1η ΕΡΓΑΣΙΑ ΣΤΟ ΜΑΘΗΜΑ ΤΗΣ ΤΕΧΝΗΤΗΣ ΝΟΗΜΟΣΥΝΗΣ

Οι φοιτητές:

- Κοτίτσας Σωτήριος Α.Μ. p3150077
- Χατζόπουλος Γεώργιος Α.Μ. p3150196
- Λεπίπας Αναστάσιος Α.Μ. p3150091

Στο παρόν Project μας ανατέθηκε η κατασκευή ωρολογίου προγράμματος μαθημάτων σε ένα γυμνάσιο για κάθε ένα από τα 3 τμήματα που έχει αντίστοιχα η α,β και γ γυμνασίου, χρησιμοποιώντας μεθόδους αναζήτησης σε χώρους καταστάσεων.

Στο project χρησιμοποιήθηκαν οι εξής κλάσεις:

1. LessonsClass.java
2. TeachersClass.java
3. LessonsClassParser.java
4. TeachersClassParser.java
5. State.java
6. GeneticAlgorithmClass.java
7. Population.java
8. Main.java

Επίσης να αναφερθεί σε αυτό το σημείο ότι χρησιμοποιήθηκαν .json αρχεία που περιέχουν τα δεδομένα που είναι απαραίτητα για τη υλοποίηση του προγράμματος. Τα αρχεία είναι 2. Το πρώτο είναι το Lessons.json που περιλαμβάνει τα ονόματα των μαθημάτων που έχει η κάθε τάξη με την εξής διάρθρωση:

1) κωδικός μαθήματος (course code)

2) όνομα μαθήματος (course name)

3) τμήμα που διδάσκεται (classroom)

4) αριθμός των ωρών που διδάσκεται (hours)

**Κάθε μάθημα π.χ. μαθηματικά έχει τοποθετηθεί 3 φορές σε μία τάξη(π.χ. α’ γυμνασίου) με διαφορετικό course code ένα για κάθε τμήμα (A1,A2,A3). Οπότε επειδή τα μαθηματικά είναι μάθημα που διδάσκεται υποχρεωτικά και στις 3 τάξεις θα έχουμε διαφορετικό κωδικό για κάθε μάθημα μαθηματικών άρα θα έχουμε 3 φορές σε κάθε τάξη ,συνολικά 9 φορές.

Το δεύτερο json αρχείο είναι το Teachers.json το οποίο περιλαμβάνει τα ονόματα των καθηγητών και έχει την εξής διάρθρωση:

1. Κωδικός καθηγητή (teacher code)
2. Όνομα καθηγητή (teacher name)

3. Κωδικός μαθήματος (course code)
4. Μέγιστες ώρες διδασκαλίας ανά βδομάδα(max hours per week)
5. Μέγιστες ώρες κάθε μέρα(max hours per day)

** Για κάθε μάθημα έχουμε βάλει να διδάσκεται και από διαφορετικό καθηγητή σε κάθε τάξη (π.χ. έναν μαθηματικό σε όλη την α γυμνασίου, έναν σε όλη την β γυμνασίου, έναν σε όλη την γ γυμνασίου). Σε μαθήματα όμως που διδάσκονται λίγες ώρες όπως π.χ. μουσική έχουμε 1 καθηγητή για όλο το σχολείο.

Προκειμένου να μπορεί το πρόγραμμά μας να αναγνωρίζει τα json files θα χρειαστεί να χρησιμοποιήσουμε ένα json jar το όποιο περιλαμβάνει και το json object όπου μετατρέπει σε object κάθε πλειάδα μαζί με τα στοιχεία της. Π.χ. Ένα json object θα είναι ένα μάθημα μαζί με τις ώρες, την τάξη που διδάσκεται ο κωδικός του , το όνομά του.

Το Jar που χρησιμοποιήθηκε είναι στον εξής σύνδεσμο:

http://grepcode.com/snapshot/repo1.maven.org/maven2/org.glassfish/javax.json/1.0.3

Και το κατεβάζουμε πατώντας binary download και στην συνέχεια πρέπει να γίνει include στο project μας για να αναγνωρίζει τα json files.

ΑΝΑΛΥΣΗ ΚΛΑΣΕΩΝ:

1. **LESSON CLASS PARSER (LessonClassParser.java)**

   Αυτή η κλάση είναι κατασκευασμένη έτσι ώστε να κάνει parse και περνάει ένα ένα τα μαθήματα από το json file Lessons.json και να τα αποθηκεύει σε ένα Array List τύπου LessonClass (θα αναλυθεί παρακάτω). Αν δούμε πιο προσεκτικά θα παρατηρήσουμε ότι υπάρχουν 2 constructors ,ένας κενός και ένας που δέχεται ένα Json file. Στον δεύτερο λοιπόν βλέπουμε ότι δημιουργεί ένα json array το οποίο περιέχει τα μαθήματα και στοιχεία του κάθε μαθήματος όπως αυτά διαβάστηκαν από το json file που δώσαμε στο πρόγραμμα μας. Αφού περαστούν όλα τα μαθήματα στο json array αντιγράφεται κάθε Object από του LessonsArray στο lessons όπου εν τέλει θα χρησιμοποιηθεί στο πρόγραμμά μας. Τέλος, υπάρχουν 2 ακόμα μέθοδοι getLessons όπου μας επιστρέφει το array list lessons που περιέχει τα μαθήματά μας και μία setCourses η οποία αλλάζει το linked list των μαθημάτων.

