ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Σχεδιασμός-Δημιουργία GPS εφαρμογής με την χρήση της γλώσσας Java.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΤΟΥ

Γεώργιου Άγγελου Κουσίδη

(ΑΕΜ: 2501)

Επιβλέπων : ΜΙΧΑΗΛ ΔΟΣΗΣ

Καστοριά Μάϊος - 2022

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Σχεδιασμός-Δημιουργία GPS εφαρμογής με την χρήση της γλώσσας Java.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΤΟΥ

Κουσίδη Γεώργιου Άγγελου

(ΑΕΜ: 2501)

Επιβλέπων : **ΜΙΧΑΗΛ ΔΟΣΗΣ**

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την ημερομηνία εξέτασης

.....	.....	.....
Ον/μο Μέλους	Ον/μο Μέλους	Ον/μο Μέλους
Ιδίοτητα Μέλους	Ιδίοτητα Μέλους	Ιδίοτητα Μέλους

Καστοριά Μάϊος - 2022

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέπων καθηγητή μου, κ. Μπάτο Παναγιώτη για την ευκαιρία που μου έδωσε με την ανάθεση αυτής της πτυχιακής εργασίας αλλά, για την βοήθεια και υποστήριξη του σε όλη την διάρκεια εκπόνησης της παρούσας εργασίας και τέλος για υπομονή που μου έδειξε.

Περίληψη

Η εργασία μου αφορά τον σχεδιασμό-δημιουργία μιας GPS εφαρμογής με την χρήση της γλώσσας Java. Αρχικά, για να κατανοήσω καλύτερα το θέμα με το οποίο ασχολήθηκα στην παρούσα εργασία, έπρεπε να κατανοήσω με απλές διαδικασίες πως λειτουργούν τα GPS και πώς χρησιμοποιούνται οι διεπαφές τους. Στην συνέχεια, έψαξα να βρω πληροφορίες για διάφορα GPS, τα οποία ανάλυσα με βάση την διεπαφή τους ως προς τον χρήστη αλλά και την χρησιμότητα τους προς τον άνθρωπο. Παράλληλα, οι GPS συσκευές προσφέρουν πάρα πολλές λειτουργίες στον άνθρωπο και τον βοηθούν στην καθημερινότητα του με έξυπνες λύσεις και συμβουλές. Έτσι, ανάλυσα κάποιες από αυτές τις συσκευές με έμφαση στις λειτουργίες που προσφέρουν. Επιπλέον, είδα και ανάλυσα τα είδη, τις κατηγορίες και τους τύπους των GPS εφαρμογών. Με βάση σε αυτές τις πληροφορίες , κατέληξα στο συμπέρασμα ότι τα μεταφορικά λογισμικά αναπτύσσονται και εξελίσσονται με ραγδαίους ρυθμούς και ότι μέρα με την μέρα, διαπιστώνονται καινούργιες πτυχές στην οδήγηση. Επιπρόσθετα, παρατήρησα ότι η επιστήμη της πληροφορικής με διάφορα μέσα έδωσε και θα δώσει χείρα βοηθείας στον μεταφορικό τομέα με την δημιουργία διάφορων καινοτομιών και λειτουργιών. Αξίζει να σημειωθεί ότι, με την εισαγωγή του τομέα της πληροφορικής στα μέσα μαζικής μετακίνησης, αναπτύσσεται και η τεχνολογία της αυτόματης οδήγησης. Τεχνολογικές καινοτομίες που εκτός από τα πλεονεκτήματα που προσφέρουν σε καθημερινό τομέα, βοηθούν και στις μετακινήσεις αγαθών ώστε να αποδίδουν καλύτερα στην δουλειά τους οι πωλητές και να επικοινωνούν με σαφήνεια ώστε να μπορούν να αντιμετωπίσουν τις όποιες δυσκολίες προκύψουν αντίστοιχα. Με βάση τις παραπάνω πληροφορίες, η αυτόματη μετακίνηση θα γίνει αναπόσπαστο κομμάτι σε διάφορους επαγγελματικούς τομείς με ποικίλες εφαρμογές. Συμπληρωματικά, για την υλοποίηση της εφαρμογής

χρησιμοποίησα το πρόγραμμα Netbeans για να δημιουργήσω τον κώδικα και το εκτελέσιμο αρχείο της εφαρμογής.

Στην συνέχεια, με την βοήθεια του προγράμματος Word αναπαράστησα την εφαρμογή μου με την χρήση διαγραμμάτων UML και άντλησα πραγματικά δεδομένα εντός του Διαδικτύου για τα κριτήρια της βέλτιστης διαδρομής, τα οποία είναι ο πυλώνας της εφαρμογής μου.

Λέξεις Κλειδιά: *GPS, Διαγράμματα UML, Υλοποίηση Εφαρμογής, Προγράμματα*

Abstract

My work concerns the design-creation of a GPS application using the Java language. Initially, in order to better understand the topic I addressed in this paper, I had to briefly understand with simple procedures how GPS functions and how their interactions are used. Next, we sought to find information on various GPS devices, which I analyzed based on their user interface and their usefulness to humans. At the same time, GPS devices provide a lot of functions to people and help them in their daily lives with smart solutions and tips. So, I analyzed some of these devices with an emphasis on the features they offer. In addition, I saw and analyzed the variety, categories and types of GPS applications. Based on this information, I have come to the conclusion that transfer software are developing and evolving rapidly and that day by day, new aspects of automatic driving are emerging. In addition, I have observed that computer science through various means has given and will give, a helping hand to the transportation field by creating various innovations and functions. It is worth noting that, with the introduction of the IT sector in the massing transferring vehicles, the technology of automatic driving is developing. Technological innovations that despite the advantages they provide in an everyday field, they also help in transferring goods so sellers are performing better in their field, and to communicate with clearance so they can face any problem respectively. Based on the above information, automatic transportation will become an integral part of various professional sectors with a variety of applications. In addition, to implement the application I used the program Netbeans to create the code and the executable file of the application. Then, with the help of the Word program, I represented my application using UML diagrams and extracted real data on the Internet for the selection criteria, which are the pillar of our application.

