ΑΝΑΦΟΡΑ ΕΡΓΑΣΙΑΣ 1 (JAVA)

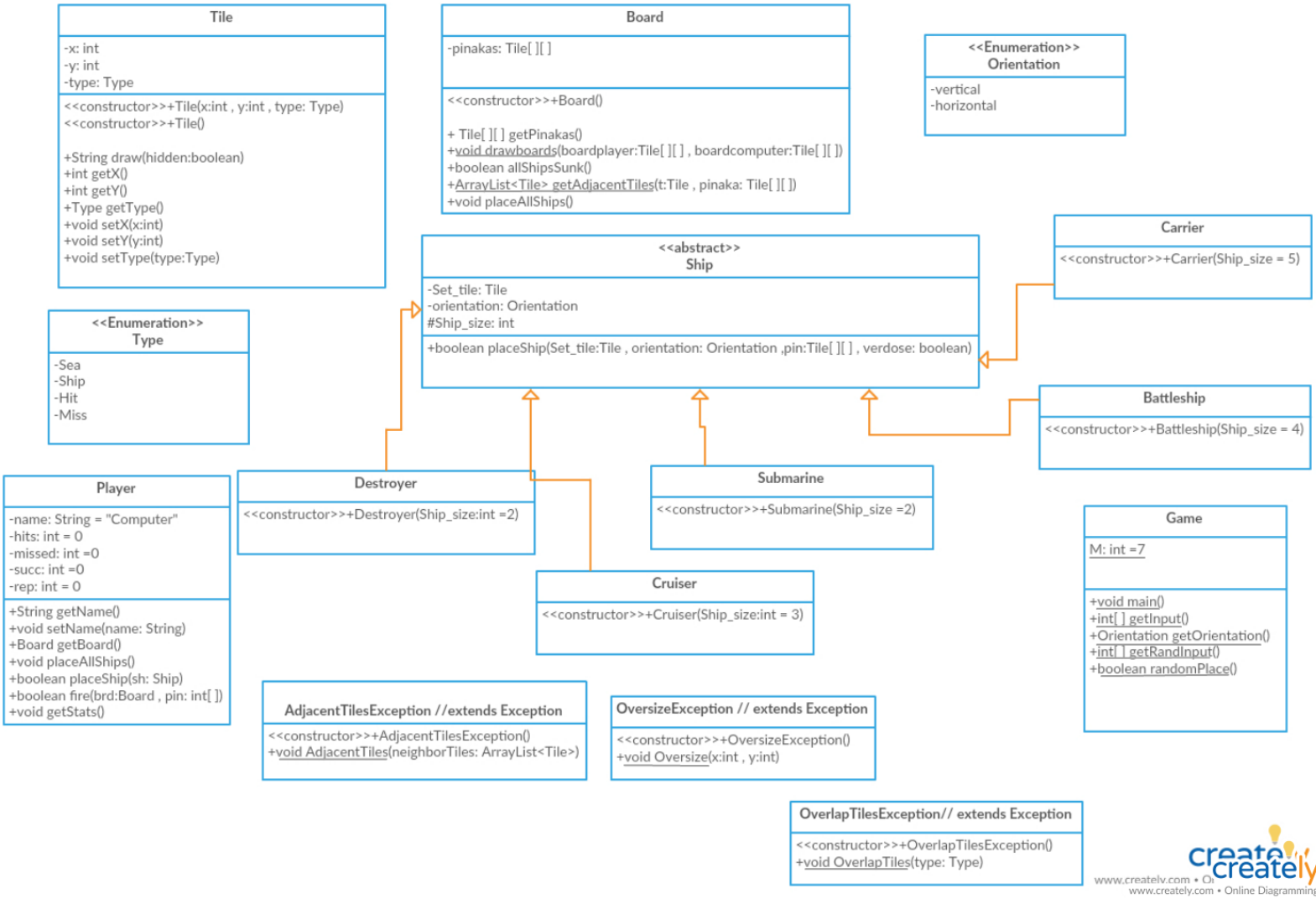
- **Στοιχεία των μελών την ομάδας:**

ΟΝΟΜΑΤΑ: Αριάδνη Μαχιά – Αθηνά Φουσέκη – Κατερίνα Δέρβου

Α.Μ.: 1059556 –1059623 – 1054185

EMAIL: up1059556@upnet.gr – up1059623@upnet.gr – up1054185@upnet.gr

- **Διάγραμμα κλάσεων (UML):**



- **Σύντομη περιγραφή της υλοποίησης:**
  - **Κλάση Tile:**

Η Tile είναι δημόσια κλάση στην οποία θα δημιουργούνται τα κελιά του πίνακα. Έχει τρεις ιδιωτικές (private) μεταβλητές, δύο ακέραιες: τις x και y που αποτελούν τις συντεταγμένες ενός κελιού (x=γραμμές και y=στήλες) και μία τύπου Type: την type που αποτελεί τον τύπου του κελιού. Η Type είναι δομή enum που περιλαμβάνει τις επιλογές: Sea, Ship, Hit και Miss.

Περιλαμβάνει δύο κατακευαστές (constructors), ένα χωρίς ορίσματα και με κενό σώμα και έναν με 3 ορίσματα (τις συντεταγμένες και τον τύπο του κελιού) με τον οποίο δίνουμε τιμές στις μεταβλητές της κλάσης. Επίσης, έχει μια μία δημόσια μέθοδο draw() που επιστρέφει αλφαριθμητικό (String) και έχει ένα όρισμα μία boolean μεταβλητή: την hidden, η οποία καθορίζει εάν θα κρύβει (αν είναι true) ή όχι (αν είναι false) τα κελιά τύπου Ship. Η μέθοδος αυτή με switch ελέγχει τον τύπου του κελιού και επιστρέφει το κατάλληλο σύμβολο-αλφαριθμητικό. Τέλος, η κλάση αυτή περιέχει getters και setters και για όλες τις ιδιωτικές μεταβλητές.

  - **Κλάση Ship:**

Η Ship είναι δημόσια αφηρημένη κλάση (abstract class) που δεν μπορεί να παράγει αντικείμενα και αναπαριστά το κάθε πλοίο. Έχει δύο private μεταβλητές: την Set_tile τύπου Tile (το κελί έναρξης) και την orientation τύπου Orientation (ο προσανατολισμό του πλοίου), και ακόμη μία protected ακέραια μεταβλητή την Ship_size (το μέγεθος του πλοίου). Η Orientation είναι δομή enum που περιλαμβάνει τις επιλογές: vertical και horizontal.