2. **LESSONS CLASS (LessonsClass.java)**

Αυτή η κλάση αναφέρθηκε και προηγουμένως, και η δουλεία που κάνει είναι να δέχεται Json objects που αφορούν το μάθημα και να «σπάει» ουσιαστικά το Json Object του μαθήματος στα διαφορετικά πεδία απο τα οποία αποτελείται, δηλαδή σε:

- Course_code -> κωδικός μαθήματος
- Course_name -> όνομα μαθήματος
- Classroom - > τάξη που γίνεται το μάθημα
- Hours -> Ώρες διδασκαλίας του μαθήματος

Εκτός από τους 2 constructors έναν κενό και έναν που κάνει την δουλεία που ειπώθηκε πιο πάνω η κλάση αυτή διαθέτει getters και setters για κάθε πεδίο του object που διαβάζει.

3. **TEACHERS CLASS PARSER (TeachersClassParser.java)**

Αυτή η κλάση είναι κατασκευασμένη έτσι ώστε να κάνει parse και περνάει ένα ένα τα μαθήματα από το json file Tecahers.json και να τα αποθηκεύει σε ένα Array List τύπου TeachersClass (θα αναλυθεί παρακάτω). Αν δούμε πιο προσεκτικά θα παρατηρήσουμε ότι υπάρχουν 2 constructors ,ένας κενός και ένας που δέχεται ένα Json file. Στον δεύτερο λοιπόν βλέπουμε ότι δημιουργεί ένα json array με το όνομα professorsArray το οποίο περιέχει τα μαθήματα και στοιχεία του κάθε μαθήματος όπως αυτά διαβάστηκαν από το json file που δώσαμε στο πρόγραμμα μας. Αφού περαστούν όλα τα μαθήματα στο json array αντιγράφεται κάθε Object από του professorsArray στο teachers όπου εν τέλει θα χρησιμοποιηθεί στο πρόγραμμά μας. Τέλος, υπάρχουν 2 ακόμα μέθοδοι getTeachers όπου μας επιστρέφει το array list lessons που περιέχει τα μαθήματά μας και μία setTeachers η οποία αλλάζει το linked list των δασκάλων.

4. **TEACHERS CLASS (TeachersClass.java)**

Αυτή η κλάση αναφέρθηκε και προηγουμένως στην ανάλυση της TeachersClassParser.java , και η δουλεία που κάνει είναι να δέχεται Json objects που αφορούν τον καθηγητή και να «σπάει» ουσιαστικά το Json Object του κάθε καθηγητή στα διαφορετικά πεδία απο τα οποία αποτελείται, δηλαδή σε:

- Teacher_code -> κωδικός καθηγητή

- Teacher_name -> όνομα καθηγητή
- Course_code - > κωδικός μαθήματος
- max_hours_per_day->Μέγιστες Ώρες διδασκαλίας του κάθε καθηγητή κάθε μέρα
- max_hours_per_week -> Μέγιστες Ώρες διδασκαλίας του κάθε καθηγητή κάθε εβδομάδα

Εκτός από τους 2 constructors έναν κενό και έναν που κάνει την δουλεία που ειπώθηκε πιο πάνω η κλάση αυτή διαθέτει getters και setters για κάθε πεδίο του object που διαβάζει.

5. **STATE CLASS (State.java)**

Να αναφέρουμε εδώ ότι το πρόγραμμα επιλέξαμε να είναι αρχικά γεμάτο δηλαδή να καλύπτεται αυτόματα όλο το 7ωρο και να μην έχουμε κενά.

Αυτή η κλάση αποτελεί ουσιαστικά τον κινητήριο μοχλό για το πρόγραμμα μας καθώς κάθε αντικείμενο της κλάση θα αποτελεί και ένα διαφορετικό σχολικό πρόγραμμα. Επίσης τσεκάρονται όλοι οι περιορισμοί και μετράμε το σκορ του κάθε παραγόμενου προγράμματος. Πάμε να την δούμε λίγο πιο αναλυτικά:

```
private int RowLength;
private int ColumnLength;
private int thirdDim;
private String [][][] Schedule;
private int fitness;
private ArrayList<LessonsClass> AvailableLessons = new ArrayList<>();
private ArrayList<Integer> AvailableHours = new ArrayList<>();
private Vector<String> ExceptLessons = new Vector<>();
```

Αρχικά βλέπουμε τις δηλώσεις κάποιον μεταβλητών .Κατά σειρά:

- RowLength: είναι το μέγεθος της κάθε σειράς που ουσιαστικά στο πρόγραμμά μας αναπαριστά την κάθε ώρα της μέρας ξεκινώντας από τις 8 μέχρι και τις 3 η ώρα που είναι η τελευταία ώρα μαθήματος. Στο πρόγραμμα οι ώρες του προγράμματος είναι η αναπαράσταση ενός πλήρους 7ώρου και έχουν χωριστεί ως εξής 8:00-9:00, 9:00-10:00, 10:00-11:00,11:00-12:00,12:00-13:00,13:00-14:00,14:00-15:00.
- ColumnLength: είναι το μέγεθος των στηλών που στο πρόγραμμά μας αναπαριστά τις μέρες τις εβδομάδας συνολικά 5 μέρες από Δευτέρα έως Παρασκευή.
- ThirdDim: είναι η Τρίτη διάσταση που χρησιμοποιούμε στον τρισδιάστατο πίνακα και αναπαριστά το κάθε τμήμα του σχολείου.