Key Words: *GPS, UML Diagrams, Application Implementation, Programs*

Πίνακας Περιεχομένων

Λίστα Εικόνων

Λίστα Πινάκων

1^{o} Κεφάλαιο: Τα πιο γνωστά GPS

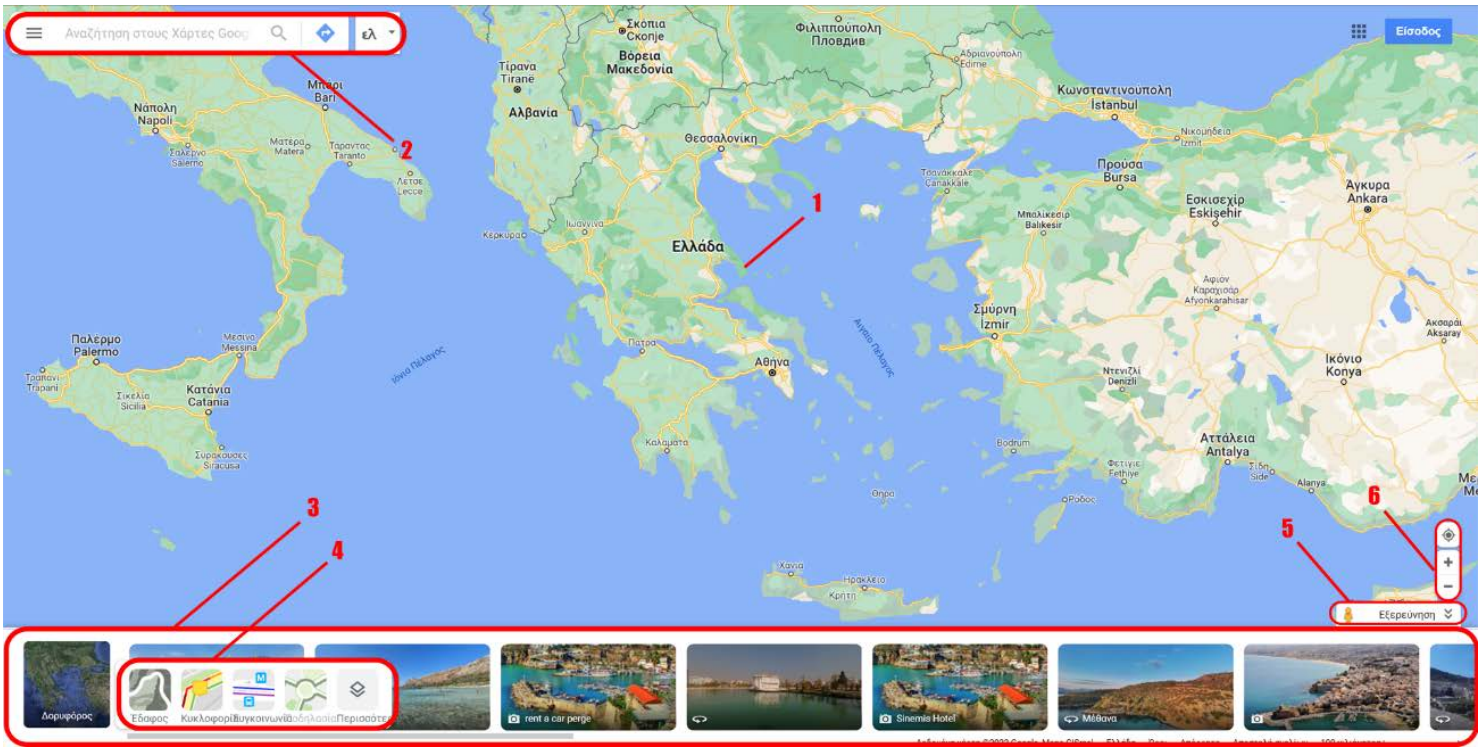
1.1 Λογισμικό gps που υπάρχει στην αγορά και στο διαδίκτυο

Σε όλα τα σύγχρονά gps γενικός υπάρχουν οι εξής λειτουργίες, που από τις πιο βασικές από όλες είναι να μπορούν να βρουν την βέλτιστη διαδρομή ανάμεσα σε δύο σημεία πάνω στον χάρτη, και οι υπόλοιπες ξεκινούν από το να βρίσκει ο χρήστης σημεία ενδιαφέροντος από τις εκάστοτε τοπικές υπηρεσίες, με τις υπόλοιπες να είναι η παρακολούθηση 360^{o} εικόνων σε δρόμους που τους έχει επιτραπεί στην εκάστοτε εταιρία της εφαρμογής να τις τραβήξουν και να τις αποθηκεύσουν. Να έχουν συστήματα που μετράνε την συμφόρηση των δρόμων από αυτοκίνητα και να μπορούν να προτείνουν έναν διαφορετικό δρόμο με λιγότερη κίνηση, και να έχουν σημεία που προειδοποιούν για πυρκαγιές ή για περιοχές με μεγάλη αύξηση κρουσμάτων κάποιας επικίνδυνης ασθένειας π.χ. κορονοϊός. Η γνωστή αναζήτηση για να μεταφερθείς στο συγκεκριμένο μέρος πάνω στον χάρτη, και η μετακίνηση του χάρτη με το ποντίκι με το οποίο χάρης στη ροδέλα γίνεται το ζουμ και με πατημένο το αριστερό κλικ η μετακίνηση σε μέρη που δεν βλέπουμε στην οθόνη. Τέλος, ως προς τις λειτουργίες τους κάθε gps υπάρχει μια κατάσταση που όταν είναι ενεργοποιημένη συνήθως με το πάτημα ενός κουμπιού ο χρήστης βλέπει την παγκόσμια τοποθεσία του πάνω στον χάρτη.

Ο τρόπος με τον οποίο τα gps βγάζουν χρήματα είναι με το να προτείνουν μαγαζιά στους χρήστες τους ως μια μορφή διαφήμισης και συλλέγοντας τις τοποθεσίες και τα μαγαζιά τα οποία επισκέπτονται για πιο στοχευμένη διαφήμιση.

Google Maps

Είναι το δημοφιλέστερο gps του κόσμου φτιαγμένο το 2005 και έχει πάνω από 1 δισεκατομμύρια κόσμο που το χρησιμοποιεί κάθε μήνα. Ακολουθεί η εξήγηση της κάθε λειτουργίας των google maps.

