ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ ΠΛΥ212/ΜΥΥ205
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ & ΠΛΗΡΟΦΟΡΙΚΗΣ Τεχνικές Αντικειμενοστρεφούς Προγραμματισμού

Πρώτη Σειρά ασκήσεων
Ημερομηνία Παράδοσης: Πέμπτη 11 Μαΐου 11:59 π.μ. (πριν το μάθημα)

Για την άσκηση αυτή θα υλοποιήσετε σε Java ένα πρόγραμμα που υλοποιεί μια ανταγωνιστική εκδοχή του παιχνιδιού MineSweeper. Παρόμοια με το γνωστό παιχνίδι, υπάρχει ένα ναρκοπέδιο το οποίο αποτελείται από ένα δισδιάστατο πλέγμα στο οποίο υπάρχουν κρυμμένες νάρκες. Ο παίκτης μπορεί να ανοίξει ένα κελί του πλέγματος και να πληροφορηθεί αν υπάρχει νάρκη, ή τον μετρητή του κελιού, δηλαδή πόσες νάρκες υπάρχουν γειτονικά του κελιού (ένα κελί που δεν είναι στα άκρα έχει 8 γειτονικά κελιά). Σε αντίθεση με την καθιερωμένη εκδοχή του παιχνιδιού στην εκδοχή που θα υλοποιήσετε υπάρχουν δύο παίχτες και ο στόχος τους είναι να βρούνε όσες περισσότερες νάρκες μπορούν. Κάθε φορά που κάποιος παίχτης ανοίγει ένα κελί με νάρκη κερδίζει ένα πόντο. Αν δεν έχει νάρκη το κελί αποκαλύπτεται ο μετρητής του. Οι παίχτες παίζουν εναλλάξ και το παιχνίδι τελειώνει όταν δεν υπάρχουν άλλες νάρκες να αποκαλυφθούν. Κερδίζει ο παίχτης που έχει αποκαλύψει τις περισσότερες νάρκες, δηλαδή τους περισσότερους πόντους.

Για την υλοποίηση σας θα πρέπει να δημιουργήσετε 5 κλάσεις. Την κλάση **Cell** η οποία κρατάει πληροφορία για ένα κελί του πλέγματος, την κλάση **MineField** η οποία κρατάει πληροφορία για το πλέγμα με τις νάρκες, την κλάση **Player** που υλοποιεί το παιχνίδι ενός παίκτη, την κλάση **MineSweeper** που κρατάει πληροφορίες για ένα παιχνίδι και υλοποιεί την ροή του παιχνιδιού και την κλάση **Game** η οποία έχει την main.

Cell: Η κλάση Cell κρατάει πληροφορίες για ένα κελί του πλέγματος. Έχει τα εξής πεδία:

- Τις συντεταγμένες (row, column) του κελιού μέσα στο πλέγμα
- Ένα Boolean πεδίο για το αν το κελί περιέχει νάρκη
- Ένα Boolean πεδίο για το αν έχει ανοιχτεί το κελί
- Τον μετρητή του κελιού, δηλαδή αριθμό των γειτόνων του κελιού που περιέχουν νάρκη
- Μια λίστα με τα γειτονικά κελιά (Cell) του κελιού.

Η κλάση θα πρέπει να έχει τις εξής μεθόδους:

- Τον **constructor** ο οποίος παίρνει σαν όρισμα τις συντεταμένες του κελιού και τις αρχικοποιεί.
- Την μέθοδο **addNeighbor** η οποία παίρνει σαν όρισμα ένα κελί και το προσθέτει στους γείτονες του κελιού. Η μέθοδος επίσης προσθέτει και το τρέχον κελί στους γείτονες του κελιού-ορίσματος (κάνετε χρήση της μεταβλητής this).
- Την μέθοδο **addMine** η οποία προσθέτει μια νάρκη στο κελί (ενημερώνει το αντίστοιχο Boolean πεδίο). Η μέθοδος επίσης αυξάνει τον μετρητή για όλα τα γειτονικά κελιά, εφόσον αυτά δεν έχουν νάρκη.
- Την μέθοδο **open** η οποία ανοίγει το κελί
- Την μέθοδο **isOpen** που ελέγχει αν ένα κελί είναι ανοιχτό.
- Την μέθοδο **containsMine** που επιστρέφει true ή false αν το κελί περιέχει νάρκη ή όχι.
- Την μέθοδο **toString** που επιστρέφει την String αναπαράσταση του κελιού. Αν το κελί δεν είναι ανοιχτό, επιστρέφει το String "-". Αν το κελί είναι ανοιχτό και περιέχει νάρκη, επιστρέφει το String "*". Αν το κελί είναι ανοιχτό και δεν περιέχει νάρκη επιστρέφει τον μετρητή του κελιού.