Έχει μέγεθος 9 καθώς τόσα είναι και τα τμήματα που συνολικά έχει το σχολείο.

- String [][][] Schedule: Είναι η δήλωση του τρισδιάστατου πίνακα που οι 3-διαστάσεις του αποτελούνται από τις παραπάνω μεταβλητές. Είναι τύπου string καθώς αποθηκεύουμε το ζεύγος όνομα μαθήματος-κωδικός καθηγητή. Για παράδειγμα το Schedule[0][0][0] μεταφράζεται ως Δευτέρα ,πρώτη ώρα , τμήμα Α1.
- Fitness: Αποθηκεύεται το σκορ του προγράμματος και προκύπτει από την παραβίαση ή όχι των περιορισμών
- ArrayList<LessonsClass> AvailableLessons: Περιέχει τα μαθήματα που αντιγράφονται σε αυτήν από το ArrayList lessons που περνάμε σαν όρισμα στον constructor που προηγουμένως έχουν διαβαστεί από το json file.
- ArrayList<Integer> AvailableHours : Περιέχει τις διαθέσιμες ώρες του κάθε μαθήματος που περνάμε στην Available lessons. Την χρησιμοποιούμε προκειμένου όταν περνάμε τα μαθήματα στο πρόγραμμα να μειώνουμε τις ώρες έτσι ώστε να μην βάλουμε κάποιο μάθημα παραπάνω από όσες φορές πρέπει.
- ArrayList<Integer> AvailableHours: Είναι ένας Vector Που χρησιμοποιούμε προκειμένου να αποθηκεύσει πόσα μαθήματα διδάσκονται παραπάνω από 2 ώρες έτσι ώστε να εξαιρεθούν από τον 4ο περιορισμό

Στη συνέχεια έχουμε τους Constructors κατά σειρά ο πρώτος δέχεται 3 ορίσματα rowlength, columnlength, thirddim και κατασκευάζει έναν 3-διάστατο πίνακα με αυτές τις διαστάσεις.

Μετά ο επόμενος δέχεται έναν πίνακα τρισδιάστατο ο οποίος δημιουργεί το this.Schedule με κάποιες default διαστάσεις 7,5,9 και στη συνέχεια αντιγράφουμε τον πίνακα που περάσαμε σαν όρισμα στον πίνακα this.Schedule και υπολογίζουμε το fitness του.

Τέλος έχουμε την State initializeState η οποία δέχεται 2 array list ένα με καθηγητές και ένα με τα μαθήματα. Στην συνέχεια γεμίζουμε τον Vector ExceptLessons με τα μαθήματα που διδάσκονται πάνω από 2 ώρες

```
for(int i=0; i<Lessons.size(); i++){
    if(Lessons.get(i).getHours()>2 && !(ExceptLessons.contains(Lessons.get(i).getCourseName()+", "+Teachers.get(i).getTeacher_c
        ExceptLessons.addElement(Lessons.get(i).getCourseName() +", " + Teachers.get(i).getTeacher_code());
    }
}
```

και στο επόμενο for περνάμε τα μαθήματα και τις ώρες αντίστοιχα στο availableLessons και AvailableHours

```
for(int i = 0; i < Lessons.size(); i++){
    AvailableLessons.add(Lessons.get(i));
    AvailableHours.add(Lessons.get(i).getHours());
```

Τώρα με ένα διπλό for έχουμε ορίσει μία τυχαία μεταβλητή random Index η οποία παίρνει τυχαία τιμή από το 0 μέχρι το μέγεθος της λίστας Available lessons και με την μεταβλητή dep αποθηκεύουμε το τμήμα που διδάσκεται το συγκεκριμένο μάθημα που βρήκαμε με δείκτη το random index. Στην συνέχεια με πολλαπλά If τσεκάρουμε το μάθημα σε ποιο τμήμα διδάσκεται και αν η συγκεκριμένη θέση που θα βάλουμε το μάθημα είναι κενή και στην συνέχεια ανακτούμε το μάθημα από το Available lessons και τις ώρες από το available. Τέλος αν οι συνολικές ώρες του μαθήματος είναι μεγαλύτερες του 0 τότε στην θέση του πίνακα [i][j][k] γράφουμε το μάθημα μαζί με τον κωδικό του καθηγητή και μειώνουμε τις ώρες του μαθήματος. Έτσι όταν θα διαβαστεί ένα μάθημα πχ Μαθηματικά Α1 θα περάσει από το If και θα μπει στην θέση [0][0][1] και θα διαβαστεί το επόμενο μάθημα μέχρι όλες οι τάξεις την πρώτη ώρα να έχουν γεμάτο πρόγραμμα . Για να το πετύχουμε αυτό με ένα if τσεκάρουμε αν την ίδια ώρα όλα τα τμήματα έχουν κάποιο μάθημα αν ναι τότε γεμίζουμε με μαθήματα την επόμενη ώρα μαθήματος αλλιώς μειώνουμε το j και ουσιαστικά παραμένουμε στο ίδιο for μέχρι να γεμίσουν όλα.

