

**ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ**

**ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ**

**ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ, ΥΠΟΛΟΓΙΣΤΩΝ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ**

**ΓΕΝΝΗΤΡΙΑ ΕΠΙΤΡΑΠΕΖΙΩΝ ΠΑΙΧΝΙΔΙΩΝ**

**Πρώτη εβδομάδα**

**Ομαδική Εργασία:**

Αναστασιάδης Αλκίνοος (20003)

Ζήνα Ελένη (20046)

Λαγιόκαπας Δημήτριος (20079)

Μακρή Στυλιανή (20060)

Επιβλέποντες:

Κουρέας Αργύριος

Λάντζος Θεόδωρος

**ΣΕΡΡΕΣ****, 27 έως 31 ΜΑΡΤΙΟΥ 2023**

Περιεχόμενα

[Περίληψη 3](#_Toc131117495)

[Εισαγωγή 4](#_Toc131117496)

[1. Μεθοδολογία 5](#_Toc131117497)

[2. Υλοποίηση 5](#_Toc131117498)

[3. Χρονοδιάγραμμα 9](#_Toc131117499)

[4.Αποτελέσματα-Επίλογος 11](#_Toc131117500)

[Βιβλιογραφία 12](#_Toc131117501)

# Περίληψη

Σκοπός της εργασίας ήταν η ομαδική υλοποίηση μιας γεννήτριας «επιτραπέζιων» παιχνιδιών με χρήση της γλώσσας προγραμματισμού Java. Η εφαρμογή θα πρέπει να δίνει τη δυνατότητα να διαβάζει μια περιγραφή του παιχνιδιού σε μορφή JSON και στη συνέχεια να δημιουργεί το παιχνίδι. Για αρχή δημιουργήσαμε ένα πολύ απλό

παιχνίδι (τύπου φιδάκι) με ταμπλό 40 τετραγώνων (tiles) και 4 παίκτες όπου ένας από τους οποίους είναι νικητής του παιχνιδιού, όταν φτάσει πρώτος στο τερματισμό με βάση κάποιους κανόνες.

# Εισαγωγή

Με την αύξηση της δημοτικότητας των βιντεοπαιχνιδιών την τελευταία περίοδο, το game development είναι μια περιζήτητη δεξιότητα. Παρ’ όλ’ αυτά, η ανάπτυξη βιντεοπαιχνιδιών είναι μια περίπλοκη και χρονοβόρα διαδικασία που απαιτεί γνώσεις προγραμματισμού, graphic design κλπ. Αυτό συχνά το κάνει δύσκολο για τους μη-έμπειρους ή τους πρωτάρηδες να φτιάξουν τα δικά τους παιχνίδια.

Για να λυθεί αυτό το πρόβλημα, αναπτύχθηκαν οι γεννήτριες παιχνιδιών έτσι ώστε να αυτοματοποιηθεί κάποιο κομμάτι της ανάπτυξης παιχνιδιών. Η γεννήτρια που θα αναπτύξουμε στη διάρκεια αυτού του εξαμήνου είναι ένα τέτοιο εργαλείο. Στόχος του προγράμματος μας είναι η γεννήτρια να διαβάζει την περιγραφή του παιχνιδιού μέσα από ένα json αρχείο και να δημιουργεί ένα επιτραπέζιο παιχνίδι.

Για να πετύχουμε τους στόχους αυτής της εργασίας, πρέπει να απαντήσουμε στα παρακάτω ερωτήματα:

* Ποια είναι τα κύρια χαρακτηριστικά ενός επιτραπέζιου παιχνιδιού;
* Πόσο αποτελεσματική είναι η γεννήτρια στο να δημιουργεί playable παιχνίδια από τις περιγραφές των json αρχείων;
* Πως θα επαναχρησιμοποιήσουμε τον κώδικα για άλλα projects;
* Πώς συγκρίνεται η γεννήτρια που φτιάξαμε με την παραδοσιακή μέθοδο ανάπτυξης παιχνιδιών με βάση την ευκολία χρήσης, την ποιότητα των παιχνιδιών που δημιουργούνται, και την ικανοποίηση του χρήστη;

# 1. Μεθοδολογία

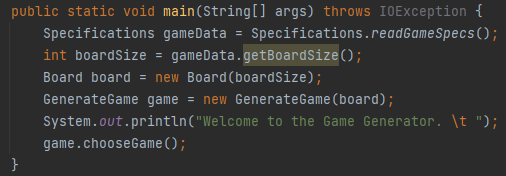
Για να δημιουργήσουμε την αρχική μορφή της γεννήτριας, χρησιμοποιήσαμε ένα συνδυασμό εργαλείων και λογισμικού, συμπεριλαμβανομένου του IntelliJ IDEA ως το περιβάλλον ανάπτυξης του προγράμματος και της βιβλιοθήκης JSON.