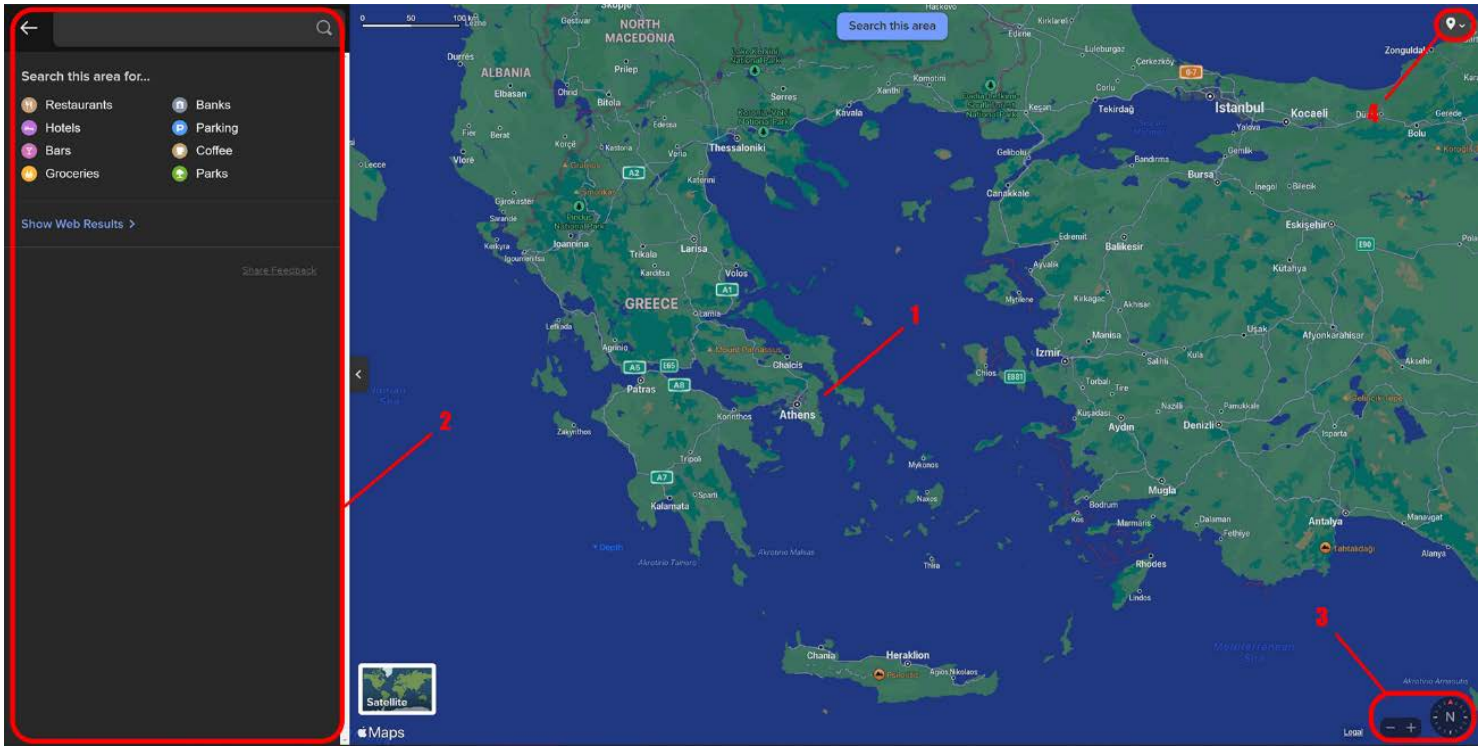


Εικόνα 1. 1

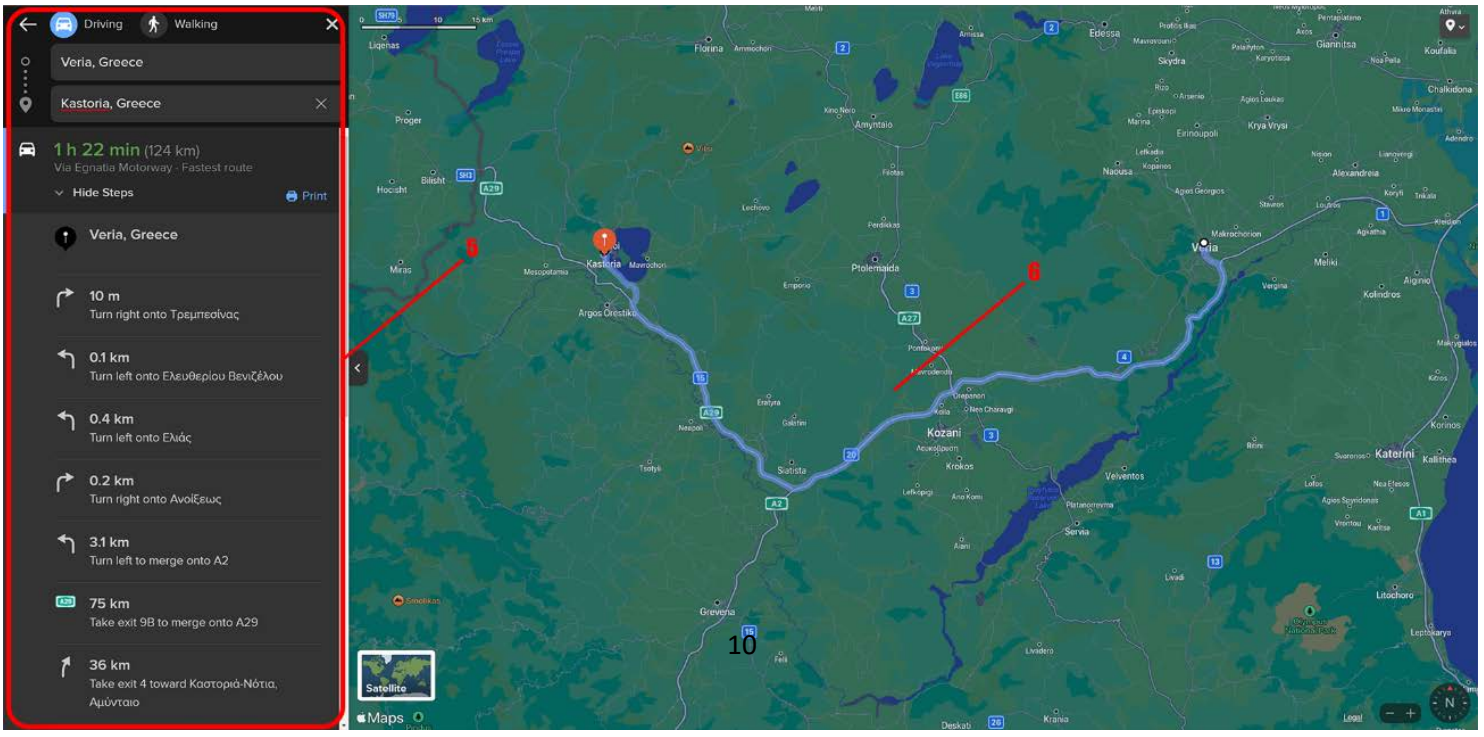
1. Ο χάρτης του gps και μπορείς να μετακινηθείς σε αυτόν έχοντας πατημένο το αριστερό κλικ του ποντικιού ενώ μετακινείται το ποντίκι
2. Η γραμμή αναζήτησης στην οποία γίνεται πολύ εύκολο ο εντοπισμός της τοποθεσίας που ψάχνει ο χρήστης
3. Οι δημοφιλείς τοπικές εικόνες της περιοχής που δείχνει ο χάρτης
4. Ο τρόπος με τον οποίο μπορεί ο χρήστης να αλλάξει τις πληροφορίες του χάρτη π.χ. (έδαφος, κυκλοφορία, σημεία ενδιαφέροντος τουριστών)
5. Πατώντας το κίτρινο ανθρωπάκι δίνεται η ικανότητα των 360^{o} εικόνων παρακολούθησης
6. Τα τρία κουμπάκια τα οποία συν πλην είναι για να δούμε τον χάρτη με πιο συγκεκριμένες ή γενικές πληροφορίες αναλόγως την οπτική μας εικόνα. Και το τρίτο που είναι για να βρούμε την ακριβή τοποθεσία μας πάνω στον χάρτη