Τέλος με την κάνουμε calculate fitness του πίνακα που μόλις γεμίσαμε.

- Calculatefitness()

  Με αυτή την μέθοδο «μαζεύουμε» το σκορ για κάθε πρόγραμμα που παράγουμε. Εδώ βλέπουμε πόσους περιορισμούς το πρόγραμμα μας ικανοποιεί. Αρχικά έχουμε την μεταβλητή NumOfConflicts που κρατάει το σκορ και είναι αρχικοποιημένη με 1.

  1ος Περιορισμός:

  Κοιτάμε αν έχει κενά το πρόγραμμά μας όπως είπαμε πριν επιλέξαμε να ξεκινάμε με γεμάτο πρόγραμμα άρα δεν τίθεται θέμα για ύπαρξη κενών οπότε ικανοποιείται πάντα. Αυξάνεται κατά 1 το σκορ.

  2ος Περιορισμός(Σοβαρός):

  Κοιτάμε αν κάποιος καθηγητής διδάσκει παραπάνω από 3 ώρες συνεχόμενα. Επειδή κάθε καθηγητής κάνει διαφορετικό μάθημα κατά συνέπεια είναι το ίδιο με το αν κάποιο μάθημα διδάσκεται 3 ώρες συνεχόμενα. Αυξάνει σκορ κατά 2

  3ος Περιορισμός(Σοβαρός)-Δικός μας:

  Ελέγχουμε αν κάποιος καθηγητής διδάσκει ένα μάθημα σε 2 διαφορετικά τμήματα την ίδια ώρα. Αυξάνει το σκορ κατά 3

  4ος Περιορισμός(Χαλαρός):

Οι ώρες διδασκαλίας είναι κατά το δυνατόν ομοιόμορφα κατανεμημένες σε όλες τις ημέρες της εβδομάδας για κάθε τμήμα. Στον συγκεκριμένο περιορισμό επιτρέπουμε μέχρι 6 δίωρα την εβδομάδα. Αυξάνεται κατά 1 το σκορ.

5ος Περιορισμός(Χαλαρός):

Ομοιόμορφη διδασκαλία των καθηγητών. Ικανοποιείται πάντα καθώς σε κάθε τάξη έχουμε διαφορετικό καθηγητή. Αυξάνεται κατά 1 το σκορ.

Στο τέλος της κλάσης έχουμε τους getters του fitness,του μεγέθους της σειράς , της στήλης και της τρίτης διάστασης. Και ένα fitness setter.

**Αποφασίσαμε να χρησιμοποιήσουμε γενετικό αλγόριθμο και πάμε να τον αναλύσουμε.

6. **Population.java**

Αρχικά αυτή η κλάση μας παράγει το population που θα χρειαστούμε παρακάτω για την υλοποίηση του genetic αλγορίθμου μας. Αρχικά βλέπουμε τον constructor της κλάσης όπου δέχεται το capacity και επιστρέφει ένα population array τύπου state.

Στην συνέχεια έχουμε την Population initializePopulation η οποία μας παράγει τόσα νέα states όσος και ο αριθμός του POPULATION_SIZE όπου κάθε στοιχείο στο array του States είναι ένα διαφορετικό πρόγραμμα και όλα αυτά μαζί αποτελούν έναν πληθυσμό . Με αυτήν την μέθοδο αρχικοποιούμε τον πληθυσμό μας ώστε να ξεκινήσει η διαδικασία του γενετικού αλγορίθμου.

Μετά έχουμε την getState που μας επιστρέφει το array από τα states, και τέλος την sortStatesByFitness η οποία μας ταξινομεί τα states ανάλογα με το πιο έχει καλύτερο fitness κάθε φορά.

7. **Genetic Algorithm Class (GeneticAlgorithmClass.java)**

Ξεκινάμε με τις μεταβλητές Population_size μεγέθους 10 που είναι ο αριθμός που επιλέξαμε να έχουμε στον πληθυσμό από τα προγράμματά μας, την score_to_achieve που είναι το σκορ που θέλουμε να μας επιστρέψει ο γενετικός αλγόριθμος, και η mutation_rate που είναι η πιθανότητα να συμβεί μετάλλαξη στον υπάρχον πληθυσμό

evolvePopulation(Population population): μας επιστρέφει

NewPopulation(Population population): δημιουργεί τον καινούργιο πληθυσμό ανάλογα με το POPULATION_SIZE.

