Οργάνωση και Σχεδίαση Υπολογιστών (ECE 219)
Χειμερινό Εξάμηνο 2021-2022

Εργαστήριο 3

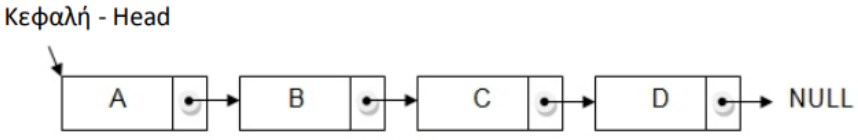
Στόχοι του εργαστηρίου

- Δομές δεδομένων
- Δυναμική δέσμευση μνήμης
- Αναδρομή

Δομές Δεδομένων

Μία δομή δεδομένων είναι μία συλλογή δεδομένων με κάποιες ιδιότητες η οποία προσφέρει εύκολη πρόσβαση στα δεδομένα αυτά. Έχουμε δει ως τώρα μία στατική δομή δεδομένων, τον πίνακα. Τι κάνουμε όμως στην περίπτωση που θέλουμε να αποθηκεύσουμε ένα συνεχώς μεταβαλλόμενο αριθμό από δεδομένα;
Είναι δομές δεδομένων που μπορούν να αυξομειώσουν το μέγεθός τους, δηλαδή να αναπτύσσονται και να συρρικνώνονται στο χρόνο εκτέλεσης. Σε αυτό το εργαστήριο θα δούμε μία σημαντική δομή δεδομένων: την διασυνδεδεμένη λίστα.

Μία **διασυνδεδεμένη λίστα** είναι μια γραμμική συλλογή δομών, που ονομάζονται κόμβοι, και συνδέονται με δείκτες. Μία συνδεδεμένη λίστα προσπελαύνεται μέσω ενός δείκτη στον πρώτο κόμβο της λίστας (κεφαλή – head). Κατά σύμβαση, ο δείκτης σύνδεσης στον τελευταίο κόμβο της λίστας έχει μηδενική τιμή (NULL) για να σηματοδοτήσει το τέλος της λίστας.



Στην πρώτη άσκηση, θα χρησιμοποιήσουμε μία δομή δεδομένων που θα αποτελεί ένα κόμβο μίας διασυνδεδεμένης λίστας (linked list). Κάθε κόμβος θα αποτελείται από δύο λέξεις (των 32 bit καθεμία): έναν ακέραιο “data” που θα περιέχει την «πληροφορία χρήστη», κι έναν δείκτη σύνδεσης (pointer) “next” που θα περιέχει τη διεύθυνση του επόμενου κόμβου στη λίστα. Στον τελευταίο κόμβο της λίστας, next=0. Τα δύο στοιχεία (λέξεις) του κόμβου μας θα βρίσκονται σε διαδοχικές θέσεις της μνήμης. Επομένως, κάθε κόμβος θα έχει μέγεθος 2 x 4 = 8 bytes. Διεύθυνση ενός κόμβου είναι η διεύθυνση

του πρώτου στοιχείου του, δηλαδή του στοιχείου με «μηδενικό offset», που για μας είναι το ”data”.

Δυναμική Εκχώρηση Μνήμης (Dynamic Memory Allocation)

Το πρόγραμμά σας θα ζητάει και θα παίρνει κόμβους από το λειτουργικό σύστημα «δυναμικά», την ώρα που τρέχει (σε run-time). Για το σκοπό αυτό θα χρησιμοποιήσετε την κλήση συστήματος (system call) “**sbrk**” (set break). Η κλήση αυτή «σπρώχνει» πιο πέρα (προς αύξουσες διευθύνσεις μνήμης) το σημείο “break”, το όριο πριν από το οποίο οι διευθύνσεις μνήμης που γεννά το πρόγραμμα είναι νόμιμες, ενώ μετά από το οποίο (και μέχρι την αρχή της στοίβας) οι διευθύνσεις είναι παράνομες. Η κλήση συστήματος “sbrk” παίρνει σαν παράμετρο το πλήθος των νέων **bytes** που επιθυμεί ο χρήστης να δεσμεύσει στον καταχωρητή $a0. Μετά την επιστροφή της κλήσης, ο καταχωρητής $v0 περιέχει την **διεύθυνση** του νέου block μνήμης, του ζητηθέντος μεγέθους, που το σύστημα δίνει στο πρόγραμμά σας (έναν pointer). Σε περίπτωση που η διεύθυνση είναι μηδενική τότε έχει γεμίσει όλη η μνήμη και σε τέτοια περίπτωση το πρόγραμμά σας θα πρέπει να τερματίσει.

