δέντρα Αναζήτησης) και βρίσκεται στην ιστοσελίδα του μαθήματος: http://cgi.di.uoa.gr/~k08/code.html . Ο κώδικας υλοποιεί διάφορες λειτουργίες των δυαδικών δένδρων αναζήτησης που συζητήσαμε στην τάξη. Μελετήστε τον κώδικα προσεκτικά και κατανοήστε τη λειτουργία του. Μεταγλωττίστε τον και εκτελέστε τον. Μετά κάνετε τις παρακάτω ασκήσεις.

1. Στην άσκηση αυτή θα θυμηθούμε την έννοια του **δείκτη σε μία συνάρτηση** από το πρώτο μάθημα προγραμματισμού. Στη C, όπως έχουμε δείκτες προς δεδομένα διαφόρων τύπων, έτσι μπορούμε να έχουμε και δείκτες σε συναρτήσεις. Μια που ο μεταγλωττιστής της C δεσμεύει μνήμη για να αποθηκεύσει τον κώδικα μιας συνάρτησης, το να έχουμε ένα δείκτη προς αυτές τις θέσεις μνήμης είναι λογικό. Η γενική μορφή της δήλωσης ενός δείκτη σε μια συνάρτηση έχει την εξής μορφή:

   ```
   return_type (*pointer_name)(parameter_type1
     parameter_name1, …, parameter_typeN parameter_nameN).
   ```

   Για να είναι σωστή η δήλωση ενός δείκτη προς μία συνάρτηση, θα πρέπει να ταιριάζει με τον τύπο επιστροφής της συνάρτησης και τη λίστα των τυπικών παραμέτρων της. Παρακάτω δίνεται ένα μικρό πρόγραμμα που παρουσιάζει την χρήση δείκτη σε συνάρτηση. Μελετήστε το πρόγραμμα προσεκτικά. Μεταγλωττίστε το και εκτελέστε το. Αφού κατανοήσετε τη έννοια του δείκτη σε συνάρτηση, μπορείτε να συνεχίσετε με την άσκηση 2.

```
/* Το παράδειγμα αυτό προέρχεται από το βιβλίο «C: Από τη
θεωρία στην εφαρμογή» των Γ. Σ. Τσελίκη και Ν.Δ. Τσελίκα, 2η
έκδοση, σελ. 252. */

#include <stdio.h>

void test(int a);
```

```
int main()
{

   void (*ptr)(int a);
      /* Η μεταβλητή ptr δηλώνεται ως δείκτης σε μία
συνάρτηση, η οποία δέχεται μια ακέραια παράμετρο και δεν
επιστρέφει τίποτα. */

   int i=10;

   ptr=test;
      /* Τώρα ο δείκτης ptr δείχνει στη διεύθυνση της
συνάρτησης test. Όπως το όνομα ενός πίνακα μπορεί να
χρησιμοποιηθεί σαν δείκτης, έτσι και το όνομα μιας
συνάρτησης μπορεί να χρησιμοποιηθεί σαν δείκτης στη
διεύθυνσή της.*/

   ptr(i);
    /* Η κλήση αυτή είναι ισοδύναμη με την (*ptr)(i); που
τονίζει περισσότερο το γεγονός ότι ο ptr είναι δείκτης προς
μία συνάρτηση. */

   return 0;
}

void test(int a)
{
   printf("Val = %d\n", 2*a);
}
```

2. Να γράψετε τρεις συναρτήσεις διάσχισης δυαδικού δένδρου αναζήτησης `STpreorder`, `STinorder` και `STpostorder` τις οποίες να προσθέσετε στο αρχείο `bst-implementation.c`. Οι συναρτήσεις αυτές διασχίζουν το δένδρο αναζήτησης με τον τρόπο που λέει το όνομα τους. Οι συναρτήσεις αυτές πρέπει να παίρνουν μια τυπική παράμετρο `void (*Visit)(Item x)` που είναι ένας δείκτης σε μια συνάρτηση η οποία παίρνει μια τυπική παράμετρο τύπου `Item`. Κατά την κλήση των τριών συναρτήσεων μπορούμε να δίνουμε σαν όρισμα το όνομα μιας συνάρτησης η οποία θα εκτελείται αντί της `Visit` (π.χ., η συνάρτηση `ITEMshow` που δίνεται στο αρχείο `item-implementation.c`). Καλέστε τις

νέες συναρτήσεις διάσχισης από το κύριο πρόγραμμα (συνάρτηση `main`) ώστε να ελέγξετε τη λειτουργία τους.

3. Θεωρείστε την αναδρομική συνάρτηση `insertR` που καλείται από την συνάρτηση εισαγωγής ενός στοιχείου σε ένα δυαδικό δένδρο αναζήτησης `STinsert` (αρχείο `bst-implementation.c`). Γράψτε μια νέα συνάρτηση STinsert1 που να χρησιμοποιεί επανάληψη αντί για αναδρομή. Να ελέγξετε την λειτουργία της με κατάλληλες κλήσεις από το κύριο πρόγραμμα.

4. Γράψτε δύο συναρτήσεις `STmin()` και `STmax()` οι οποίες θα επιστρέφουν το στοιχείο με το μικρότερο και το μεγαλύτερο κλειδί αντίστοιχα σε ένα δυαδικό δένδρο αναζήτησης. Να ελέγξετε την λειτουργία των συναρτήσεων αυτών με κατάλληλες κλήσεις από το κύριο πρόγραμμα.

5. Θεωρήστε τις συναρτήσεις `STinsert` και `insertT` οι οποίες εισάγουν ένα νέο κόμβο στη ρίζα ενός δυαδικού δένδρου αναζήτησης. Οι συναρτήσεις αυτές, στην τωρινή τους μορφή, δεν ενημερώνουν το πεδίο `N` κάθε κόμβου το οποίο μας δίνει πόσα στοιχεία περιέχονται στο υποδένδρο με ρίζα αυτό τον κόμβο. Γράψτε μια νέα έκδοση των συναρτήσεων αυτών οι οποίες ενημερώνουν κατάλληλα το πεδίο `N`. Να ελέγξετε την λειτουργία των συναρτήσεων αυτών με κατάλληλες κλήσεις από το κύριο πρόγραμμα. Υπόδειξη: Θα πρέπει να αλλάξουν και οι συναρτήσεις περιστροφής `rotR` και `rotL` οι οποίες χρησιμοποιούνται από την `insertT`.

6. Να μελετήσετε το αρχείο `test-bst.c`, το οποίο εξετάζει τη λειτουργικότητα της αρχικής έκδοσης του `bst-implementation.c` μέσω της χρήσης unit tests. Να εμπλουτίσετε το αρχείο αυτό, προσθέτοντας tests τα οποία θα ελέγχουν την λειτουργικότητα των συναρτήσεων που έχετε προσθέσει στα ερωτήματα 2-5."