Έπειτα ορίζουμε ένα arraylist FitnessBound το οποίο αναπαριστά την πιθανότητα κάποιο χρωμόσωμα να επιλεγεί και κάθε χρωμόσωμα ανάλογα με το σκορ του δηλαδή το fitness του έχει έναν συγκεκριμένο αριθμό από θέσεις μέσα στο arraylist.

```
for(int i=0; i < POPULATION_SIZE/2; i++) {
    int xIndex = FitnessBounds.get(r.nextInt(FitnessBounds.size()));
    State chromosome1 = population.getState()[xIndex];
    int yIndex = FitnessBounds.get(r.nextInt(FitnessBounds.size()));
    while(yIndex == xIndex)//the indexes of the two selected chromosomes must not be the
    {
        yIndex = FitnessBounds.get(r.nextInt(FitnessBounds.size()));
    }
    State chromosome2 = population.getState()[yIndex];
    int intersectionPoint = r.nextInt( bound: 8) + 1;//we select the intersection point. w
                                                     //which is the departments.
    //based on the intersection point the chromosome1 gets the k to 9 departments of the
    for(int x=0; x < chromosome1.getRowLength(); x++){
        for(int j=0; j < chromosome1.getColumnLength(); j++){
            for(int k=intersectionPoint; k < chromosome1.getThirdDim(); k++){
                temp = chromosome1.getSchedule()[x][j][k];
                chromosome1.getSchedule()[x][j][k] = chromosome2.getSchedule()[x][j][k];
                chromosome2.getSchedule()[x][j][k] = temp;
            }
        }
    }
    //mutating the two new chromosomes and adding them to the new population.
    chromosome1 = mutateChromosome(chromosome1);
    chromosome2 = mutateChromosome(chromosome2);
    NewPopulation.getState()[index] = chromosome1;
    NewPopulation.getState()[++index] = chromosome2;
    index++;
}
```

Στην συνέχεια όπως βλέπουμε και πιο πάνω στην εικόνα δημιουργούμε ένα νέο πληθυσμό και κάθε φορά παράγουμε 2 χρωμοσώματα με τυχαίο τρόπο. Με την xIndex διαλέγουμε τον αριθμό ενός χρωμοσώματος από το list του fitnessBounds και το χρησιμοποιούμε αμέσως μετά για να δημιουργήσουμε το νέο χρωμόσωμα. Αντίστοιχα φτιάχνουμε το yIndex με τον ίδιο τρόπο και έπειτα ελέγχουμε αν είναι ίδια με το xIndex. Αν είναι τότε επιλέγουμε άλλο αριθμό για το yIndex. Αυτό που πετυχαίνουμε με τα Index είναι ότι σε κάθε χρωμόσωμα καλούμε την getState του εκάστωτε population και ο δείκτης αντιστοιχεί σε μία θέση στον πίνακα του population και ουσιαστικά δίνουμε ένα σχολικό πρόγραμμα σε κάθε χρωμόσωμα. Μετά πάλι με τυχαίο τρόπο διαλέγουμε ένα σημείο, το οποίο κυμαίνεται από 0-9 που είναι ουσιαστικά η Τρίτη διάσταση (τα τμήματα δηλαδή) , έτσι ώστε το ένα χρωμόσωμα να πάρει το κομμάτι από IntersectionPoint μέχρι το 9 και το ίδιο και το δεύτερο και στην συνέχεια κάνουμε Mutate τα χρωμοσώματα και τα προσθέτουμε στο καινούργιο population.

Μετά αλλάζουμε θέσεις στα δύο χρωμοσώματα και ουσιαστικό το πρώτο θα πάρει τα περιεχόμενα του δεύτερου και το δεύτερο του πρώτου(αντιμετάθεση).

Η τελευταία μέθοδος είναι η mutateChromosome η οποία θα μεταλλάξει το χρωμόσωμα με πιθανότητα μετάλλαξης όση η μεταβλητή Mutation rate. Ουσιαστικά η δουλεία της Mutate είναι να μας αλλάζει μαθήματα σε ένα τμήμα και να υπολογίζοντας κάθε φορά το καινούργιο πρόγραμμα που προέκυψε. Πάλι διαλέγουμε τυχαία 3 αριθμούς ένα για την ώρα έναν για την μέρα και ένα για το τμήμα και η θέση που προκύπτει στον πίνακα βάση αυτών των στοιχείων την αποθηκεύουμε στην temp. Μετά διαλέγουμε πάλι 2 τυχαίους αριθμούς για την ημέρα και την ώρα κρατώντας ίδιο το τμήμα και αντιμεταθέτουμε τα μαθήματα. Αυτό όλο βρίσκεται μέσα σε μία for που θα εκτελεστεί 5 φορές έτσι ώστε να προκύψουν πιο πολλές αλλαγές στο πρόγραμμά μας.

8. **Main.java**

Στην main αρχικά έχουμε μία μέθοδο getJsonObjectFromPath η οποία παίρνει το path του json file και το διαβάζει(πιο πάνω έχουν οριστεί οι μεταβλητές MainLessonsJSONPath και MainTeachersJSONPath που περιέχουν το Path των 2 json files τα οποία βρίσκονται μέσα στο src)και μετά μέσα στα 2 arraylist mainLessons, mainTeachers αποθηκεύουμε τα μαθήματα και τους καθηγητές που θα μας επιστρέψουν το LessonClassParser και το TeachersClassParser. Στην συνέχεια ξεκινάμε και δημιουργούμε τον πρώτο πληθυσμό τον αρχικοποιούμε και περιμένουμε μέχρι να μας επιστρέψει ένα πρόγραμμα με σκορ >= του SCORE_TO_ACHIEVE. Τώρα αν ο πληθυσμός ξεπεράσει τα 100.000 θα κάνει Break και θα ξεκινήσει να παράγει πληθυσμούς και πάλι από το μηδέν. Σε περίπτωση που δεν βρεθεί κρατάμε το δεύτερο καλύτερο πρόγραμμα . Τέλος καλούμε την printFinalSchedule που μας εκτυπώνει το τελικό μας πρόγραμμα με όνομα FinalSchedule σε txt μορφή.

