ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Προγραμματιστικές Ασκήσεις, Φυλλάδιο 1

1η ομάδα ασκήσεων: Να ολοκληρωθούν μέχρι 10' πριν τη λήξη του εργαστηρίου

1. Κατασκευάστε ένα πρόγραμμα που υλοποιεί το εξής απλό παιχνίδι, με δύο παίχτες: το χρήστη και τον παίχτη-υπολογιστή. Αρχικά ο υπολογιστής διαλέγει έναν τυχαίο φυσικό αριθμό μεταξύ 0 και 9. Υπάρχουν τέσσερις γύροι στους οποίους ο χρήστης παίζει εναντίον του παίχτη-υπολογιστή. Σε κάθε γύρο ο χρήστης δίνει έναν πιθανό αριθμό. Ο παίχτης-υπολογιστής διαλέγει κι αυτός έναν πιθανό αριθμό, από αυτούς που δεν έχει ήδη διαλέξει. Αν κάποιος από τους παίχτες βρει τον αριθμό το παιχνίδι τελειώνει. Αν κανείς δεν βρει τον αριθμό τότε δεν υπάρχει νικητής. Στην υλοποίηση σας θα ορίσετε μια κλάση που να υλοποιεί τον παίχτη-υπολογιστή, καθώς και το βασικό loop του παιχνιδιου.

2η ομάδα ασκήσεων: Να ολοκληρωθούν μέχρι 22/11/09 στις 12 μ.μ.

Οι παρακάτω ασκήσεις πρέπει να παραδοθούν μέχρι τις 22 Νοεμβρίου τα μεσάνυχτα. Για την υλοποίηση τους θα πρέπει να χρησιμοποιήσετε κλάσεις και αντικείμενα. Στις κλάσεις δεν πρέπει να χρησιμοποιείτε public μεταβλητές. Εκτός από σταθερές, δεν πρέπει να χρησιμοποιήσετε καθολικές μεταβλητές (global variables). Για να πάρετε όλους τους βαθμούς θα πρέπει να φροντίσετε και τις οριακές καταστάσεις (δηλαδή τις περιπτώσεις που μπορεί να δημιουργηθεί κάποιο λάθος, π.χ. να προσπαθήσουμε να διαβάσουμε κάποια θέση εκτός ορίων ενός πίνακα).

2. Κατασκευάσετε μια κλάση η οποία ορίζει ένα πίνακα μεγέθους N = 10, και υπερφορτώνει τον τελεστή `[]`, έτσι ώστε να έχει τα εξής χαρακτηριστικά:
   - Η πρόσβαση (διάβασμα και γράψιμο) των στοιχείων του πίνακα γίνεται χρησιμοποιώντας αριθμούς θέσεων 1…N, αντί για 0…N όπως γίνεται παραδοσιακά στην C++.
   - Εγγυάται ασφαλή πρόσβαση στα δεδομένα του πίνακα. Αν προσπαθήσει ο χρήστης να ζητήσει μια θέση εκτός των ορίων του πίνακα, το πρόγραμμα τυπώνει ένα μήνυμα λάθους και σταματάει.

3. Ο δυναμικός πίνακας είναι ένας πίνακας που επιτρέπει την πρόσθεση και αφαίρεση στοιχείων στο τέλος του και η χωρητικότητα του (capacity) αλλάζει δυναμικά. Αρχικά δεσμεύει ένα χώρο μνήμης μιας συγκεκριμένης χωρητικότητας C, όπου C θα ορίζεται ως σταθερά. Νέα στοιχεία προστίθενται στην επόμενη διαθέσιμη θέση του πίνακα. Αν η πρόσθεση ενός νέου στοιχείου έχει ως αποτέλεσμα να ξεπεραστεί η χωρητικότητα του πίνακα, τότε νέα ποσότητα μνήμης διπλάσιας χωρητικότητας δεσμεύεται και όλα τα στοιχεία του πίνακα αντιγράφονται σε αυτό το νέο χώρο μνήμης και το νέο στοιχείο προστίθεται στην κατάλληλη θέση. Η αφαίρεση αφαιρεί το τελευταίο στοιχείο του πίνακα. Αν η αφαίρεση έχει ως αποτέλεσμα το μέγεθος του πίνακα να γίνει το ¼ της χωρητικότητας, και μεγαλύτερο από C, τότε δεσμεύεται νέος χώρος μνήμης της μισής χωρητικότητας και τα στοιχεία του πίνακα αντιγράφονται στο νέο χώρο μνήμης.