Άσκηση 1 - Δυναμική μνήμη (5 μονάδες)

Για την περιγραφή της άσκησης θα χρησιμοποιήσουμε την παρακάτω αναπαράσταση κόμβων:

Κόμβος λίστας:

```
typedef struct node {
        int data;
        struct node *next;
} nodeL;
```

a. Να γραφεί μία συνάρτηση **nodeL* insertElement(nodeL *head, int data)** όπου head είναι δείκτης στην αρχή της διασυνδεδεμένης λίστας και data είναι ο ακέραιος που θέλουμε να εισάγουμε στην λίστα. Η λίστα πρέπει να παραμένει πάντα **ταξινομημένη κατά αύξουσα σειρά** με βάση την τιμή του πεδίου data ανεξάρτητα από την σειρά με την οποία διαβάζουμε τους ακέραιους από την κονσόλα. Η συνάρτηση επιστρέφει την κεφαλή της λίστας head.

Η main θα διαβάζει επαναληπτικά από το πληκτρολόγιο θετικούς αριθμούς (> 0) και θα τους εισάγει στην λίστα καλώντας την συνάρτηση **insertElement**. Μόλις δοθεί το 0, τότε η main σταματάει να ζητάει στοιχεία και εκτυπώνει την λίστα (ερώτημα β).

Προσοχή:

- Η συνάρτηση πρέπει να ελέγχει την τιμή που επιστρέφει η κλήση συστήματος για την δυναμική δέσμευση μνήμης. Σε περίπτωση που αυτή είναι μηδέν, τότε εκτυπώνει κατάλληλο μήνυμα λάθους και το πρόγραμμα τερματίζει.
- Να προσέχετε τις ειδικές περιπτώσεις όπως την εισαγωγή σε κενή λίστα.

b. Να γραφεί μια αναδρομική συνάρτηση **void printList (nodeL *head)** όπου head είναι δείκτης στην αρχή της διασυνδεδεμένης λίστας, η οποία θα εκτυπώνει το περιεχόμενο της λίστας στην οθόνη ως εξής:

List: (1 2 2 3 6 12 15)

Η main θα καλέσει την συνάρτηση printList μόλις τελειώσει η επανάληψη που διαβάζει και εισάγει τα στοιχεία στην ταξινομημένη λίστα.

Άσκηση 2 - Αναδρομή (5 μονάδες)

Παρακάτω δίνεται η συνάρτηση ***pow*** που παίρνει δύο ορίσματα (*n* και *m*, και οι 32-bit) και επιστρέφει το n^m (δηλαδή το *n* υψωμένο στην *m*-οστή δύναμη). Η συνάρτηση θεωρεί ότι πάντα το *m* είναι μεγαλύτερο ή ίσο του 1.

```
int pow(int n, int m) {
        if (m == 1)
                return n;
        return n * pow(n, m-1);
}
```

Μεταφράστε αυτό το πρόγραμμα σε MIPS assembly, υλοποιώντας την αναδρομική συνάρτηση.

Άσκηση 3 - Ακρίβεια Αριθμητικής Κινητής Υποδιαστολής (1 μονάδα)

Βλέπετε κάποιο πρόβλημα στον παρακάτω κώδικα στο να υπολογίσει αυτό που περιμένετε; Εξηγήστε λεπτομερώς την απάντηση σας.

```
int test(float f, double d) {
     return (((double) f) == d);

}
```