Apple Maps

Είναι μεγαλύτερος ανταγωνιστής των google maps γιατί έρχεται προεγκατεστημένο στις συσκευές της apple. Το apple maps είναι διαθέσιμο μόνο στα προϊόντα της apple ,και έχει την εξής διεπαφή για τον χειρισμό του.



Εικόνα 1. 2



Εικόνα 1. 3

1. Ο χάρτης του gps
2. Η γραμμή αναζήτησης
3. Το ζουμ του χάρτη και μια πυξίδα για προσανατολισμό
4. Η επιλογή εντοπισμού της παγκόσμιας θέσης του χρήστη
5. Έπειτα από την αναζήτηση δείχνει σε πόσο χρόνο και τι ταχύτητα θα φτάσει ο χρήστης στον προορισμό του μαζί με οδηγίες για το που θα στρίψει
6. Η διαδρομή πάνω στον χάρτη

1.2 Εργαλεία Που Χρησιμοποιήθηκαν Για Την Ανάπτυξη Του GPS

Για την ανάπτυξη του GPS χρησιμοποιήθηκε το προγραμματιστικό περιβάλλον Netbeans

1.3 Ερευνητικά Δεδομένα

https://www.google.com/maps/ Εδώ βρίσκεται η ιστοσελίδα των google maps που από την οποία πάρθηκαν πολλά δεδομένα για την ολοκλήρωση της πτυχιακής

https://en.wikipedia.org/wiki/Google_Maps Η βικιπαίδεια των google maps η οποία χρησίμευσε στο να γραφτεί το εγχειρίδιο των gps

https://en.wikipedia.org/wiki/GPS_tracking_unit Ιστοσελίδα στην οποία εξηγείται ο τρόπος που τα gps βρίσκουν την τοποθεσία του χρήστη στο χάρτη

https://en.wikipedia.org/wiki/Global_Positioning_System Παρόμοια ιστοσελίδα της βικιπαίδεια με τον παραπάνω σύνδεσμο

https://www.sixt.com/magazine/tips/top-free-navigation-apps/#Gaia Ιστοσελίδα με διάφορα δημοφιλή gps στην αγορά

2ο Κεφάλαιο: Ανάλυση UML διαγραμμάτων

Σε αυτό το κεφάλαιο θα αναλύσουμε τα μέρη από τα οποία αποτελείται η εφαρμογή καθώς και τον τρόπο/μέσα δημιουργίας της.

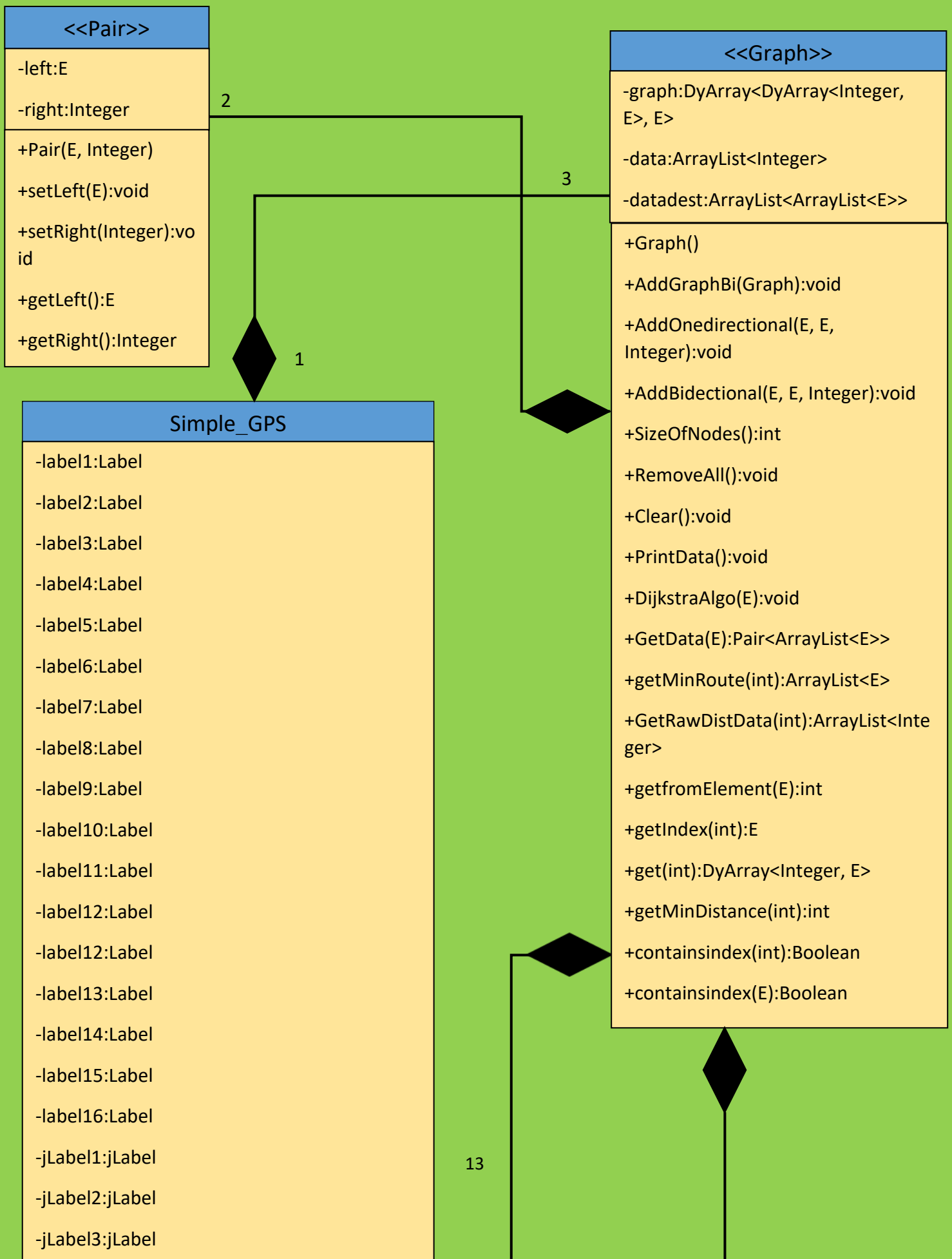
2.1 UML Διαγράμματα

Για την δημιουργία της εφαρμογής χρησιμοποιήθηκε η γλώσσα προγραμματισμού Java. Επομένως, με την βοήθεια σχεδιαγραμμάτων UML θα σας παρουσιάσω την λειτουργία της εφαρμογής μου. Τα σχεδιαγράμματα δημιουργήθηκαν με το πρόγραμμα Word, το οποίο είναι ένα περιβάλλον εύχρηστο προς τον χρήστη και πολύ αποτελεσματικό και αποδοτικό ως προς την δημιουργία εικονικών σχεδιαγραμμάτων με βάση σενάρια ή εφαρμογές [4].