Αρχικά, χρησιμοποιήσαμε το IntelliJ IDEA ως το ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment – IDE) για να γράψουμε τον Java κώδικα που σχηματίζει τον κορμό του προγράμματος. Επιλέξαμε το IntelliJ IDEA λόγω των εύκολων στη χρήση χαρακτηριστικών του που μας επέτρεψε να γράψουμε έναν αποδοτικό καλογραμμένο κώδικα. Το JSON είναι ένας τρόπος αναπαράστασης δεδομένων δομημένος σε ένα αρχείο, έτσι ώστε με κατάλληλη μέθοδο να παίρνουμε τα δεδομένα που εμπεριέχονται και να τα κάνουμε επαναχρησιμοποίηση σε πολλά προγράμματα. Γι’ αυτό το λόγο, τα αποθηκεύσαμε όλα τα χαρακτηριστικά σε ένα αρχείο για να έχουμε πρόσβαση στην περιγραφή των παιχνιδιών με ευκολία.

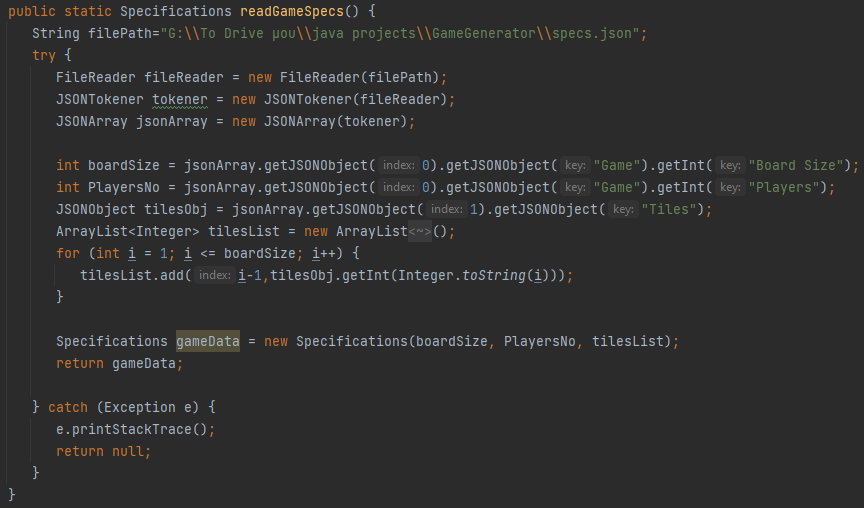
# 2. Υλοποίηση

Το πρόγραμμα περιέχει μια κύρια κλάση, την GenerateGame και τις Board, Player και Specifications (σε αυτό το στάδιο).