Περιλαμβάνει την δημόσια μέθοδο placeShip(), η οποία έχει ως σκοπό την τοποθέτηση των πλοίων. Επιστρέφει μία τιμή boolean ανάλογα με τον η τοποθέτηση ήταν επιτυχής (true) ή όχι (false) και τέσσερα ορίσματα: το κελί έναρξης, ο προσανατολισμός, ο πίνακας τύπου Board και μια Boolean μεταβλητή, την verbose, που καθορίζει εαν θα εμφανίζονται μηνύματα ή οχι. Μέσα σε αυτήν, οι Set_tile και orientation της κλάσης παίρνουν τις τιμές των αντίστοιχων ορισμάτων και δημιουργείται ένας τοπικός διδιάστατος πίνακας τύπου Tile που παίρνει τον πίνακα μέσω της μεθόδου getPinakas() από το αντικείμενο τύπου Board του ορίσματος. Έπείτα, μέσα σε ενα try block υπάρχει συνθήκη if που ελέγχει τον προσανατολισμό του ορίσματος και μέσα σε αυτήν μία επανάληψη for που καλεί, για όλα τα πιθανά κελιά τοποθέτησης πλοίου, τις μεθόδους των τριών εξαιρέσεων κλάσεων που εξετάζουν αν θα πετάξουν εξαίρεση ή οχι. Έτσι, ελέγχει εάν μπορεί να τοποθετηθεί όλο το πλοίο, πριν τοποθετηθεί. Εαν πετάξει εξαίρεση τοτέ αναλόγα με την εξαίρεση θα το πιάσει το κατάλληλο catch block, ανάλογα με την τιμή της verbose θα εκτυπωθεί μήνυμα ή οχι και θα επιστραφεί η τιμή false, δηλαδή ότι η τοποθέτηση απέτυχη, δηλαδή δεν τοποθετήθηκε κάποιο πλοίο. Στην περίπτωση που δεν υπάρξη κάποια εξαίρεση τότε με τις ίδιες συνθήκες if και μέσα τους τις ίδιες κατάλληλες for με πριν, αυτές οι for αυτή την φορά θα αλλάζουν τον τύπο όλο των κελιών (με την setter της μεταβλητής Type)

που θα καταλαμβάνει το πλοίο ανάλογα με το μέγεθος του και τέλος θα επιστρέφεται η τιμή true, δηλαδή ότι τοποθετήθηκε επιτυχώς το πλοίο.