Υπόδειξη: Αν θέλετε να κάνετε debug τον κώδικα σας μπορεί να σας βολέψει να τυπώσετε τις συντεταγμένες των γειτονικών κελιών ενός κελιού.

MineField: Η κλάση κρατάει πληροφορίες για το πλέγμα με τις νάρκες. Η κλάση έχει πεδίο το μέγεθος του πλέγματος (αν το μέγεθος είναι 10, θα δημιουργήσετε ένα πλέγμα 10×10), τον αριθμό από τις νάρκες, και ένα δισδιάστατο πίνακα με κελιά.

Η κλάση θα πρέπει να έχει τις εξής μεθόδους:

- Τον **constructor** ο οποίος παίρνει σαν όρισμα το μέγεθος και τον αριθμό των ναρκών και αρχικοποιεί το πλέγμα. Ο constructor θα πρέπει να δημιουργήσει τον πίνακα, να αρχικοποιήσει και να συνδέσει τα κελιά, και να τοποθετήσει τις νάρκες σε τυχαίες θέσεις μέσα στο πλέγμα. Για τις δύο τελευταίες λειτουργίες, δημιουργείστε δύο βοηθητικές μεθόδους initializeCells και initializeMines.
- Η μέθοδος **initializeCells** θα πρέπει να δημιουργήσει τα κελιά για κάθε θέση του πίνακα και να τα συνδέσει μεταξύ τους (δηλαδή να φτιάξετε τις σχέσεις γειτνίασης) . Σας προτείνετε να κάνετε πρώτα την δημιουργία των κελιών και μετά να τα προσθέσετε. Θυμηθείτε ότι η μέθοδος addNeighbor προσθέτει γείτονα και στο κελί που καλεί την μέθοδο, και στο κελί-όρισμα, οπότε να είσαστε προσεκτικοί να μην προσθέσετε ένα κελί δύο φορές. Χρειάζεται προσοχή στις οριακές περιπτώσεις όπου ένα κελί δεν έχει 8 γείτονες.
- Η μέθοδος **initializeMines** προσθέτει τις νάρκες σε τυχαίες θέσεις του πλέγματος. Χρησιμοποιήστε την κλάση Random για να επιλέξετε τυχαίες συντεταγμένες. Πρέπει να τοποθετηθούν όλες οι νάρκες. Ένα κελί δεν μπορεί να περιέχει πάνω από μία νάρκη.
- Την μέθοδο **getCell** η οποία παίρνει σαν όρισμα συντεταγμένες στο πλέγμα και επιστρέφει το αντίστοιχο κελί.
- Την μέθοδο **print**, η οποία τυπώνει το πλέγμα. Τυπώστε και τον αύξοντα αριθμό της γραμμής και στήλης. Συνίσταται, τα κελιά να απέχουν μεταξύ τους κατά 2 κενά, και ο αύξον αριθμός της γραμμής κατά ένα tab. Θα βρείτε παραδείγματα στα Παραδείγματα Εξόδου, στο τέλος της εκφώνησης.
- Δημιουργείστε μία **main** για να τεστάρετε τον κώδικά σας. Δημιουργείστε ένα ναρκοπέδιο μεγέθους 10 με 20 νάρκες. Ανοίξτε όλα τα κελιά και εκτυπώστε το αποτέλεσμα ώστε να βεβαιωθείτε ότι έχετε κάνει σωστά τον υπολογισμό των μετρητών. Στα Παραδείγματα Εξόδου φαίνεται πως πρέπει να φαίνεται η έξοδος. Για να πάρετε την ίδια έξοδο, πριν την τοποθέτηση των ναρκών καλέστε την μέθοδο setSeed από το αντικείμενο Random με όρισμα 2023.