2.2 Πως χρησιμοποιούνται τα διαγράμματα

Το **διάγραμμα κλάσης** έχει τις εξής πέντε κλάσεις οι οποίες είναι η Simple_GPS, Graph, DyArray, Pair, και Comparator_Pair. Από τις οποίες οι Graph, DyArray, και Pair είναι διεπαφές, η DyArray και η Pair είναι διεπαφές στην Graph, και η Graph είναι η μόνη διεπαφή της Simple_GPS. Η Comparator_Pair, Pair, και DyArray έχουν συσσωμάτωση στην Graph πού σημαίνει ότι, δεν μπορούν να υπάρχουν χωρίς την ύπαρξη της Graph, και με την σειρά της η Graph έχει συσσωμάτωση στην Simple_GPS. Η Comparator_Pair βοηθάει στη σύγκριση της κλάσης Pair στη Graph και υπάρχει μόνο μία οντότητα στην Graph. Ενώ η DyArray και η Pair έχουν δύο οντότητες στην Graph. Τέλος η Graph έχει τρείς οντότητες στην Simple_GPS ενώ η Simple_GPS έχει μόνο μια [8].

Class Diagram



-jLabel5:jLabel

-textField1:TextField

-textField2:TextField

-textField3:TextField

-temp:ArrayList<Checkbox>

-states2:ArrayList<Checkbox>

-citydata:ArrayList<Checkbox>

-citydata2:ArrayList<Checkbox>

-Arrlabel:ArrayList<ArrayList<Label>>

-indexes:ArrayList<Integer>

-collide:ArrayList<Rectangle>

-colindex:int

-addflag:Boolean

-moveflag:Boolean

-movetemp:Boolean

-obj:Graph

-obj2:Graph

-objt:Graph

-jDialog1:jDialog

-jPanel1:jPanel

-jFrame2:jFrame

-button1:Button

-button2:Button

-button3:Button

-button4:Button

-button5:Button

-button6:Button

-button7:Button

-button8:Button

-button9:Button

-button10:Button

-button11:Button

-button12:Button

-button13:Button

-choice1:Choice

1

Helps To Compare Pair

Comparator_Pair

+compare(Pair<T>, Pair<T>):int

2

<<DyArray>>

-array:ArrayList<E>

-indexcount:ArrayList<T>

-indexcounter:int

+DyArray()

+DyArray(T, E)

+addElementAt(T, E):void

+setElementAt(T, E):void

+getfromindex(T):E

+getfromindexcounter(T):int

+getindex(int):T

+get(int):E

+size():int

+remove(int):void

+clear():void

+containsElement(E):Boolean

+containsIndex(T):Boolean

+containsIndex(int):Boolean

-choice2:Choice

-choice3:Choice

-choice4:Choice

+Simple_GPS()