- **Υποκλάσεις της Ship (Carrier, Battleship, Cruiser, Submarine, Destroyer):**

Η Ship έχει 5 δημόσιες υποκλάσεις που η κάθε μία αναπαριστά ένα διαφορετικό πλοίο. Κάθε μία από αυτές τις 5 κλάσεις είναι extends της Ship (επειδή είναι υποκλάση της και κληρονομεί από αυτήν) και μπορεί να παράγει αντικείμενα. Επίσης, κάθε μία περιέχει μόνο ένα κατασκευαστή χωρίς ορίσματα που αρχικοποιεί την μεταβλητή Ship_size της Ship με το κατάλληλο ακέραιο αριθμό που αποτελεί το μέγεθος του πλοίου. Η Carrier δίνει στην μεταβλητή την τιμή 5, η Battleship με 4, η Cruiser με 3, η Submarine με 3 και η Destroyer με 2.

- **Κλάσεις Εξαιρέσεων (OversizeException, OverlapException, AdjacentTilesException) :**

Και οι 3 κλάσεις είναι δημόσιες και είναι extends της κλάσης Exception της Java, επειδή αποτελούν ειδικές εξαιρέσεις που τις δημιουργήσαμε εμείς. Κάθε μία, περιέχει έναν κατασκευαστή χωρίς ορίσματα που στο σώμα του έχει μόνο ένα κενό super για να καλέσει τον default κατασκευαστή της υπερκλάσης. Ακόμα, περιέχουν μία μέθοδο που εξετάζει έαν πρέπει να πετάξει την εξαίρεση. Ειδικότερα:

Η OversizeException περιέχει μία static μέθοδο, την Oversize, που δεν επιστρέφει κάτι (void) , έχει δύο όρισμα: δύο ακέραιους που αντιπροσωπεύουν τις συντεταγμένες ενός κελιού του πίνακα και δηλώνει πως μπορεί να πετάξει (throws) εξαίρεση τύπου OversizeException. Σε αυτήν την μέθοδο υπάρχει μία συνθήκη if που ελέγχει εαν οι συντεταγμένες βρίσκονται εκτός των ορίων του πίνακα κι αν όντως ισχυεί αυτό τότε πετάει την εξαίρεση (throw new OversizeException();).

Η OverlapException περιέχει μία static μέθοδο, την OverlapTiles, που δεν επιστρέφει κάτι (void) , έχει ένα όρισμα: τύπου Type που αντιπροσωπεύει τον τύπο του κελιού και δηλώνει πως μπορεί να πετάξει (throws) εξαίρεση τύπου OverlapTilesException. Σε αυτήν την μέθοδο υπάρχει μία συνθήκη if που ελέγχει αν ο τύπος του κελιού είναι τύπου Type.Ship (δηλαδή εαν υπάρχει ήδη πλοίο στο κελί αυτό) κι αν όντως ισχυεί αυτό τότε πετάει την εξαίρεση (throw new OverlapTilesException();).

Η AdjacentTilesException περιέχει μία static μέθοδο, την AdjacentTiles, που δεν επιστρέφει κάτι (void) , έχει ένα όρισμα: τύπου ArrayList που περιέχει Tile, και αντιπροσωπεύει το ArrayList (δυναμικό πίνακα) που περιέχει γειτονικά κελιά και δηλώνει πως μπορεί να πετάξει (throws) εξαίρεση τύπου AdjacentTilesException. Σε αυτήν την μέθοδο περιέχει μια επανάληψη for που περιέχει μία συνθήκη if που ελέγχει εάν ο τύπος του i γειτονικού κελιού είναι τύπου Type.Ship (δηλαδή εαν υπάρχει πλοίο στο κελί αυτό) κι αν όντως ισχυεί αυτό τότε πετάει την εξαίρεση (throw new AdjacentTilesException();). Ο έλεγχος αυτός γίνεται για όλα τα γειτονικά κελιά χάρη της επανάληψης for.