Player: Η κλάση αυτή κρατάει πληροφορία για ένα παίκτη και υλοποιεί το παιχνίδι του. Τα πεδία που χρειαζόμαστε είναι το όνομα του παίχτη και το score του, δηλαδή τον αριθμό των ναρκών που έχει βρει μέχρι τώρα.
Η κλάση έχει τις εξής μεθόδους:

- Τον **constructor** ο οποίος αρχικοποιεί το όνομα.
- Την μέθοδο **play**, η οποία παίρνει σαν όρισμα ένα αντικείμενο MineField, και υλοποιεί το παιχνίδι του παίχτη. Ρωτάει τον παίχτη (με το όνομα του) ποιο κελί θέλει να ανοίξει (δεν είναι απαραίτητο να ελέγξετε αν οι συντεταγμένες είναι εντός των ορίων του πλέγματος). Αν ο παίχτης επιλέξει ένα ανοιγμένο κελί του ξαναζητάει είσοδο. Αλλιώς ανοίγει το κελί. Αν υπάρχει νάρκη τυπώνεται ένα μήνυμα και ενημερώνει το score. Η μέθοδος επιστρέφει μια boolean τιμή αν βρήκε νάρκη ή όχι.
- Την μέθοδο **printStatus** που τυπώνει το όνομα του παίχτη και τον σκορ του.
- **Accessor** μεθόδους για τα πεδία της κλάσης.

MineSweeper: Η κλάση που υλοποιεί τη βασική ροή του παιχνιδιού. Έχει πεδία το ναρκοπέδιο MineField και τους δύο παίχτες (μπορείτε να προσθέσετε κι άλλα πεδία αν θέλετε). Ο constructor παίρνει σαν όρισμα το μέγεθος του ναρκοπεδίου, τον αριθμό των ναρκών και τα ονόματα των παιχτών και αρχικοποιεί κατάλληλα τα πεδία. Η βασική μέθοδος είναι η **play** η οποία υλοποιεί το παιχνίδι. Τυπώνεται αρχικά το (κλειστό) πλέγμα. Στη συνέχεια, οι παίχτες παίζουν εναλλάξ. Αφού παίξει ένας παίκτης τυπώνεται το πλέγμα, ο αριθμός των ναρκών που έχουν βρεθεί και αυτών που είναι ακόμη κρυμμένες και τα scores και των δύο παιχτών (θα σας εξυπηρετήσει να φτιάξετε μια βοηθητική μέθοδο για τις εκτυπώσεις). Το παιχνίδι σταματάει όταν βρεθούν όλες οι νάρκες και ανακηρύσσεται ο τελικός νικητής (αν δεν είναι ισοπαλία).

Game: Η κλάση που έχει τη **main**. Ζητάει από τον χρήστη τις παραμέτρους του παιχνιδιού (μέγεθος πλέγματος, αριθμό από νάρκες, ονόματα παιχτών), δημιουργεί το αντικείμενο MineSweeper και καλεί την play.

Bonus: Σε κάποιες περιπτώσεις όταν ανοίγετε ένα κελί αυτό σας δίνει την δυνατότητα να αποκλείσετε κάποια άλλα κελιά ως υποψήφια για να έχουν νάρκη. Π.χ., αν ανοίξετε ένα κελί με μετρητή μηδέν μπορείτε να αποκλείσετε όλους τους γείτονες του κελιού. Αν ανοίξετε ένα κελί με νάρκη το οποίο έχει γείτονα με μετρητή 1 μπορείτε να

αποκλείσετε όλους τους γείτονες του γείτονα. Προσθέστε ένα επιπλέον Boolean πεδίο isCandidate στην κλάση Cell για το αν το κελί είναι υποψήφιο (αρχικά true για όλα τα κελιά). Στη μέθοδο open όπου ανοίγεται το κελί, ενημερώνετε το isCandidate για όλα τα κελιά που πλέον ξέρουμε ότι δεν μπορεί να είναι υποψήφια. Μπορείτε να βρείτε αυτά τα κελιά από τους γείτονες, και τους γείτονες των γειτόνων. Η toString θα επιστρέφει "x" για τα κλειστά, μη υποψήφια κελιά.