ΠΑΡΑΔΕΙΓΜΑΤΑ ΕΚΤΕΛΕΣΗΣ ΜΕ SCREENSHOT

```
Generation # 41182| Best score: 5 round: 1
---------------------------------------------------------
Generation # 41183| Best score: 5 round: 1
---------------------------------------------------------
Generation # 41184| Best score: 5 round: 1
---------------------------------------------------------
Generation # 41185| Best score: 5 round: 1
---------------------------------------------------------
Generation # 41186| Best score: 8 round: 1
---------------------------------------------------------
Generation # 41187| Best score: 5 round: 1
---------------------------------------------------------
Generation # 41188| Best score: 5 round: 1
---------------------------------------------------------
Generation # 41189| Best score: 5 round: 1
---------------------------------------------------------
Generation # 41190| Best score: 6 round: 1
---------------------------------------------------------
Generation # 41191| Best score: 8 round: 1
---------------------------------------------------------
Generation # 41192| Best score: 5 round: 1
---------------------------------------------------------
Generation # 41193| Best score: 5 round: 1
---------------------------------------------------------
Generation # 41194| Best score: 5 round: 1
---------------------------------------------------------
Generation # 41195| Best score: 9 round: 1


----------A TXT FILE WILL BE CREATED CONTAINING THE SCHEDULE----------

The final Schedule is in the txt file:



    CREATED FinalSchedule.txt CHECK IT FOR THE SCHEDULE


Process finished with exit code 0
```

Program A1	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
8:00-9:00	Agglika , T010	Mathimatika , T001	Geografia , T008	Mathimatika , T001	Neoellhnikh logotexnia , T004
9:00-10:00	Mousiki , T014	Project , T018	Texnologia , T012	Gallika/Germanika , T011	Gallika/Germanika , T011
10:00-11:00	Geografia , T008	Kallitexnika , T015	Trhskeytika , T009	Pliroforiki , T013	Fysikh Agwgh , T016
11:00-12:00	Arxaia Ellhnika apo metafrash, T003	Istoria , T058	Oikiaki oikonomia , T017	Fysikh , T006	Mathimatika , T001
12:00-13:00	Trhskeytika , T009	Biologia , T007	Neoellhnikh logotexnia , T004	Neoellhnikh Glwssa , T005	Arxaia Ellhnika apo metafrash, T003
13:00-14:00	Istoria , T058	Arxaia Ellhnikh glwssa , T002	Neoellhnikh Glwssa , T005	Mathimatika , T001	Oikiaki oikonomia , T017
14:00-15:00	Arxaia Ellhnikh glwssa , T002	Fysikh Agwgh , T016	Agglika , T010	Arxaia Ellhnikh glwssa , T002	Biologia , T007

Program A2	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
8:00-9:00	Arxaia Ellhnikh glwssa , T002	Oikiaki oikonomia , T017	Mathimatika , T001	Trhskeytika , T009	Gallika/Germanika , T011
9:00-10:00	Oikiaki oikonomia , T017	Gallika/Germanika , T011	Pliroforiki , T013	Texnologia , T012	Neoellhnikh logotexnia , T004
10:00-11:00	Agglika , T010	Geografia , T008	Arxaia Ellhnikh glwssa , T002	Geografia , T008	Arxaia Ellhnika apo metafrash, T003
11:00-12:00	Fysikh , T006	Mathimatika , T001	Kallitexnika , T015	Neoellhnikh logotexnia , T004	Neoellhnikh Glwssa , T005
12:00-13:00	Arxaia Ellhnikh glwssa , T002	Mathimatika , T001	Agglika , T010	Fysikh Agwgh , T016	Istoria , T058
13:00-14:00	Mousiki , T014	Project , T018	Mathimatika , T001	Neoellhnikh Glwssa , T005	Fysikh Agwgh , T016
14:00-15:00	Trhskeytika , T009	Biologia , T007	Arxaia Ellhnika apo metafrash, T003	Biologia , T007	Istoria , T058

Program A3	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
8:00-9:00	Biologia , T007	Pliroforiki , T013	Trhskeytika , T009	Gallika/Germanika , T011	Agglika , T010
9:00-10:00	Arxaia Ellhnikh glwssa , T002	Oikiaki oikonomia , T017	Istoria , T058	Arxaia Ellhnika apo metafrash, T003	Neoellhnikh Glwssa , T005
10:00-11:00	Kallitexnika , T015	Gallika/Germanika , T011	Neoellhnikh logotexnia , T004	Texnologia , T012	Geografia , T008
11:00-12:00	Neoellhnikh Glwssa , T005	Arxaia Ellhnikh glwssa , T002	Biologia , T007	Istoria , T058	Arxaia Ellhnika apo metafrash, T003
12:00-13:00	Project , T018	Fysikh Agwgh , T016	Oikiaki oikonomia , T017	Mousiki , T014	Geografia , T008
13:00-14:00	Neoellhnikh logotexnia , T004	Mathimatika , T001	Agglika , T010	Trhskeytika , T009	Fysikh , T006
14:00-15:00	Fysikh Agwgh , T016	Arxaia Ellhnikh glwssa , T002	Mathimatika , T001	Mathimatika , T001	Mathimatika , T001