- **Κλάση Board:**

Η Board δημιουργεί και έχει τον πίνακα του κάθε παίκτη. Έχει μία private μεταβλητή, την pinakas, που είναι δισδιάστατος πίνακας τύπου Tile με γραμμές και στήλες ίσες με την μεταβλητή M της κλάσης Game . Επίσης, έχει public getter μάθοδο για την μεταβλητή αυτή.

Από μεθόδους περιέχει:

Έναν μόνο κατασκευαστή χωρίς ορίσματα, οποίος με δύο εμφωλευμένες for loops (από το 0 έως και την M-1 της Game, δηλαδή μέγεθος του πίνακα και να αυξάνεται κατά 1) αρχικοποιεί τον πίνακα ώστε όλα τα κελιά να έχουν τις κατάλληλες συντεταγμένες (i, j) και να είναι τύπου Sea.

Την drawboards() που είναι static και έχει δύο ορίσματα τύπου δισδιάστατου πίνακα τύπου Tile, ένα που αντιστοιχεί τον παίκτη και ένα στον αντίπαλο-υπολογιστή. Περιέχει μια for loop (που αντιστοιχεί στις γραμμές των πινάκων) και μέσα της 2 for loops (η μία μετά την άλλη και η μία αντιστοιχεί στις στήλες του ενός πίνακα ενώ η άλλη στον άλλον) και μέσα σε κάθε μία κατάλληλες συνθήκες if ώστε να εκτυπωθούν τα κατάλληλα μηνύματα και οι πίνακες με την καταάλληλη μορφή. Στην συνθήκη που εκτυπώνεται το σύμβολο του κελιού καλείται η μέθοδος draw() της κλάσης Tile μέσω των μεταβλητών των ορισμάτων. Η draw() έχει όρισμα false όταν εκτυπώνεται ο πίνακας του παίκτη και true όταν εκτυπώνεται ο πίνακα του αντιπάλου, γιατί θέλουμε να αποκρύπτονται τα πλοία του.

Την allShipsSunk() που επιστρέφει μία boolean τιμή και δεν έχει ορίσματα. Έχει δύο εμφωνευμένες for loops (από το 0 έως και την M-1 της Game, δηλαδή μέγεθος του πίνακα και να αυξάνεται κατά 1) που μέσα στην δεύτερη for, υπάρχει μία συνθήκη if που ελέγχει εαν το κελί με συντεταγμένες (i,j) του πίνακα pinakas είναι τύπου Ship. Αν ισχυεί τότε επιστρέφει την τιμή false αλλιώς θα διατρέξει όλο τον πίνακα χωρίς να μπεί μέσα στην συνθήκη και θα επσιτρέψει true.

Την getAdjacentTiles() που επιστρέφει ArrayList<Tile> (δηλαδή ArrayList που περιέχει Tile) και έχει ένα όρισμα τύπου Tile. Ορίζονται 2 τοπικές ακέραιες μεταβλητές tempX και tempY και μία μεταβλητή τύπου ArrayList<Tile>. Έχει δύο εμφωνευμένες for loops (από το 0 έως και την M-1 της Game, δηλαδή μέγεθος του πίνακα και να αυξάνεται κατά 1). Μέσα στην δεύτερη for, η tempX παίρνει το αποτέλεσμα της διαφοράς της μεταβλητής X από το κελί του ορίσματος (μέσω getter) και του i, ενώ η tempY παίρνει το αποτέλεσμα της διαφοράς της μεταβλητής Y από το κελί του ορίσματος (μέσω getter) και του j. Παρατηρήσαμε ότι τα γειτονικά κελιά που ψάχνουμε έχουν χαρακτηριστικά αποτελέσματα διαφοράς με την συντεταγμένη X του ορίσματος και το i των γραμμών του πίνακα, αντίστοιχα για το Y και j. Πιο συγκεκριμένα, αν υπάρχει πάνω γειτονικό κελί τότε κάθε φορά το tempX=1 (γιατί το κελί του ορίσματος βρίσκεται σε γραμμή κατά 1 μεγαλύτερη από αυτή του από πάνω του) ενώ το tempY=0 (αφού είναι από πάνω του θα βρίσκεται στην ίδια στήλη με το κελί του ορίσματος). Αστίστοιχα, σκεφτήκαμε και για τα άλλα 3 κελιά. Έτσι, έπειτα, υπάρχει συνθήκη if που ελέγχει ένα ισχύει ένα από τα τέσσερα ζευγάρια τιμών των

tempX και tempY, κι αν ισχύει τότε το κελί αυτό προστίθεται στην ArrayList που δημιουργήσαμε. Αφού έχει γίνει προσπέλαση όλου πίνακα επιστρέφεται το ArrayList.

