ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

Οντοκεντρικός Προγραμματισμός

Ενότητα 8: C++ ΒΙΒΛΙΟΘΗΚΗ STL, ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ

Δομές Δεδομένων

ΔΙΔΑΣΚΟΝΤΕΣ: Ιωάννης Χατζηλυγερούδης, Χρήστος Μακρής

Πολυτεχνική Σχολή

Τμήμα Μηχανικών Η/Υ & Πληροφορικής

Δομές Δεδομένων

Εισαγωγή

- Δομές σταθερού μεγέθους
  - Arrays, structs
- Δυναμικές δομές δεδομένων
  - Το μέγεθος αυξάνεται και μειώνεται κατά την εκτέλεση του προγράμματος
  - Διασυνδεμένες λίστες (Linked lists)
    - Είσοδος και αφαίρεση από οποιοδήποτε σημείο
  - Στοίβες (Stacks)
    - Είσοδος και αφαίρεση από την κορυφή
  - Ουρές (Queues)
    - Είσοδος από το τέλος, αφαίρεση από την αρχή
  - Δυαδικά Δέντρα (Binary trees)
    - Αποδοτική αναζήτηση/ταξινόμηση

Αυτό-αναφερόμενη κλάση

- Αυτό-αναφερόμενες κλάσεις (Self-referential)
  - Έχουν δείκτη σε αντικείμενο της ίδια κλάσης
  - Σύνδεση μεταξύ τους για σχηματισμό χρήσιμων δομών
    - Lists, stacks, queues, trees
  - Η δομή τερματίζεται με το τελευταίο στοιχείο να δείχνει στο **NULL**

```
class Node {
   public:
      Node( int );
   private:
      int data;
      Node *nextPtr;
};
```

Δέσμευση/Αποδέσμευση Μνήμης

- Δυναμική δέσμευση μνήμης
  - Δέσμευση και απελευθέρωση μνήμης κατά την εκτέλεση
  - Δημιουργία και αφαίρεση κόμβων
- Τελεστής **`new`**
  - Επιστρέφει δείκτη στο νέο αντικείμενο που δημιουργεί
    - **`Node *newPtr = new Node( 5 );`**
    - Πετάει εξαίρεση **`bad_alloc`** αν δεν υπάρχει διαθέσιμη μνήμη
- Τελεστής **`delete`**
  - Αποδεσμεύει την μνήμη που δεσμεύτηκε κατά την δημιουργία
    - **`Delete newPtr;`**
  - Προσοχή: Δεν διαγράφεται ο δείκτης, απλά αποδεσμεύεται η θέση στην οποία εξακολουθεί να δείχνει.

Διασυνδεμένη Λίστα

- Διατηρεί δείκτη στο πρώτο στοιχείο (Node) της λίστας
  - Κάθε στοιχείο διαθέτει δείκτη προς το επόμενο οπότε μπορούμε να τα διαπεράσουμε σειριακά
- Το τελευταίο στοιχείο δείχνει στο null (zero), σηματοδοτώντας ότι φτάσαμε στο τέλος της λίστας
- Δυναμική Λίστα
  - Κόμβου προστίθενται και αφαιρούνται ανάλογα τις ανάγκες
  - Η τιμή που αποθηκεύεται στον κόμβο (value) μπορεί να είναι οποιοδήποτε τύπου
- Λειτουργίες:
  - Είσοδος κόμβου στην αρχή
  - Είσοδος κόμβου στο τέλος
  - Αφαίρεση κόμβου από την αρχή
  - Αφαίρεση κόμβου από το τέλος

Τύποι Διασυνδεμένων Λιστών

- Τύποι Διασυνδεμένων Λιστών
  - Μονά συνδεδεμένη Λίστα
    - Pointer to first node
    - Διαπέραση προς μια κατεύθυνση μόνο (το τελευταίο δείχνει στο null)
  - Κυκλική, μονά συνδεδεμένη
    - Όμοια, αλλά το τελευταίο δείχνει στο πρώτο
  - Διπλά συνδεδεμένη
    - Κάθε κόμβος έχει δείκτη στο επόμενο και στο προηγούμενο στοιχείο
    - Διαπέραση και προς τις δύο κατευθύνσεις
    - Ο τελευταίος κόμβος δείχνει στο null ως επόμενο
    - Ο πρώτος κόμβος δείχνει στο null ως προηγούμενο
  - Κυκλική, διπλά συνδεδεμένη
    - Όμοια, αλλά το πρώτο δείχνει στο τελευταίο και αντόστροφα

Στοίβα

- Στοίβα (Stack)
  - Κόμβοι προστίθενται και αφαιρούνται από την κορυφή
    - Περιορισμένη έκδοση διασυνδεμένης λίστας
  - Δομή: Last-in, first-out (LIFO)
  - Ο πάτος της στοίβας δείχνει στο null
- Λειτουργίες
  - Push: προσθήκη κόμβου στην κορυφή
  - Pop: Αφαίρεση κόμβου από την κορυφή

Υλοποίηση στοίβας

```
class Node{
  private:
    int value;
    Node* next;
  public:
    Node (int valueIn, Node* nextIn): value(valueIn), next(nextIn) {}
    ~Node() {} int getValue() const {return value;}
    Node* getNext() const {return next;}
};
```

```
class Stack{
  private:
    Node* top;
  public:
    Stack(){ top=0; }
    ~Stack() {
      Node* A;
      while(top!=0){
        A=top;
        top=top->getNext();
        delete A;
      }
    }
    void push(int k){
      Node* n = new Node(k,top);
      top = n;
    }
    int pop(){
      if (top==0)
        return 0;
      Node* A=top;
      int k=top->getValue();
      top=top->getNext();
      delete A;
      return k;
    }
};
```

```
int main() {
  Stack s;
  s.push(1);
  s.push(2);
  s.push(3);
  cout<< s.pop(); // 3
  cout<< s.pop(); // 2
  cout<< s.pop(); // 1
  cout<< s.pop(); // 0
  return 0;
}
```