Program B1	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
8:00-9:00	Neoellhnikh logotexnia , T022	Mathimatika , T019	Oikiaki oikonomia , T036	Project , T037	Fysikh , T024
9:00-10:00	Trhskeytika , T028	Fysikh Agwgh , T035	Neoellhnikh Glwssa , T023	Arxaia Ellhnikh glwssa , T020	Arxaia Ellhnikh glwssa , T020
10:00-11:00	Fysikh Agwgh , T035	Arxaia Ellhnika apo metafrash, T021	Xhmeia , T038	Gallika/Germanika , T030	Istoria , T027
11:00-12:00	Gallika/Germanika , T030	Mathimatika , T019	Mousiki , T033	Fysikh , T024	Kallitexnika , T034
12:00-13:00	Biologia , T025	Pliroforiki , T014	Geografia , T026	Agglika , T029	Texnologia , T031
13:00-14:00	Istoria , T027	Neoellhnikh logotexnia , T022	Geografia , T026	Mathimatika , T019	Arxaia Ellhnika apo metafrash, T021
14:00-15:00	Arxaia Ellhnikh glwssa , T020	Trhskeytika , T028	Mathimatika , T019	Neoellhnikh Glwssa , T023	Agglika , T029

Program B2	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
8:00-9:00	Istoria , T027	Arxaia Ellhnikh glwssa , T020	Neoellhnikh Glwssa , T023	Fysikh Agwgh , T035	Arxaia Ellhnikh glwssa , T020
9:00-10:00	Geografia , T026	Mathimatika , T019	Project , T037	Neoellhnikh logotexnia , T022	Mousiki , T033
10:00-11:00	Mathimatika , T019	Arxaia Ellhnikh glwssa , T020	Trhskeytika , T028	Fysikh , T024	Arxaia Ellhnika apo metafrash, T021
11:00-12:00	Fysikh Agwgh , T035	Geografia , T026	Arxaia Ellhnika apo metafrash, T021	Oikiaki oikonomia , T036	Biologia , T025
12:00-13:00	Xhmeia , T038	Trhskeytika , T028	Neoellhnikh logotexnia , T022	Pliroforiki , T014	Gallika/Germanika , T030
13:00-14:00	Texnologia , T031	Mathimatika , T019	Fysikh , T024	Neoellhnikh Glwssa , T023	Mathimatika , T019
14:00-15:00	Kallitexnika , T034	Agglika , T029	Agglika , T029	Gallika/Germanika , T030	Istoria , T027

Program B3	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
8:00-9:00	Arxaia Ellhnikh glwssa , T020	Texnologia , T031	Fysikh Agwgh , T035	Agglika , T029	Kallitexnika , T034
9:00-10:00	Istoria , T027	Arxaia Ellhnikh glwssa , T020	Fysikh , T024	Geografia , T026	Pliroforiki , T014
10:00-11:00	Arxaia Ellhnikh glwssa , T020	Mathimatika , T019	Geografia , T026	Project , T037	Biologia , T025
11:00-12:00	Mathimatika , T019	Gallika/Germanika , T030	Trhskeytika , T028	Mousiki , T033	Agglika , T029
12:00-13:00	Gallika/Germanika , T030	Istoria , T027	Neoellhnikh Glwssa , T023	Neoellhnikh Glwssa , T023	Trhskeytika , T028
13:00-14:00	Fysikh , T024	Arxaia Ellhnika apo metafrash, T021	Mathimatika , T019	Oikiaki oikonomia , T036	Neoellhnikh logotexnia , T022
14:00-15:00	Mathimatika , T019	Neoellhnikh logotexnia , T022	Xhmeia , T038	Fysikh Agwgh , T035	Arxaia Ellhnika apo metafrash, T021

Program C1	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
8:00-9:00	Istoria , T047	Istoria , T047	Arxaia Ellhnikh glwssa , T040	Project , T056	Mathimatika , T039
9:00-10:00	Arxaia Ellhnikh glwssa , T040	Fysikh , T044	Mousiki , T014	Neoellhnikh logotexnia , T042	Project , T056
10:00-11:00	Trhskeytika , T048	Biologia , T045	Trhskeytika , T048	Fysikh , T044	Texnologia , T051
11:00-12:00	Fysikh Agwgh , T055	Gallika/Germanika , T050	Neoellhnikh logotexnia , T042	Mathimatika , T039	Koinwnikh & Politiki Agwgi , T046
12:00-13:00	Koinwnikh & Politiki Agwgi , T046	Neoellhnikh Glwssa , T043	Mathimatika , T039	Neoellhnikh Glwssa , T043	Fysikh Agwgh , T055
13:00-14:00	Mathimatika , T039	Arxaia Ellhnika apo metafrash, T041	Arxaia Ellhnika apo metafrash, T041	Gallika/Germanika , T050	Pliroforiki , T052
14:00-15:00	Agglika , T049	Arxaia Ellhnikh glwssa , T040	Xhmeia , T057	Agglika , T049	Kallitexnika , T054