Την placeAllShips() που δεν επιστρέφει κάτι. Ορίζουμε ένα μονοδιάστατο πίνακα ακεραίων, rand, με μεγέθος 2(που θα περιέχει τις συντεταγμένες), μία μεταβλητή orientation τύπου Orientation (που θα αποθηκεύεται ο προσανατολισμός), μία boolean μεταβλητή success (που θα παίρνει την επιστρεφόμενη τιμή της placeShip και δείχνει εαν τοποθετήθηκε επιτυχώς το πλοίο) και ένα στιγμιότυπο Tile με αρχικοποιημένες συντεταγμένες (0,0) και τύπο Sea (που θα το χρησιμοποιήσουμε στην συνέχεια όταν καλέσουμε την placeShip). Στην συνέχεια δημιουρούνται 5 αντικείμενα (πλοία) ένα κάθε υποκλάσης της Ship. Έπειτα μέσα σε μία for loop (από το 0 έως και το 4 με βήμα αύξησης κατά 1, που υποδηλώνει τον αριθμό των πλοίων που έχουμε τοποθετηθεί) υπάρχει μία while loop που θα επαναλαμβάνεται όσο η success δεν είναι true, δηλαδή ώσπου να έχει τοποθετηθεί επιτυχώς ένα πλοίο. Μέσα της, καλείται η getRandInput() της Game και η rand παίρνει (ουσιαστικά αντιγράφει) τον επιστρεφόμενο πίνακα και μέσω των setter το κελί tile αλλάζει τις συντεταγμένες του σε αυτές του πίνακα rand. Ακόμα με μία if η orientation παίρνει τυχαίο προσανατολισμό, δηλαδή αν η ThreadLocalRandom.current().nextInt(0,2) επιστρέψει την τιμή μηδέν τότε ο προσανατολισμός είναι horizontal αλλίως είναι vertical. Η ThreadLocalRandom.current().nextInt(0,2) είναι μέθοδος της βιβλιοθήκης της java (import java.util.concurrent.ThreadLocalRandom;) και επιστρέφει τυχαίες ακέραιες τιμές από το 0 εώς και το 2-1=1. Μετά υπάρχει μία switch ανάλογα με το i της for πάει στο κατάλληλο case και καλείται η placeShip μέσω ενός απο τα αντικείμενα πλοίων που δημιουργήσαμε προηγουμένως με ορίσματα το τυχαίο tile, τον τυχαίο orientation, το this που υποδηλώνει το αντικείμενο Board που βρισκόμαστε και true, γιατί δεν θέλουμε να εκτυπώνονται μηνύματα και η success παίρνει την επιστρεφόμενη τιμή της. Έξω και μετά από την while αλλά μέσα στην for αρχικοποιείται πάλι το success με false.

- **Κλάση Player:**

Η Player συμβολίζει του παίκτες. Έχει private μεταβλητές: την name είναι τύπου String (και αποτελεί το όνομα του παίκτη), 4 ακέραιες μεταβλητές: την hits που είναι οι συνολικές βολές, την missed που είναι ο αριθμός των αστοχιών, την succ που είναι οι επιτυχημένες βολές και την rep που είναι ο αριθμός των επαναλήψεων και ένα αντικείμενου board τύπου Board.

Από μεθόδους περιέχει:

Getter και setter για την name και getter για την board .

Την placeAllShips() που δεν έχει ορίσματα και μέσω της board καλεί την placeAllShips() της Board.

Την placeShip() που έχει ένα όρισμα τύπου Ship, καλούμε την getInput της Game και αυτό που επιστρέφει το αποθηκεύουμε σε ένα τοπικό πίνακα, τον coo, μεγέθους 2. Έπειτα, δημιουργείται ένα αντικείμενο Tile, το Set_Tile, με συντεταγμένες του ακέραιους της coo και τύπου Sea, και μέσω του ορίσματος καλείται η placeShip με

ορίσματα το Set_Tile, ότι επιστραφεί απο την getOrientation() της Game, την board και false, γιατί θέλουμε να εμφανίζονται μηνύματα. Τέλος, την τιμή boolean που επιστρέφει η placeShip αποθηκεύεται σε μια μεταβλητή και αυτή επιστρέφεται.