ΓΕΝΙΚΕΣ ΟΔΗΓΙΕΣ

- Μια κλάση που δεν κάνει compile **μηδενίζεται** αυτόματα.
- Δεν επιτρέπεται η χρήση public ή protected πεδίων στην άσκηση. Επίσης ο κώδικας θα πρέπει να είναι σωστά στοιχισμένος και καλά γραμμένος. Θα αφαιρεθούν βαθμοί από προγράμματα που είναι πολύ κακά γραμμένα.
- Θα τεστάρουμε και θα βαθμολογήσουμε την κάθε κλάση ξεχωριστά. Γι αυτό και θα πρέπει να σώσετε την κάθε κλάση σε ξεχωριστό αρχείο. Θα πρέπει επίσης να κρατήσετε τα ονόματα και τα ορίσματα των public μεθόδων ακριβώς όπως σας ζητούνται.
- Κάντε turnin τα προγράμματα σας στο assignment1@myy205.

  π.χ. turnin assignment1@myy205 MineSweeper.java

  Μπορείτε να κάνετε turnin πολλά αρχεία μαζί στην ίδια εντολή. Διαβάστε προσεκτικά τις οδηγίες για το trunin στο ecourse και βεβαιωθείτε ότι μπορείτε να κάνετε την διαδικασία κάποιες μέρες πριν την προθεσμία. Μην το αφήσετε αυτό για την τελευταία στιγμή! Μπορείτε να κάνετε πολλαπλές φορές turnin τα ίδια αρχεία, θα κοιτάξουμε το τελευταίο. Κάθε φορά πρέπει να κάνετε turnin όλα τα αρχεία που θέλετε να παραδώσετε. Δεν μπορείτε να κάνετε turnin zip αρχείο, ή αρχείο με ελληνικούς χαρακτήρες.

  Στον κώδικα να αναγράφονται σε σχόλια το όνομα και ο ΑΜ σας (**με λατινικούς χαρακτήρες**).

Παραδείγματα Εξόδου:

Παρακάτω σας δίνονται μερικά παραδείγματα εξόδου. Δεν είναι ανάγκη η έξοδος σας να είναι ακριβώς έτσι αλλά πρέπει να είναι παρόμοια.

Παράδειγμα εξόδου για την main της κλάσης Minefield: (Χρησιμοποιήθηκε το seed 2023)

```
>java MineField

     0  1  2  3  4  5  6  7  8  9

0    1  2  2  1  1  1  2  1  1  0
1    1  *  *  1  1  *  2  *  1  0
2    1  2  2  2  2  2  3  2  3  1
3    0  0  1  2  *  2  2  *  2  *
4    0  1  2  *  2  3  *  4  3  1
5    0  1  *  2  2  3  *  *  3  1
6    0  1  1  2  3  *  5  *  *  3
7    0  0  1  2  *  *  3  3  *  *
8    0  0  1  *  3  2  1  1  2  2
9    0  0  1  1  1  0  0  0  0  0
```