+main(String):void

+getCheckBoxes():ArrayList<Checkbox>

+TikCheckBoxes(Choice):void

+distancecond():Integer

+applychecksfromtextfields():void

+iscollided(Point):Boolean

+getcollided(Point):Boolean

+movecheck(Checkbox, Point):void

+gethalfdist(Point, Point):Point

+showroad(ActionEvent):void

-choice1ItemStateChanged(ItemEvent):void

-choice2ItemStateChanged(ItemEvent):void

-choice3ItemStateChanged(ItemEvent):void

-choice4ItemStateChanged(ItemEvent):void

-button2ActionPerformed(ActionEvent):void

-button3ActionPerformed(ActionEvent):void

-button5ActionPerformed(ActionEvent):void

-button6ActionPerformed(ActionEvent):void

-button7ActionPerformed(ActionEvent):void

-button8ActionPerformed(ActionEvent):void

-button9ActionPerformed(ActionEvent):void

-button10ActionPerformed(ActionEvent):void

-button11ActionPerformed(ActionEvent):void

-button12ActionPerformed(ActionEvent):void

-button13ActionPerformed(ActionEvent):void

-textField1MouseClicked(MouseEvent):void

-textField2MouseClicked(MouseEvent):void

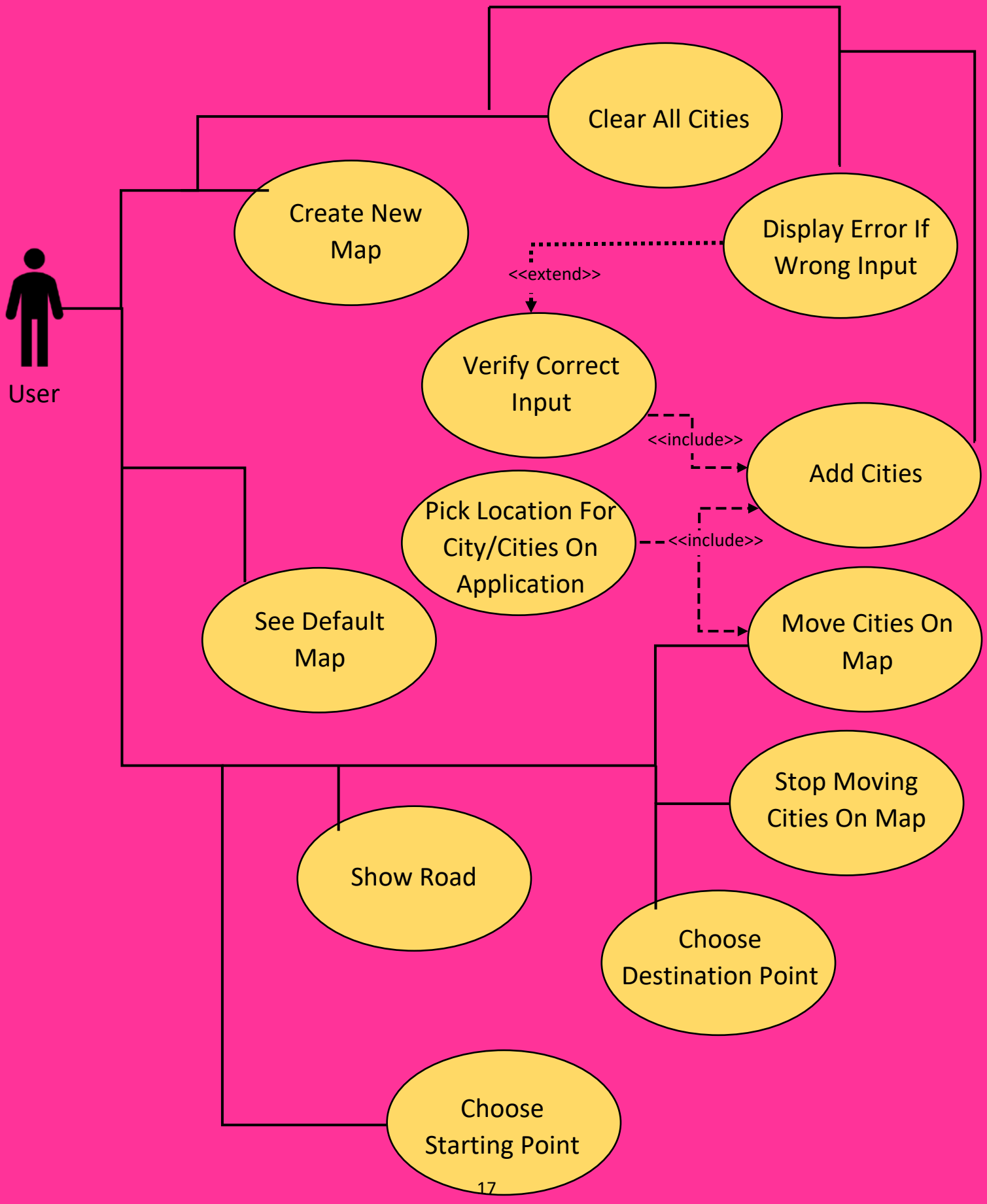
-textField3MouseClicked(MouseEvent):void

-jFrame2MouseClicked(MouseEvent):void

-jFrame2MouseMoved(MouseEvent):void

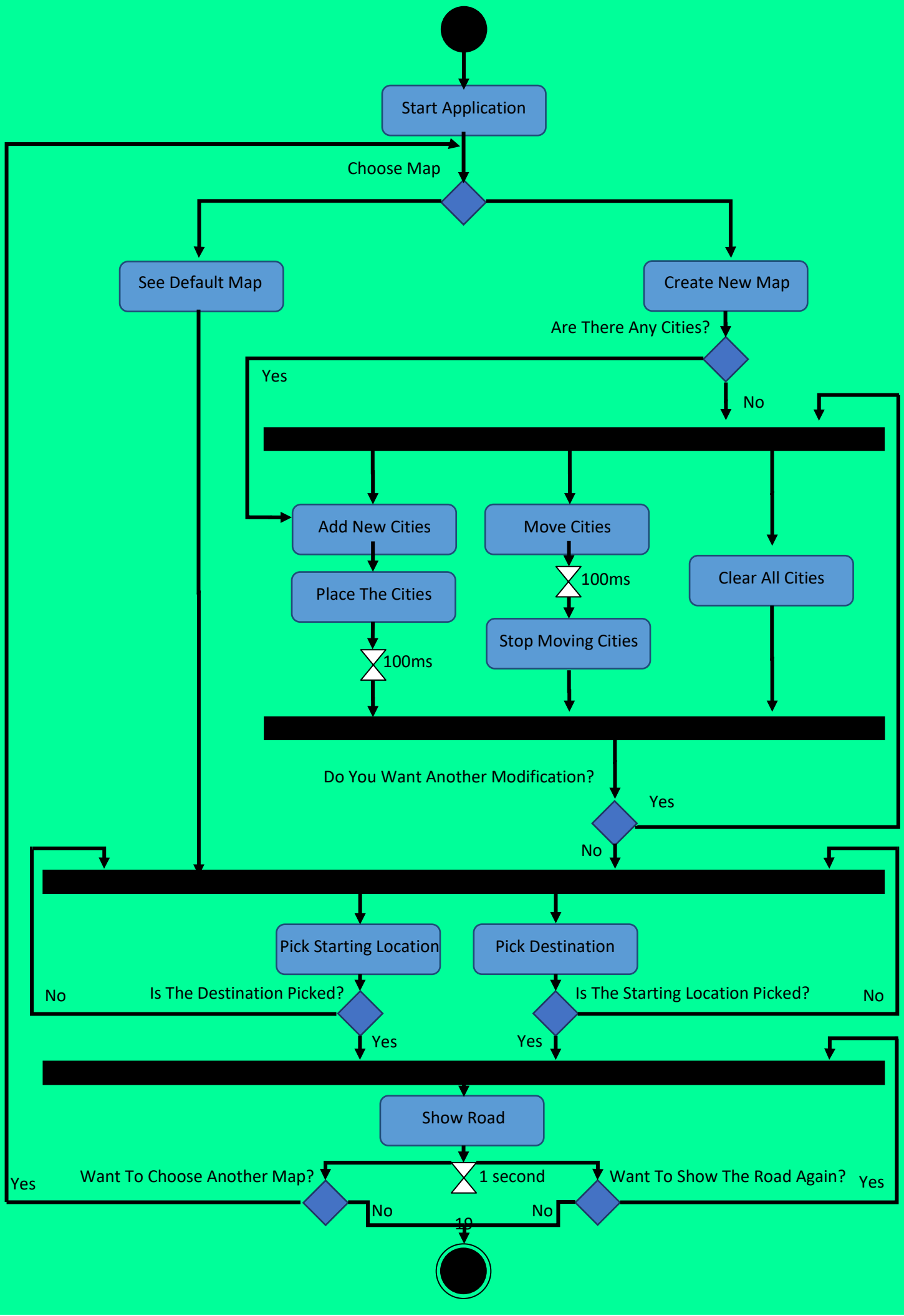
Το **διάγραμμα περίπτωσης χρήσης** έχει ως κύριο ηθοποιό τον χρήστη για την λειτουργία του. Ο χρήστης έχει ως δυνατότητες να επιλέξει τον προκαθορισμένο χάρτη είτε να φτιάξει έναν χάρτη της επιλογής του. Και στους δύο χάρτες έχει την επιλογή να δει την βέλτιστη διαδρομή ο ηθοποιός, αφότου πρώτα διαλέξει δύο πόλεις την πόλη αφετηρίας και την πόλη προορισμού. Στην επιλογή του αυθαίρετου χάρτη ο χρήστης έχει επιπλέον τις επιλογές να εισάγει τις πόλεις με τις διαδρομές τους, και να μετακινήσει τις πόλεις πάνω στο παράθυρο της εφαρμογής όπως και να σταματήσει την κατάσταση μετακίνησης πόλεων. Σε περίπτωση που ο χρήστης έχει κάνει κάποιο λάθος τότε έχει την επιλογή να σβήσει όλες τις πόλεις με το πάτημα ενός κουμπιού, και στην περίπτωση που ο χρήστης δώσει λανθασμένες πληροφορίες για την απόσταση των πόλεων ή έχει δώσει την ίδια πόλη δύο φορές, τότε παράγονται μηνύματα για να καθοδηγήσουν τον χρήστη ώστε να κάνει τις σωστές επιλογές για να περιληφθούν[8].