Την fire που επιστρέφει μία boolean τιμή και έχει δύο ορίσματα μία μεταβλητή τύπου Board, την brd, και έναν ακέραιο μονοδιάστατο πίνακα pin. Ορίζονται 2 τοπικές μεταβλητές x και y που παίρνουν τις τιμές από το pin και έναν δισδιάστατο πίνακα τύπου Tile, τον pinakas, που παίρνει τον πίνακα της brd μέσω της getPinakas(). Μέσα σε ένα try block καλείται η Oversize της OversizeException με όρισμα τα x και y κι υπάρχει ένα catch block που εκτυπώνει κατάλληλο μήνυμα και επιστρέφει false, δηλαδή ότι απέτυχε η επίθεση. Έπειτα, υπάρχει συνθήκη if αν το κελί είναι τύπου Sea το κελί αλλάζει τύπο σε Miss, αυξάνοται οι hits και missed κατά 1 και εκυπώνεται το καταάλληλο μήνυμα, αν το κελί είναι τύπου Ship το κελί αλλάζει τον τύπο του κελιού σε Hit, αυξάνονται οι hits και succ κατά 1 και εκτυπώνεται το κατάλληλο μήνυμα, αν το κελί είναι τύπου Miss αυξάνονται οι missed , rep και hits κατά 1 και εκτυπώνεται το κατάλληλο μήνυμα, διαφορετικά (δηλαδή αν είναι τύπου Hit) αυξάνονται οι rep και hits και εκτυπώνεται το κατάλληλο μήνυμα. Τέλος επιστρέφει true.

Την getStats που δεν επιστρέφει κάτι και δεν έχει ορίσματα. Σε αυτή εκτυπώνεται με κατάλληλα μηνύματα τις hits, missed, succ και rep.

- **Κλάση Game:**

Η Game είναι η κεντρική κλάση, η οποία περιλαμβάνει και την main. Έχει μία static final ακέραια μεταβλητή , M, αρχικοποιημένη με 7 που αποτελεί το μέγεθος του πίνακα και μια static μεταβλητή input που είναι Scanner και δέχεται τιμές από το πληκτρολόγιο

Από μεθόδους περιέχει:

Την main που είναι static και δεν επιστρέφει κάτι. Αυτή ορίζεται το αλφαριθμητικό answer, 6 boolean μεταβλητές: την success (που υποδηλώνει εάν είναι επιτυχής η τοποθέτηση πλοίου), τις ship1, ship2, ship3, ship4, ship5 (που υποδηλώνει εαν έχει τοποθετηθεί το πλοίο) και όλες ειναι αρχικοποιημένες με false. Επίσης, έχει δύο ακέραιες μεταβλητές τηνς shots που είναι οι συνολικές βολές και είναι αρχικοποιημένη με 0, την maxShots που αποτελεί τον μέγιστο αιρθμό βολών που μπορεί να γίνουν μέχρι να τερματιστεί το παιχνίδι και έναν μονοδιάστατο πίνακα ακεραίων, fireShots , μεγέθους 2 που αποθηκεύει τις συντεταγμένες για επίθεση. Δημιουργόνται δύο αντικείμενα Player: user και computer, μετά ζητείται το όνομα του παίκτη και αλλάζει το name του παίκτη μέσω setter με ότι εισαχθεί από το πληκτρολόγιο. Καλείται η drawboards (χωρίς αντικείμενο γιατί είναι static) με ορίσματα τους πίνακες του user και computer που καλούνται μέσω των αντικείμένων Player που καλούν την getBoard() και η επιστρεφόμενη Board καλεί την getPinakas(). Μετά, καλείται η placeAllShips() μέσω της computer και καλείται η randomPlace() της Game κι αν επιστρέχει true, τότε καλείται η placeAllShips() του user και καλείται η drawBoards(). Διαφορετικά, ο χρήστης θα τοποθετήσει μόνο τους τα πλοία και ανάλογα με το τι θα δώσει ο χρήστης από το πληκτρολόγιο με μια switch μπαίνει στην ανάλογη case αν η μεταβλητή

