ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

1η ΣΕΙΡΑ ΑΣΚΗΣΕΩΝ ΣΤΗΝ ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΥΠΟΛΟΓΙΣΤΩΝ

Ακ. έτος 2015-2016, 5ο Εξάμηνο, Σχολή ΗΜ&ΜΥ

(τμήμα Μ – Ω)

Ημερομηνία Παράδοσης: 15/11/2015

Απορίες στο: ca2015-2016-tmima2@cslab.ece.ntua.gr

ΑΣΚΗΣΗ 1η

Δίνεται η συνάρτηση `foo` σε C καθώς και ο ημιτελής μεταφρασμένος της κώδικας σε MIPS assembly. Σημειώνεται πως ο τύπος `int` της C έχει μέγεθος 4 bytes (1 word για την αρχιτεκτονική MIPS). Όσον αφορά τους καταχωρητές που χρησιμοποιούνται, ακολουθούνται οι κανόνες που διδάχθηκαν στο μάθημα (`$a0`-`$a3` για ορίσματα, `$v0`-`$v1` για τιμές αποτελέσματος, κτλ.), ενώ έχετε υπόψη πως το instruction set του MIPS περιέχει εντολές μόνο για πρόσθεση και αφαίρεση. Τέλος, θεωρήστε θετικές μεταβλητές και στοιχεία πίνακα.

```
int foo(int *x, int n) {
   int s = 0, i = 0;
   while (i < n) {
      s += *(x + i);
      i++;
   }
   return s / n;
}
```

```
foo:
   add $t0, _____, _____   # s = 0
   add $t1, _____, _____   # i = 0
L1:
   slt $t4, _____, $a1     # $t4 = 1 if i < n
   beq _____, $zero, L2    # if $t4 = 0 jump L2
   sll $t9, $t1, _____     # 4 * i
   add $t2, _____, $a0     # x + 4 * i
   _____ $t3, 0(_____)     # load *(x + 4 * i)
   add $t0, $t0, _____     # s += *(x + 4 * i)
   addi $t1, $t1, _____    # i++
   j _____
L2:
   add $v0, $zero, _____
L3:
   slt _____, $t0, $a1
   bne $t4, $zero, _____
   _____ $t0, $t0, $a1
   _____ $v0, $v0, _____
```

	j _____ EXIT: _____ _____ # return

A) Συμπληρώστε κατάλληλα τα κενά.

Β) Ποια λειτουργία επιτελεί η `foo`;

ΑΣΚΗΣΗ 2η

Δίνονται οι δύο ακόλουθες συναρτήσεις σε C που υλοποιούν τον αλγόριθμο της ταξινόμησης φυσαλίδας (bubblesort). Ζητούνται οι αντίστοιχες υλοποιήσεις σε MIPS assembly.

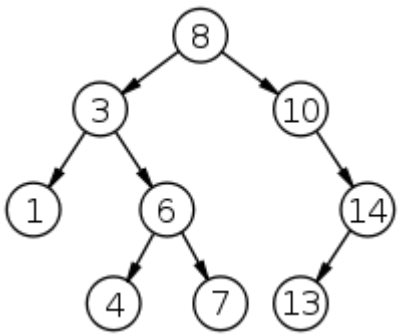
```
void swap(int *x, int *y) {
   int tmp = *x;
   *x = *y;
   *y = tmp;
}

void bubblesort(int array[], int n) {
   int i, j;
   for (i = 0 ; i < n - 1); i++) {
      for (j = 0; j < n - i - 1; j++) {
         if (array[j] > array[j+1]) {
            swap(array + j, array + j + 1);
         }
      }
   }
}
```

ΑΣΚΗΣΗ 3η

Δυαδικό Δέντρο Αναζήτησης (Binary Search Tree - BST) ονομάζεται το δυαδικό δένδρο με διακριτά κλειδιά (τιμές) στους κόμβους και τις εξής ιδιότητες:

- Τα κλειδιά, αν υπάρχουν, στο αριστερό υποδέντρο της ρίζας είναι μικρότερα από το κλειδί της ρίζας.
- Τα κλειδιά, αν υπάρχουν, στο δεξιό υποδέντρο της ρίζας είναι μεγαλύτερα από το κλειδί της ρίζας
- Το αριστερό και το δεξιό υποδέντρο είναι επίσης ΔΔΑ.