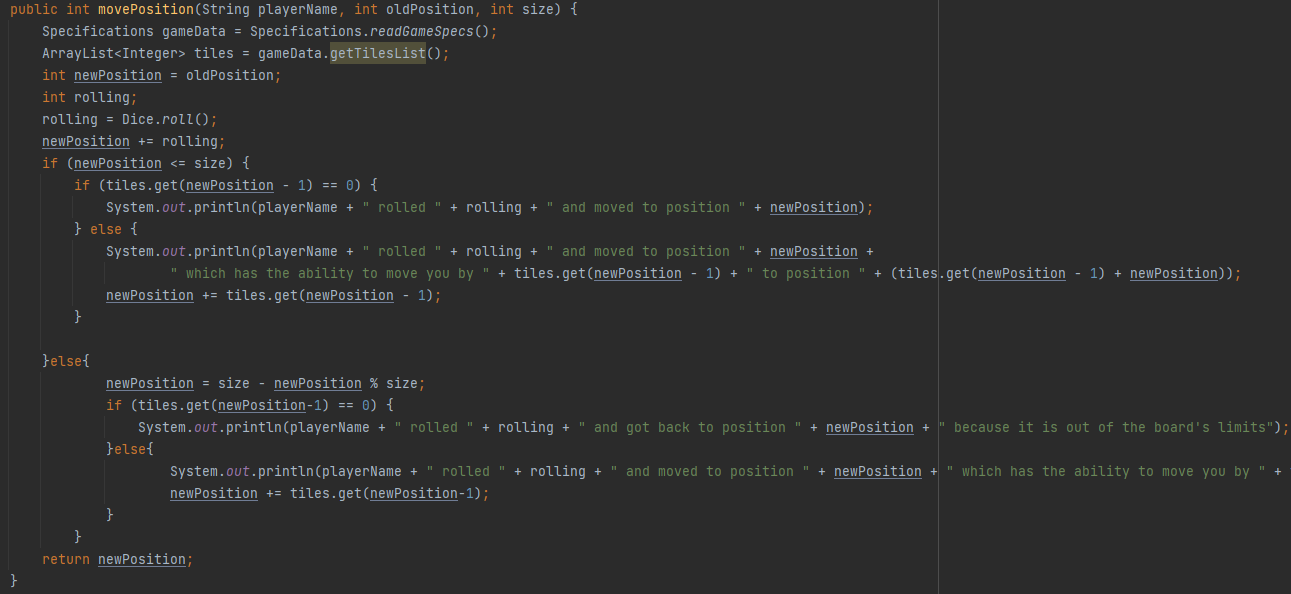
Η συνάρτηση main() βρίσκεται στην κλάση GenerateGame και σ’ αυτήν δημιουργείται το ταμπλό με τα χαρακτηριστικά που δώσαμε από το specs.json .



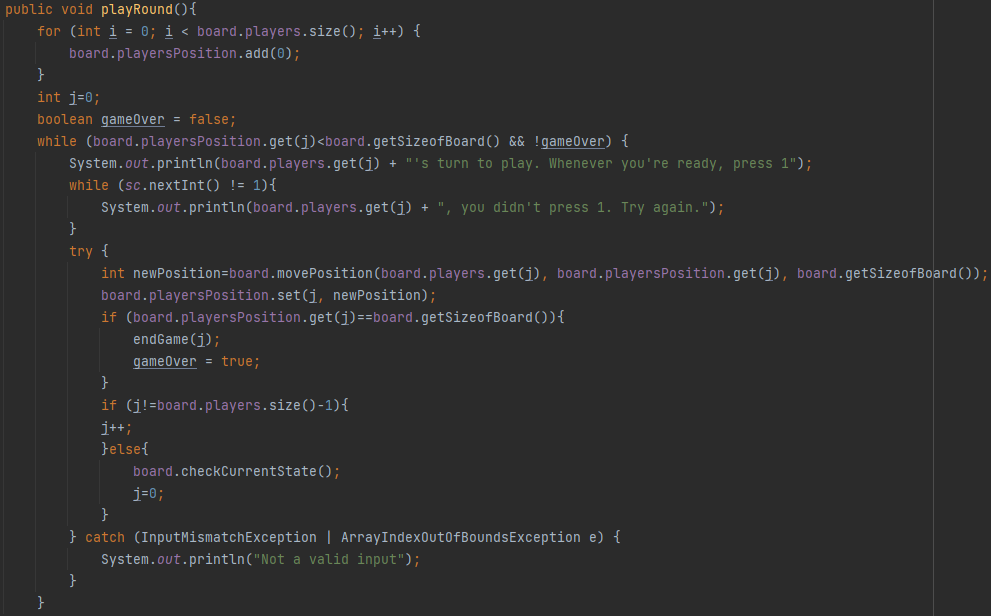
Specifications είναι η κλάση, μέσα στην οποία γίνεται το «διάβασμα» των δεδομένων μέσω της συνάρτησης readGameSpecs(), η οποία επιστρέφει αντικείμενο, ώστε να πάρουμε μεμονωμένα τα χαρακτηριστικά που θέλουμε με getters.