ship(1,2,3,4ή5) είναι false τότε δημιουργείται αντικείμενο μιάς υποκλάσης της Ship και καλείται η placeShip με όρισμα το αντικείμενο αυτό κι αν true (δηλαδή τοποθετήσει το πλοίο) τότε η success γίνεται true, καλείται η drawboards και η ship γίνεται true αλλιώς success γίνεται false. Αυτό θα επαναλαμβάνεται μέχρι η success να γίνει true. Μετά ο χρήστης θα πρέπει να ξαναδώσει αλφαριθμητικό απο το πληκτρολόγιο και θα ξαναμπαίνει στην switch κι όλο αυτό θα επαναλαμβάνεται μέχρι όλες οι μεταβλητές ship(1,2,3,4 και 5) είναι true. Στη συνέχεια, μέσα σε μια while loop που επαναλαμβάνεται μέχρι βυθιστούν όλα τα πλοία του user ή του computer ή οι συνολικές βολές ξεπεράσουν τις maxShots. Σε αυτήν καλείται η getInput() της Game και ο επιστρεφόμενος πίνακας αποθηκεύεται στην fireShots που χρησιμοποιείται ως όρισμα μαζί με την board του computer, στην συνέχεια που καλείται η fire μεσω του user. Αυτό επαναλαμβάνεται μέχρι η fire να επιστρέψει true με μια do while loop. Ύστερα αυξάνονται οι συνολικές βολές (shots), καλείται η getRandInput() που ό,τι επιστρέψει αποθηκεύται στην fireShots και καλείται η fire μέσω του computer με τα αντίστοιχα ορίσματα και μετά καλείται πάλι η drawboards(). Έπιετα με μια συνθήκη if – else if – else ανάλογα με ποιανού πίνακα κι αν βυθίστηκαν όλα τα πλοία, εκτυπώνονται τα κατάλληλα μηνύματα για τον νικητή και μετα καλείται η getStats() κάθε αντικειμένου Player.

Την getInput() που είναι static, δεν έχει ορίσματα και επιστρέφει μονοδιάστατο πίνακα ακεραίων. Ο χρήστης εισάγει δύο ακέραιες τιμές από το πληκτρολόγιο και αυτές αποθηκεύονται σε ένα μονοδιάστατο πίνακα ακεραίων μεγέθους 2, ο οποίος και επιστρέφεται.

Την getOrientation() που είναι static, δεν έχει ορίσματα και επιστρέφει Orientation.Μέσα σε μια do while δέχεται έναν χαρακτήρα από το πληκτρολόγιο, με μια συνθήκη if ελέγχει εαν ο χρήστης έδωσε το γράμμα “h” για οριζόντιο προσανατολισμό, κάνει το a ίσο με 2 και επιστρέφει Orientation.horizontal ή “v” για κατακόρυφο, κάνει το a ίσο με 2 και επιστρέφει Orientation.vertical, διαφορετικά το a γίνεται 1 και επαναλαμβάνεται όλο αυτό μέχρι να δώσει δεχτεί τιμή.

Την getRandInput() που είναι static, δεν έχει ορίσματα και επιστρέφει μονοδιάστατο πίνακα ακεραίων. Καλείται η μέθοδος ThreadLocalRandom.current().nextInt(0,M) από βιβλιοθήκη της Java (import java.util.concurrent.ThreadLocalRandom;) η οποία επιστρέφει μία τυχαία τιμή από το 0 έως και το M-1. Αυτό γίνεται δύο φορές και οι τυχαίες τιμές αποθηκεύονται σε έναν μονοδιάστατο πίνακα ακεραίων μεγέθους 2, ο οποίος και επιστρέφεται στο τέλος.

Την randomPlace() που είναι static, δεν έχει ορίσματα και επιστρέφει μία τιμή boolean. Δέχεται έναν χαρακτήρα από το πληκτρολόγιο (χρησιμοποιώντας, φυσικά, την μεταβλητή input που είναι Scanner) με if –else if – else εξετάζει εάν ήταν το “y” και τότε επιστρέφει true, εαν ήταν το “n” και τότε επιστρέφει false ή διαφορετικά εκτυπώνει ένα μήνυμα. Αυτό επαναλαμβάνεται μέχρι να δωθεί “y” ή “n” γιατί υπάρχει do while loop που θα επαναλαμβάνεται μέχρι η τιμή boolean μιας τοπικής μεταβλητής να μην είναι false.