Παράδειγμα παιχνιδιού: (Χρησιμοποιήθηκε το seed 2023)

```
>java Game
Give the board size:
10
Give the number of mines:
20
Give the names of the players:
Alice Bob

        0  1  2  3  4  5  6  7  8  9

0       -  -  -  -  -  -  -  -  -  -
1       -  -  -  -  -  -  -  -  -  -
2       -  -  -  -  -  -  -  -  -  -
3       -  -  -  -  -  -  -  -  -  -
4       -  -  -  -  -  -  -  -  -  -
5       -  -  -  -  -  -  -  -  -  -
6       -  -  -  -  -  -  -  -  -  -
7       -  -  -  -  -  -  -  -  -  -
8       -  -  -  -  -  -  -  -  -  -
9       -  -  -  -  -  -  -  -  -  -

Player Alice give the coordinates for the cell to open:
4 4
        0  1  2  3  4  5  6  7  8  9

0       -  -  -  -  -  -  -  -  -  -
1       -  -  -  -  -  -  -  -  -  -
2       -  -  -  -  -  -  -  -  -  -
3       -  -  -  -  -  -  -  -  -  -
4       -  -  -  -  2  -  -  -  -  -
5       -  -  -  -  -  -  -  -  -  -
6       -  -  -  -  -  -  -  -  -  -
7       -  -  -  -  -  -  -  -  -  -
8       -  -  -  -  -  -  -  -  -  -
9       -  -  -  -  -  -  -  -  -  -

0 mines found, 20 mines left
Player Alice: 0 mines found
Player Bob: 0 mines found

Player Bob give the coordinates for the cell to open:
7 7
        0  1  2  3  4  5  6  7  8  9

0       -  -  -  -  -  -  -  -  -  -
1       -  -  -  -  -  -  -  -  -  -
2       -  -  -  -  -  -  -  -  -  -
3       -  -  -  -  -  -  -  -  -  -
4       -  -  -  -  2  -  -  -  -  -
5       -  -  -  -  -  -  -  -  -  -
6       -  -  -  -  -  -  -  -  -  -
7       -  -  -  -  -  -  -  3  -  -
8       -  -  -  -  -  -  -  -  -  -
9       -  -  -  -  -  -  -  -  -  -
0 mines found, 20 mines left
Player Alice: 0 mines found
Player Bob: 0 mines found
```

```
Player Alice give the coordinates for the cell to open:
6 7
Mine found!
        0  1  2  3  4  5  6  7  8  9

0       -  -  -  -  -  -  -  -  -  -
1       -  -  -  -  -  -  -  -  -  -
2       -  -  -  -  -  -  -  -  -  -
3       -  -  -  -  -  -  -  -  -  -
4       -  -  -  -  2  -  -  -  -  -
5       -  -  -  -  -  -  -  -  -  -
6       -  -  -  -  -  -  -  *  -  -
7       -  -  -  -  -  -  -  3  -  -
8       -  -  -  -  -  -  -  -  -  -
9       -  -  -  -  -  -  -  -  -  -

1 mines found, 19 mines left
Player Alice: 1 mines found
Player Bob: 0 mines found

Player Bob give the coordinates for the cell to open:
7 6


        0  1  2  3  4  5  6  7  8  9

0       -  -  -  -  -  -  -  -  -  -
1       -  -  -  -  -  -  -  -  -  -
2       -  -  -  -  -  -  -  -  -  -
3       -  -  -  -  -  -  -  -  -  -
4       -  -  -  -  2  -  -  -  -  -
5       -  -  -  -  -  -  -  -  -  -
6       -  -  -  -  -  -  -  *  -  -
7       -  -  -  -  -  -  3  3  -  -
8       -  -  -  -  -  -  -  -  -  -
9       -  -  -  -  -  -  -  -  -  -

1 mines found, 19 mines left
Player Alice: 1 mines found
Player Bob: 0 mines found

Player Alice give the coordinates for the cell to open:
6 6
        0  1  2  3  4  5  6  7  8  9

0       -  -  -  -  -  -  -  -  -  -
1       -  -  -  -  -  -  -  -  -  -
2       -  -  -  -  -  -  -  -  -  -
3       -  -  -  -  -  -  -  -  -  -
4       -  -  -  -  2  -  -  -  -  -
5       -  -  -  -  -  -  -  -  -  -
6       -  -  -  -  -  -  5  *  -  -
7       -  -  -  -  -  -  3  3  -  -
8       -  -  -  -  -  -  -  -  -  -
9       -  -  -  -  -  -  -  -  -  -

1 mines found, 19 mines left
Player Alice: 1 mines found
Player Bob: 0 mines found
```

```
Player Bob give the coordinates for the cell to open:
7 6
Player Bob give the coordinates for the cell to open:
7 5
Mine found!
        0  1  2  3  4  5  6  7  8  9

0       -  -  -  -  -  -  -  -  -  -
1       -  -  -  -  -  -  -  -  -  -
2       -  -  -  -  -  -  -  -  -  -
3       -  -  -  -  -  -  -  -  -  -
4       -  -  -  -  2  -  -  -  -  -
5       -  -  -  -  -  -  -  -  -  -
6       -  -  -  -  -  -  5  *  -  -
7       -  -  -  -  -  *  3  3  -  -
8       -  -  -  -  -  -  -  -  -  -
9       -  -  -  -  -  -  -  -  -  -

2 mines found, 18 mines left
Player Alice: 1 mines found
Player Bob: 1 mines found
```