Ουρά

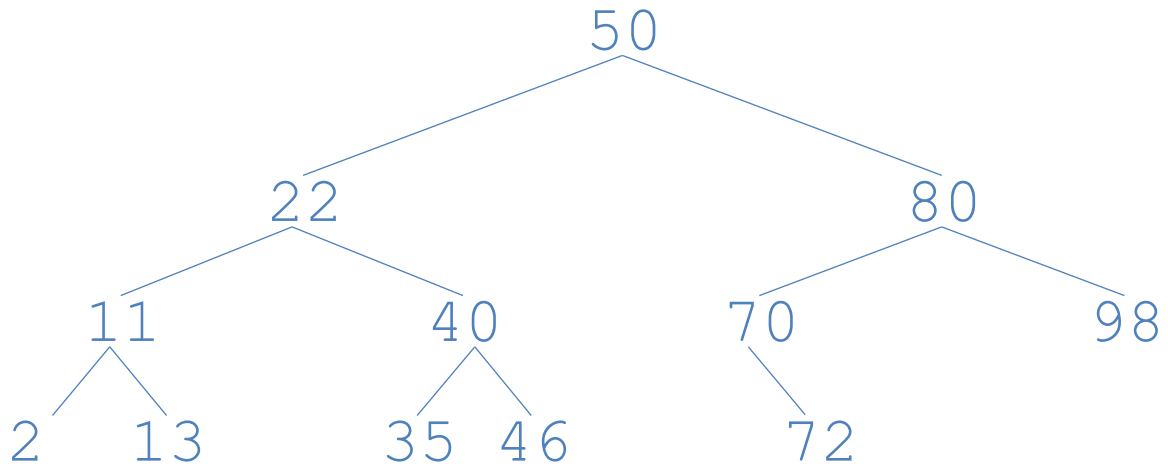
- Ουρά (Queue)
  - Σαν γραμμή αναμονής
  - Είσοδος στο τέλος(ουρά) , αφαίρεση από την αρχή.
  - Δομή: First-in, first-out (FIFO)
- Λειτουργίες
  - Enqueue: Κόμβοι προστίθενται στο τέλος (*ουρά - tail*)
  - Dequeue: αφαίρεση από μπροστά (*κεφαλή - head*)

Δέντρα

- Δέντρα - Trees
  - Μη-γραμμικές δομές δύο διαστάσεων
  - Οι κόμβοι έχουν δύο ή περισσότερες συνδέσεις
  - Τα δυαδικά έχουν ακριβώς 2 συνδέσεις/δείκτες από τους οποίους μπορεί ένας ή και οι δύο να είναι null (π.χ φύλλα του δέντρου)
- Ορολογία
  - Ρίζα – Root: ο πρώτος κόμβος του δέντρου
  - Οι συνδέσεις/δείκτες ενός κόμβου αναφέρονται στα παιδιά του
  - Φύλλο – Leaf: κόμβος χωρίς παιδιά

Δυαδικό Δέντρο Αναζήτησης

- Δυαδικό δέντρο αναζήτησης
  - Οι τιμές στο αριστερό υπο-δέντρο κάθε κόμβου είναι μικρότερες από την τιμή του πατέρα
  - Οι τιμές στο δεξιό υπο-δέντρο κάθε κόμβου είναι μεγαλύτερες από την τιμή του πατέρα
  - Δεν επιτρέπονται διπλές τιμές
  - Γρήγορη αναζήτηση, log2n συγκρίσεις για ισορροπημένα δέντρα



Δυαδικό Δέντρο Αναζήτησης

- Είσοδος κόμβων
  - Χρήση αναδρομικής συνάρτησης
  - Ξεκινάει από την ρίζα
  - Αν ο τρέχον κόμβος είναι άδειος, εισαγωγή του κόμβου σε αυτόν (τέλος αναδρομής)
  - Αλλιώς,
    - Αν η τιμή του νέου κόμβου είναι μεγαλύτερη από του τρέχοντα κόμβου, εισαγωγή στο δεξιά υποδέντρο (αναδρομική κλήση)
    - Αν η τιμή του νέου κόμβου είναι μικρότερη από του τρέχοντα κόμβου, εισαγωγή στο αριστερά υποδέντρο (αναδρομική κλήση)
    - Αλλιώς (ίδια τιμή), αγνόησε τον κόμβο (υπάρχει ήδη)

Πρόσθετο Υλικό

- Μελετήστε και τα παραδείγματα από το **Κεφάλαιο 19** του βιβλίου:
  «C++ How to Program, 9/e Paul & Harvey Deitel»
  http://media.pearsoncmg.com/ph/esm/deitel/cpp_htp_9/code_examples/Code_Examples.zip

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σημείωμα Ιστορικού Εκδόσεων Έργου

Το παρόν έργο αποτελεί την έκδοση 1.0.1

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.

Σημείωμα Χρήσης Έργων Τρίτων

- Οι διαφάνειες βασίζονται στο βιβλίο «C++ How to Program, 8th Edition, Harvey M. Deitel, Paul J. Deitel, Prentice Hall.»