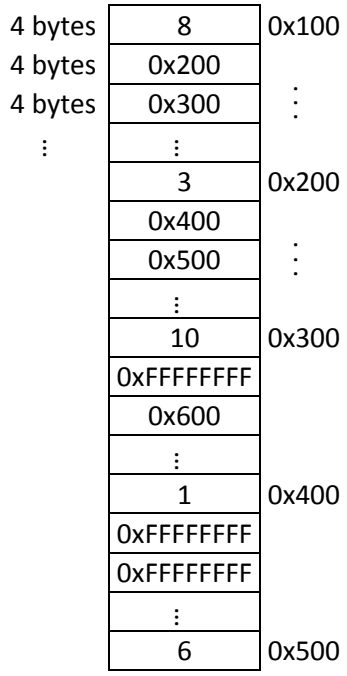
Ένα παράδειγμα ΔΔΑ απεικονίζεται στην κάτωθι εικόνα:

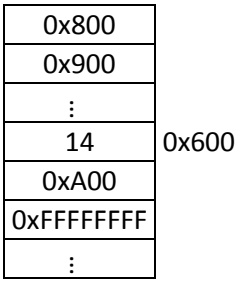


Οι παραπάνω ιδιότητες του ΔΔΑ παρέχουν μια διάταξη μεταξύ των κλειδιών έτσι ώστε βασικές ενέργειες πάνω τους να γίνονται γρήγορα. Μία από αυτές είναι και η αναζήτηση. Ακολουθεί μια πιθανή υλοποίησή της σε C:

```
struct node {
   int key;
   struct node* left;
   struct node* right;
}

// C function to search a given key in a given BST
struct node* search(struct node* root, int key) {
    // Base Cases: root is null or key is present at root
    if (root == NULL || root->key == key)
       return root;
    // Key is greater than root's key
    if (root->key < key)
       return search(root->right, key);
    // Key is smaller than root's key
    return search(root->left, key);
}
```

Κάθε δομή `node` της C μπορεί να θεωρηθεί ότι καταλαμβάνει 12 συνεχόμενες θέσεις μνήμης. Πιο συγκεκριμένα, 4 bytes για την ακέραια τιμή του κλειδιού, 4 bytes με τη διεύθυνση του αριστερού παιδιού-κόμβου και 4 bytes με τη διεύθυνση του δεξιού παιδιού-κόμβου. Για διευκόλυνση θεωρήστε πως κόμβοι-φύλλα, εκτός από το κλειδί τους, περιέχουν την τιμή `0xFFFFFFFF` (-1) στη θέση των διευθύνσεων των παιδιών τους (κωδικοποιείται με τον τρόπο αυτό η τιμή `NULL` της C). Με βάση τα παραπάνω, μέρος του ΔΔΑ της εικόνας αποθηκεύεται στη μνήμη όπως φαίνεται στη συνέχεια (οι διευθύνσεις επιλέχθηκαν τυχαία):





Ζητείται η συνάρτηση `bstSearch` σε MIPS assembly που δέχεται δύο ορίσματα – στον καταχωρητή `$a0` παρέχεται η διεύθυνση του πρώτου byte της ρίζας του δέντρου και στον καταχωρητή `$a1` η τιμή του κλειδιού προς αναζήτηση – και επιστρέφει στον καταχωρητή `$v0` τη διεύθυνση του πρώτου byte του κόμβου που περιέχει την τιμή που αναζητείται. Αν το δέντρο είναι άδειο ή η τιμή δεν υπάρχει σε αυτό, να επιστρέφεται η τιμή `0xFFFFFFFF`.

* * *

Σημείωση: Για τη διευκόλυνσή σας, προτρέπουμε να χρησιμοποιήσετε το προσομοιωτή Qtspim (http://spimsimulator.sourceforge.net/).

Παραδοτέο της άσκησης θα είναι ένα ηλεκτρονικό κείμενο (κατά προτίμηση pdf, για λόγους συμβατότητας) που θα περιέχει τις απαντήσεις των τριών ασκήσεων. Το έγγραφο πρέπει να φέρει στην αρχή του τα στοιχεία σας (όνομα, επώνυμο και αριθμό μητρώου).

Οι κώδικες που θα παραδοθούν οφείλουν να είναι σε ευανάγνωστη μορφή και να φέρουν επαρκή σχόλια.