Program C2	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
8:00-9:00	Arxaia Ellhnikh glwssa , T040	Gallika/Germanika , T050	Trhskeytika , T048	Neoellhnikh Glwssa , T043	Gallika/Germanika , T050
9:00-10:00	Fysikh Agwgh , T055	Koinwnikh & Politiki Agwgi , T046	Arxaia Ellhnikh glwssa , T040	Istoria , T047	Arxaia Ellhnika apo metafrash, T041
10:00-11:00	Mathimatika , T039	Texnologia , T051	Mousiki , T014	Trhskeytika , T048	Project , T056
11:00-12:00	Istoria , T047	Arxaia Ellhnika apo metafrash, T041	Fysikh Agwgh , T055	Fysikh , T044	Mathimatika , T039
12:00-13:00	Kallitexnika , T054	Neoellhnikh logotexnia , T042	Neoellhnikh Glwssa , T043	Agglika , T049	Mathimatika , T039
13:00-14:00	Fysikh , T044	Pliroforiki , T052	Agglika , T049	Mathimatika , T039	Koinwnikh & Politiki Agwgi , T046
14:00-15:00	Biologia , T045	Project , T056	Neoellhnikh logotexnia , T042	Xhmeia , T057	Arxaia Ellhnikh glwssa , T040

Program C3	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
8:00-9:00	Fysikh Agwgh , T055	Texnologia , T051	Mathimatika , T039	Arxaia Ellhnika apo metafrash, T041	Xhmeia , T057
9:00-10:00	Koinwnikh & Politiki Agwgi , T046	Neoellhnikh logotexnia , T042	Biologia , T045	Mathimatika , T039	Neoellhnikh Glwssa , T043
10:00-11:00	Fysikh Agwgh , T055	Trhskeytika , T048	Arxaia Ellhnikh glwssa , T040	Kallitexnika , T054	Neoellhnikh Glwssa , T043
11:00-12:00	Arxaia Ellhnikh glwssa , T040	Arxaia Ellhnikh glwssa , T040	Mousiki , T014	Istoria , T047	Pliroforiki , T052
12:00-13:00	Project , T056	Trhskeytika , T048	Koinwnikh & Politiki Agwgi , T046	Mathimatika , T039	Project , T056
13:00-14:00	Arxaia Ellhnika apo metafrash, T041	Gallika/Germanika , T050	Gallika/Germanika , T050	Neoellhnikh logotexnia , T042	Mathimatika , T039
14:00-15:00	Istoria , T047	Fysikh , T044	Agglika , T049	Fysikh , T044	Agglika , T049

Class :

A	B	C
T001 --> ALAMPOURTZIDOU XRISTINA	T019 --> STIKOYDI GEWRGIA	T039 --> FILIPAS GEORGIOS
T002 --> ATHANASIOU GEWRGIA	T020 --> ALAMPOURTZIDOU XRISA	T040 --> THANASIA KATERINA
T003 --> ARGYRAKIS GEWRGIOS	T021 --> ARGYRAKIS KOSTAS	T041 --> KOURNOUTAS XRISTOS
T004 --> VARVARIGOS NIKOLAOS	T022 --> VARVARIGOS BASILHS	T042 --> KAPELLA MARIA
T005 --> GRIVAS IWANNIS	T023 --> GRIVAKOU KATERINA	T043 --> KONTOPIDI STELLA
T006 --> GRIVAKOU MARIA	T024 --> MPAKIRIS SOTIRIS	T044 --> MYLWNA MARIA
T007 --> POLYDWROS KWSTANTINOS	T025 --> DWRHS KWSTANTINOS	T045 --> MARGETIS LEFTERIS
T008 --> PRATA ELENH	T026 --> KATZABELOY MARIA	T046 --> RIGOPOULOS XRISTOS
T009 --> DIALINOS NIKOLAOS	T027 --> DIALAKIS NIKOLAOS	T047 --> MIRARAKI VASILIKI
T010 --> LALAS ANDREAS	T028 --> KOTITSAS ANDREAS	T048 --> PANOURGIA FOFI
T011 --> THEODOROPOULOU AIKATERINH	T029 --> PSILAKI NASIA	T049 --> DERVENITSOU KYRGIAKOULA
T012 --> KWTSELH MARIA	T030 --> KATSELH MARIA	T050 --> WOLFGAG RODE
T013 --> DALIS LAMBROS	T031 --> DALAS LAMBROS	T051 --> DEOUDE KOULA
T014 --> IWANNOU VARVARA	T014 --> IWANNOY VARVARA	T052 --> KATSAROS KWSTAS
T015 --> ATHANASIOU ELENH	T033 --> ATHANASIA ELENH	T014 --> IWANNOY VARVARA
T016 --> LYKARDIOPOULOU ELISAVET	T034 --> LYKARDI ELISAVET	T054 --> PSILOUTSAKOU ELENH
T017 --> GOYSHS KWSTANTINOS	T035 --> KOYSHS KWSTANTINOS	T055 --> SGOUROS GIANNIS
T018 --> PAPAFILIS THEODWROS	T036 --> PAPAFILIS KWSTAS	T056 --> KORRRE STELLA
T058 --> VISKAS MANOLIS	T037 --> KATSAROY SOFIA	T057 --> ILIADI KOSTANTINA
	T038 --> PILAVAKI PANAGIOTA	