Use Case Diagram



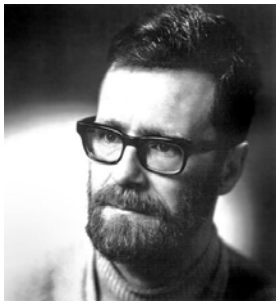
Το **διάγραμμα δραστηριοτήτων** ξεκινάει με την επιλογή του χάρτη που επιθυμεί ο χρήστης. Στην περίπτωση του νέου χάρτη ο χρήστης έχει τις επιπλέον επιλογές που είναι να προσθέσει πόλεις, να μετακινήσει πόλεις αν υπάρχουν, και να σβήσει τις πόλεις αν υπάρχουν. Οι επιλογές να προσθέσει πόλεις και να τις μετακινήσει έχουν χρονική καθυστέρηση εκατό χιλιοστών. Έπειτα, ο χρήστης επιλέγει την πόλη αφετηρίας και την πόλη προορισμού, οι οποίες παίρνουν τα χρώματα πράσινο και κίτρινο αντιστοίχως, και αφότου επιλεχθούν και η δύο μπορεί ο χρήστης να δει την βέλτιστη διαδρομή πάνω σε όποιο από τους δύο χάρτες έχει επιλέξει. Για να δει την διαδρομή ο χρήστης περιμένει για ένα δευτερόλεπτο την κάθε πόλη οι οποίες βάφονται με το χρώμα πορτοκαλί μέχρι να φτάσει στην τελευταία πόλη προορισμού [4][8].

Activity Diagram



3ο Κεφάλαιο: Αλγόριθμος του Dijkstra

3.1 Λίγα λογία για τον Dijkstra:



Εικόνα 3. 1

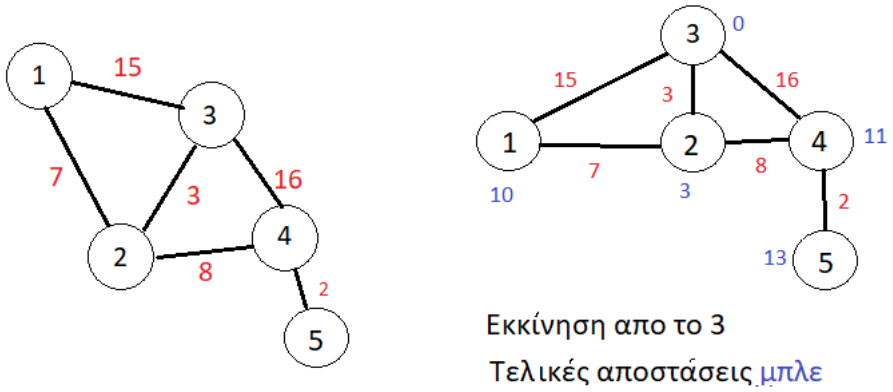
Ο Edsger Wybe Dijkstra ήταν ένας επιστήμονας συστημάτων, προγραμματιστής, μηχανικός λογισμικού, στην ειδικότητα των υπολογιστών. Γεννήθηκε στο Ρότερνταμ της Ολλανδίας και αποφοίτησε από το σχολείο 1948, με μεγάλη προτροπή των γονέων του να επικεντρωθεί στα μαθηματικά και στην φυσική στις αρχές του 1950. Δούλεψε ως προγραμματιστής στο Άμστερνταμ από το 1952 έως το 1962. Ό Dijkstra ξεκίνησε να επεξεργάζεται τον αλγόριθμο το 1956 και το 1959 δημοσίευσε ένα τρισέλιδο άρθρο Ένα μονοπάτι για διασύνδεση με γράφους'. Ήταν ένας, αλγόριθμος για να βρίσκει το συντομότερο μονοπάτι σε έναν γράφο μεταξύ δυο σημείων, που έχει μέχρι και την σήμερον ημέρα την ονομασία ο αλγόριθμος του Dijkstra.

Δούλεψε, σαν προφέσορας για το μεγαλύτερο μέρος της ζωής του, έχοντας μια θέση στον τομέα υπολογιστικές επιστήμες στο πανεπιστήμιο του Τέξας στο Άουστιν από το 1984 έως την συνταξιοδότηση του το 1999.

3.2 Γενικές γνώσεις για τον αλγόριθμο:

Ο αλγόριθμος του Dijkstra είναι ένας αλγόριθμος για να βρίσκει την συντομότερη διαδρομή μεταξύ δυο κόμβων σε ένα γράφημα, ο οποίος μπορεί να προσομοιάζει για παράδειγμα, δίκτυα δρόμων.

Υπάρχουν, πολλές εκδοχές του αλγορίθμου, και, ο πιο κοινός είναι όταν επιλέγουμε μια κορυφή σαν το σημείο εκκίνησης. Από εκεί βρίσκουμε την ελάχιστη απόσταση από όλες τις κορυφές, φτιάχνοντας έτσι, το **δέντρο ελάχιστης διαδρομής**.



Εικόνα 3. 2

Μπορεί, επίσης, να χρησιμοποιηθεί για να βρεθεί μόνο μια απόσταση με δυο σημεία σταματώντας τον αλγόριθμο μόλις, έχει καθοριστεί η μικρότερη απόσταση. Για παράδειγμα, αν οι κομβόι ενός γραφήματος αναπαριστούν πόλεις και οι πλευρές στάθμισης αναπαριστούν αποστάσεις δρόμων μεταξύ δυο ενωμένων πόλεων ,τότε, έχουμε την δυνατότητα να βρούμε τις αποστάσεις των πόλεων μεταξύ τους.

Ο αρχικός αλγόριθμος δεν χρησιμοποιούσε ουρά ελάχιστης προτεραιότητας και έτρεχε σε βαθμό χρόνου πολυπλοκότητας **O(|V|^2)**, οπού **V** είναι ο αριθμός των κόμβων.