   Κατασκευάσετε μια κλάση που υλοποιεί ένα δυναμικό πίνακα από `integers`. Η κλάση θα έχει τις μεθόδους `Add(int)`, και `Delete()`. Η μέθοδος `Add(int x)` προσθέτει την τιμή `x` στο τέλος του πίνακα, και διπλασιάζει την χωρητικότητα του πίνακα αν χρειάζεται. Η μέθοδος `Delete()` αφαιρεί το τελευταίο στοιχείο του πίνακα, και υποδιπλασιάζει το μέγεθος του πίνακα, αν είναι απαραίτητο. Η συνάρτηση δόμησης (constructor) δεσμεύει την αρχική μνήμη για τον πίνακα, και η συνάρτηση αποδόμησης (destructor) αποδεσμεύει τον χώρο μνήμης. Η κλάση θα πρέπει να έχει και μια μέθοδο `GetElement(int i)` η οποία επιστρέφει το στοιχείο στη θέση `i` (εναλλακτικά μπορείτε να υπερφορτώσετε τον τελεστή `[]` για διάβασμα της θέσης i). Επίσης μεθόδους `GetSize()` και `GetCapacity()` που επιστρέφουν το μέγεθος και τη χωρητικότητα του πίνακα.

   Στη main() θα υλοποιήσετε ένα τεστ για την κλάση σας. Ορίσετε ένα αντικείμενο της κλάσης που ορίσατε, με αρχική χωρητικότητα C = 4. Κάνετε 2 Add, 1 Delete, 9 Add, 8 Delete, καθώς και μερικές προσβάσεις σε στοιχεία του πίνακα. Σε κάθε κλήση τυπώνετε το μέγεθος και τη χωρητικότητα του πίνακα.

4. Υλοποιήστε ένα απλό παιχνίδι ναυμαχίας μεταξύ ενός χρήστη και του υπολογιστή. Στο παιχνίδι αυτό ο κάθε παίχτης δημιουργεί ένα πλοίο μεγέθους δύο θέσεων το οποίο τοποθετείται σε ένα μονοδιάστατο

χώρο 10 θέσεων. Στη συνέχεια οι δύο παίχτες παίζουν σε γύρους, όπου ο κάθε παίχτης κάνει μια βολή, επιλέγοντας μια θέση μέσα στο διάστημα των δέκα θέσεων για να χτυπήσει. Το παιχνίδι συνεχίζεται μέχρι ο ένας παίχτης να βυθίσει το πλοίο του άλλου.

Όπως και στο παιχνίδι με χαρτί, για κάθε παίχτη υπάρχουν δύο βασικές οντότητες. Η μια είναι το `ShipBoard` στο οποίο ο παίχτης κρατάει λογαριασμό για το πού είναι τα δικά του πλοία, και η άλλη είναι το `StrikeBoard` όπου κρατάει λογαριασμό για το ποιες θέσεις έχει χτυπήσει και ποιο ήταν το αποτέλεσμα της κάθε βολής. Αυτές οι οντότητες θα πρέπει να υλοποιηθούν ως κλάσεις.

Η κλάση `ShipBoard` θα πρέπει να έχει μεθόδους που να κάνουν τα εξής.