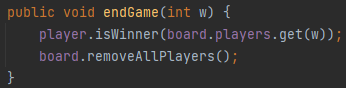
Η συνάρτηση chooseGame() καλείται μέσα στην main() για να επιλεχθεί το παιχνίδι και ανάλογα την επιλογή, καλείται η συνάρτηση startGame(), η οποία ξεκινάει το παιχνίδι. Μέσα στην startGame(), καλείται η addPlayers() της Board, η οποία ορίζει τους παίκτες ανάλογα με το πλήθος που δίνει το specs.json και έπειτα καλείται η συνάρτηση playRound(), η οποία κάνει και την περισσότερη «δουλειά» από όλο το πρόγραμμα. Σ’ αυτήν υλοποιούνται όλες οι κινήσεις των παικτών με τη βοήθεια κι άλλων συναρτήσεων, όπως η movePosition() για την ανανέωση της θέσης τους και η checkCurrentState() για την ενημέρωση των θέσεων τους μέσω μηνυμάτων.

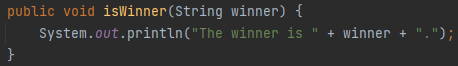


Ξεκινάει το παιχνίδι με τους παίκτες να ρίχνουν ζάρι με τη σειρά. Αν καταλήξουν σε τετράγωνο, το οποίο έχει κάποιο νούμερο που θα τους πάει μπροστά ή πίσω, εμφανίζεται κατάλληλο μήνυμα με την τιμή του τετραγώνου και αυτό αθροίζεται στην θέση στην οποία βρίσκεται για να πάει σε νέα θέση. Παρόμοια λογική υπάρχει κι αν δεν έχει κάποια τιμή (δηλαδή 0), αλλά τότε δεν κάνουμε κάποια επιπλέον πρόσθεση, κρατάμε μόνο την αρχική με τον αριθμό του ζαριού.

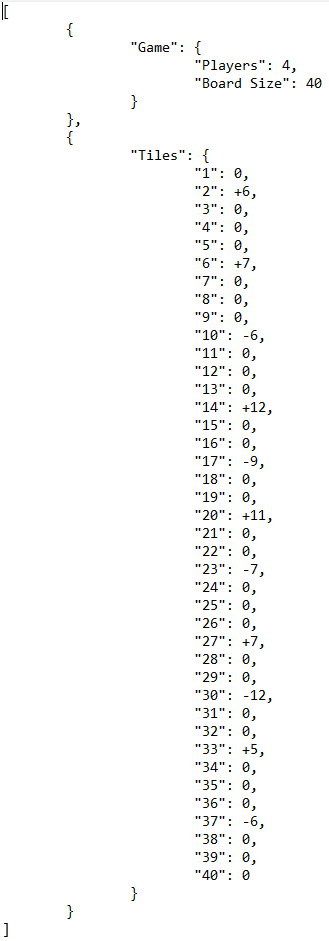


Το σημαντικότερο όμως που συμβαίνει είναι το εξής: έχουμε ορίσει σαν κανόνα ότι αν δεν τερματίσει ένας παίκτης, ρίχνοντας συγκεκριμένο αριθμό ζαριού ακριβώς στο τελικό τετράγωνο τότε θα γυρνάει τόσες θέσεις πίσω μέχρι να πετύχει 40, στην προκειμένη περίπτωση. Μόλις το πετύχει αυτό, καλείται η endGame() από την playRound(), η οποία παίρνει την θέση του παίκτη από την λίστα των παικτών για να εμφανίσει το όνομά του, τον καθορίζει νικητή και αφαιρεί όλους τους παίκτες από το παιχνίδι.





Μορφή αρχείου .json:



# 3. Χρονοδιάγραμμα

Διαχωρισμός εργασιών:

Αλκίνοος: Με τα νούμερα που παίρνουμε για τα Tiles (από το specs.json στο ArrayList<Integer> tilesList), προσδιορίζουμε ουσιαστικά τα φίδια (- βήμα) και τις σκάλες (+ βήμα). Αυτά στέλνονται στη συνάρτηση movePosition() της κλάσης Board, η οποία προσαρμόθηκε, και έτσι προστίθενται μετά την κίνηση του παίκτη, εφόσον η θέση στην οποία σταμάτησε, είχε κάποια τιμή και τον μετακινεί ανάλογα. Η Java αναγνωρίζει προσημασμένους αριθμούς και γίνονται οι προσθέσεις κανονικά.