- Μέθοδος που να επιτρέπει την τοποθέτηση του πλοίου. Η θέση του πλοίου θα καθορίζεται ορίζοντας τη θέση στην οποία ξεκινάει. Υπερφορτώνοντας την μέθοδο τοποθέτησης θα πρέπει η μέθοδος να μπορεί να τοποθετεί το πλοίο είτε σε μια τυχαία θέση, είτε σε μια θέση που έχει καθορίσει ο χρήστης.
- Μέθοδος η οποία δεδομένης μιας θέση βολής, επιστρέφει μια `Boolean` τιμή αν χτύπησε το πλοίο ή όχι.

Η κλάση `StrikeBoard` θα πρέπει να έχει μεθόδους για τα ακόλουθα.

- Μια μέθοδο αρχικοποίησης που να δίνει τις αρχικές τιμές στις μεταβλητές της κλάσης. Αυτό μπορεί να γίνει και στη συνάρτηση δόμησης της κλάσης.
- Μια μέθοδο που τυπώνει την κατάσταση για όλες τις θέσεις.
- Μια μέθοδο, η οποία θα παίρνει το αποτέλεσμα μιας βολής και θα ενημερώνει το ιστορικό των θέσεων.
- Μία μέθοδο που θα αποφασίζει αν ο παίχτης έχει κερδίσει. (Σημ. Αυτή τη μέθοδο μπορείτε να την κάνετε και μέλος της κλάσης `ShipBoard`).
- Μία μέθοδο που θα υλοποιεί την τακτική του παίχτη-υπολογιστή. Ο υπολογιστής όσο δεν έχει χτυπήσει κάποια θέση στην οποία να είναι το πλοίο θα διαλέγει τυχαία μια θέση από αυτές που δεν έχει χτυπήσει ήδη. Αν χτυπήσει μία θέση του πλοίου τότε στον επόμενο γύρο, κοιτάει τις θέσεις αριστερά και δεξιά της θέσης που χτύπησε, και διαλέγει τυχαία από τις θέσεις που δεν είναι ήδη χτυπημένες.

Η συνάρτηση `main` θα μοιάζει ως εξής:

```
int main()
{
// ορισμός αντικειμένων και αρχικοποίηση όσο δεν έχουμε νικητή
// Ο παίχτης-χρήστης βλέπει την παρούσα κατάσταση και
// δίνει την επόμενη θέση να χτυπήσει, γίνεται η βολή,
// παίρνει το αποτέλεσμα και ενημερώνει την κατάσταση του.
// Αν έχει κερδίσει σταματάμε
// Ο παίχτης-υπολογιστής επιλέγει την θέση να χτυπήσει
// γίνεται η βολή, παίρνει το αποτέλεσμα και ενημερώνει την
// κατάσταση του. Αν έχει κερδίσει σταματάμε
}
```

Δημιουργείστε δύο παίχτες που ελέγχονται και οι δύο από τον υπολογιστή. Στην περίπτωση αυτή στο βασικό loop της `main` τα βήματα για τους δύο παίχτες είναι πανομοιότυπα. Δημιουργείστε μια συνάρτηση που υλοποιεί τα βήματα ενός παίχτη με δύο τρόπους: (α) Ως συνάρτηση του κύριου προγράμματος, και (β) ως μέθοδο της κλάσης `StrikeBoard`.

ΓΕΝΙΚΕΣ ΟΔΗΓΙΕΣ

Ο κώδικας C++ προγραμμάτων αποθηκεύεται σε αρχεία με την επέκταση .cpp αντί για .c
Π.χ. lab1_dddd.cpp

Η μετάφραση των αρχείων γινεται με το g++ αντί του gcc
Π.χ. g++ lab1_dddd.cpp –o lab1_dddd

Κάντε turnin τα προγράμματα σας στο lab1@cs435. Για διευκόλυνση **ονομάστε το αρχείο με τον κώδικά σας lab1_dddd.cpp όπου dddd είναι ο Α.Μ. σας.**

π.χ. turnin lab1@cs435 lab1_dddd.cpp

Στον κώδικα να αναγράφονται σε σχόλια το όνομα το login και ο ΑΜ σας.