Δημήτρης: Υλοποιήθηκε η συνάρτηση endGame() της κλάσης GenerateGame και η isWinner() της κλάσης Player που καλείται μέσα σ’ αυτήν, για να βγάζει τον νικητή. Προστέθηκε στη main() ένα case, το οποίο δίνει την δυνατότητα στον χρήστη να διαλέγει 1 από τα 2 παιχνίδια (επειδή δεν υπάρχει ακόμα δεύτερο παιχνίδι, θα είναι απλά ένα break;).

Στέλλα: Υλοποιήθηκε η συνάρτηση checkCurrentState() στην κλάση Board, η οποία εκτυπώνει σε μήνυμα την κατάσταση όλων των παικτών, δηλαδή σε ποια Tiles βρίσκονται. Αυτή η συνάρτηση καλείται μετά απο κάθε γύρο που θα παίζουν οι παίκτες εκτός απο την περίπτωση ότι κάποιος τερμάτισε.

Ελένη: Έδωσε τον αρχικό απλό κώδικα (χωρίς specs.json) έτοιμο και εκτελέσιμο στα υπόλοιπα μέλη της ομάδας για να τον επεκτείνουν με βάση τις εργασίες που τους δόθηκαν. Διόρθωσε τον τρόπο εισαγωγής των δεδομένων του κώδικα από το specs.json κι έτσι η ομάδα εξέλιξε τον κώδικα.

26/3/2023:

Ελένη: Δοθήκαν εργασίες στο κάθε μέλος της ομάδας με διορία μέχρι την Τετάρτη.

28/3/2023:

Στέλλα: Υλοποιήθηκε η συνάρτηση checkCurrentState() που της δόθηκε.

Ελένη: Βοήθησε την ομάδα και αυτοί με τη σειρά τους για την επίλυση προβλημάτων που προκλήθηκαν.

Αλκίνοος : Υλοποίησε τον κώδικα που του δόθηκε για να γίνονται οι ανάλογες μετακινήσεις και προσάρμοσε την movePosition() με αυτές τις προδιαγραφές.

Δημήτρης: Υλοποίησε τα τμήματα κώδικα που του δόθηκαν σχετικά με την επιλογή του παιχνιδιού και τον τερματισμό του.

29/3/2023:

Αλκίνοος : Επεξεργάστηκε ξανά τον κώδικα που έκανε στις 28/3/2023 έτσι ώστε να είναι ξεχωριστή συνάρτηση στην κλάση Tiles η οποία λέγεται movePlayer και καλείται στην συνάρτηση movePosition της κλάσης Board για να διαχειριστεί το position του παίκτη σε περίπτωση που πατήσει σε τετράγωνο με σκάλα ή φίδι. Τελικά, όμως δε χρειάστηκε, γιατί θα είχαμε δύο συναρτήσεις με ίδιο σκοπό, οπότε κρατήσαμε την movePosition(), αλλά ανανεωμένη.

Ομάδα: Τελείωσε και η τελευταία υλοποίηση που έπρεπε να γίνει.

30/3/2023:

Ομάδα: Ένωση των υλοποιήσεων σε ένα πρόγραμμα και διόρθωση λαθών.

Στέλλα: Σύνταξη του report.

Ελένη: Επεξεργάστηκε το report και το έφερε στην τελική του μορφή.

# 4.Αποτελέσματα-Επίλογος

Επιλογικά, την πρώτη εβδομάδα υλοποιήσαμε ένα απλό πρόγραμμα όπου δημιουργεί ένα παιχνίδι όπως το «Φιδάκι». Τις επόμενες εβδομάδες το πρόγραμμά μας θα επεκταθεί με νέες λειτουργίες και κατά πάσα πιθανότητα προσθήκη νέου παιχνιδιού.

# 

# Βιβλιογραφία

|  |  |
| --- | --- |
| [1] | Information for method JSON  <https://github.com/stleary/JSON-java> |
| [2] | How to Create JSON File  <https://codebeautify.org/blog/how-to-create-json-file/> |
| [3] | Definition of Switch Java  <https://www.javatpoint.com/java-switch> |
| [4] | Converter from TXT to JSON File  <https://mconverter.eu/convert/txt/json/> |
| [5] | JSONTokener Examples Guide  <https://www.tabnine.com/code/java/classes/org.json.JSONTokener> |
|  |  |