Η υλοποίηση που είναι βασισμένη στην ουρά ελάχιστης προτεραιότητας από την σωρό του Fibonacci και τρέχει σε χρόνο **O(|E|+|V|/log|V|)** οπού **E** είναι ο αριθμός των πλευρών.

Αυτός, ο αλγόριθμος θεωρείτε άπληστος γιατί, αποφεύγει να πάει σε μεγάλες σταθμισμένες πλευρές παραπάνω από όσες φορές χρειάζεται.

Τελικά, ο αλγόριθμος **BFS** (best first search) θα μπορούσε να θεωρηθεί σαν μια πρωτόγονη εκδοχή του αλγορίθμου του Dijkstra, οπού η ουρά προτεραιότητας χρησιμοποιείτε σε κατάσταση **FIFO** (first in fist out) [3].

3.3 Λειτουργία του αλγορίθμου Dijkstra:

Ξεκινάμε, με έναν αρχικό κόμβο ο οποίος θα είναι το σημείο που μετράμε την απόσταση από τους υπολοίπους. Δηλαδή, η απόσταση από έναν κόμβο **X** είναι η απόσταση του κόμβου **X** από τον αρχικό. Στον αλγόριθμο του Dijkstra θα δηλώσουμε μερικές αρχικές αποστάσεις από τον αρχικό κόμβο προς τους υπόλοιπους και, θα την βελτιώνουμε αυτήν την απόσταση βήμα προς βήμα [1][2].

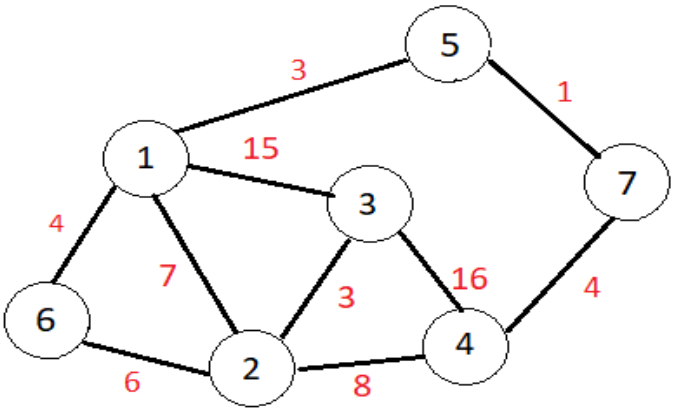
1) Θέτουμε όλους τους κόμβους ως ανεξερεύνητους. Δημιουργούμε, ένα σετ με όλους τους ανεξερεύνητους κόμβους ονομάζοντας το ως το ανεξερεύνητο σετ.
2) Εκχωρούμε, σε κάθε κόμβο μια βοηθητική απόσταση τιμής: επιλέγουμε το μηδέν για τον αρχικό κόμβο και το άπειρο για όλους τους υπόλοιπους. Τότε επιλέγουμε, τον αρχικό κόμβο ώστε να είναι έτοιμος για αναζήτηση.
3) Για κάθε, κόμβο που είναι στο κεφάλι αναζήτησης, αναθεωρούμε όλους τους ανεξερεύνητους γείτονες του και υπολογίζουμε την απόσταση μέσω του κόμβου αναζήτησης. Συγκρίνουμε, την καινούρια απόσταση με την βοηθητική και εκχωρούμε την μικρότερη. Για παράδειγμα, αν ο τρέχοντας κόμβος **Y** έχει την απόσταση 4 από τον αρχικό κόμβο και μπορεί να πάει στον κόμβο **Z** με την πλευρά που τους ενώνει να είναι 3, τότε, η καινούρια απόσταση του **Z** είναι 4 + 3 = 7. Αν, ο **Z** έχει προηγούμενος εξερευνηθεί με μια βοηθητική απόσταση μικρότερη την καινούριας, τότε, ο Z θα κρατήσει την παλιά του απόσταση, ειδάλλως, θα πάρει την καινούρια απόσταση 7.
4) Όταν, έχουμε τελειώσει να ελέγχουμε όλους του γείτονες του κόμβου με τον οποίο εξερευνούμε, θέτουμε τον τωρινό κόμβο ως εξερευνημένο και τον αφαιρούμε από το σετ των ανεξερεύνητων κόμβων. Ένας εξερευνημένος κόμβος δεν θα κοιταχτεί ξανά από τον αλγόριθμο.
5) Στην περίπτωση που επισκεφτούμε τον κόμβο που θέλαμε να βρούμε με την ελάχιστη απόσταση(όταν έχουμε σκοπό μόνο για έναν κόμβο να βρούμε την ελάχιστη απόσταση αλλιώς είναι για όλο το σετ των ανεξερεύνητων κόμβων), τότε αν η μικρότερη βοηθητική απόσταση

που βρέθηκε σε όλους τους κόμβους είναι το άπειρο(άδειασε το σετ των ανεξερεύνητων κόμβων. Σημαίνει, ότι δεν υπάρχει πλευρά που να ενώνει τον κόμβο με τους υπόλοιπους),τότε σταματά. Ο αλγόριθμος έχει πλέον τελειώσει.

6) Ειδάλλως, επιλέγουμε τον ανεξερεύνητο κόμβο που έχει την μικρότερη βοηθητική απόσταση, τον κάνουμε να είναι το καινούριο κεφάλι αναζήτησης, και πηγαίνουμε πίσω στο βήμα 3).

   Όταν σχεδιάσει το μονοπάτι, δεν χρειάζεται να επισκεφτούμε τους κόμβους που έχουν μεγάλες αποστάσεις. Με αυτό τον τρόπο γλιτώνει πολύ χρόνο ο αλγόριθμος και σταματάει πιο νωρίς.

3.4 Παράδειγμα:



Εικόνα 3. 3

Έχουμε το παραπάνω γράφημα. Και, επιλέγουμε να ξεκινήσουμε από τον κόμβο 3. Οπότε θα προκύψει ο παρακάτω πίνακας που μας δείχνει τα βήματα που κάνουμε κρατώντας την απόσταση από τον 3 προς τους υπόλοιπους κόμβους.

Πίνακας επανάληψης Dijkstra:

Πίνακας 1

Vertex	1	2	3	4	5	6	7
3	15_3	3_3	0_3	16_3	∞	∞	∞
2	10_2	·	·	11_2	∞	9_2	∞
1	·	·	·	·	13_1	·	∞
4	·	·	·	·	·	·	15_4
5	·	·	·	·	·	·	14_5
6	·	·	·	·	·	·	·
7	·	·	·	·	·	·	·

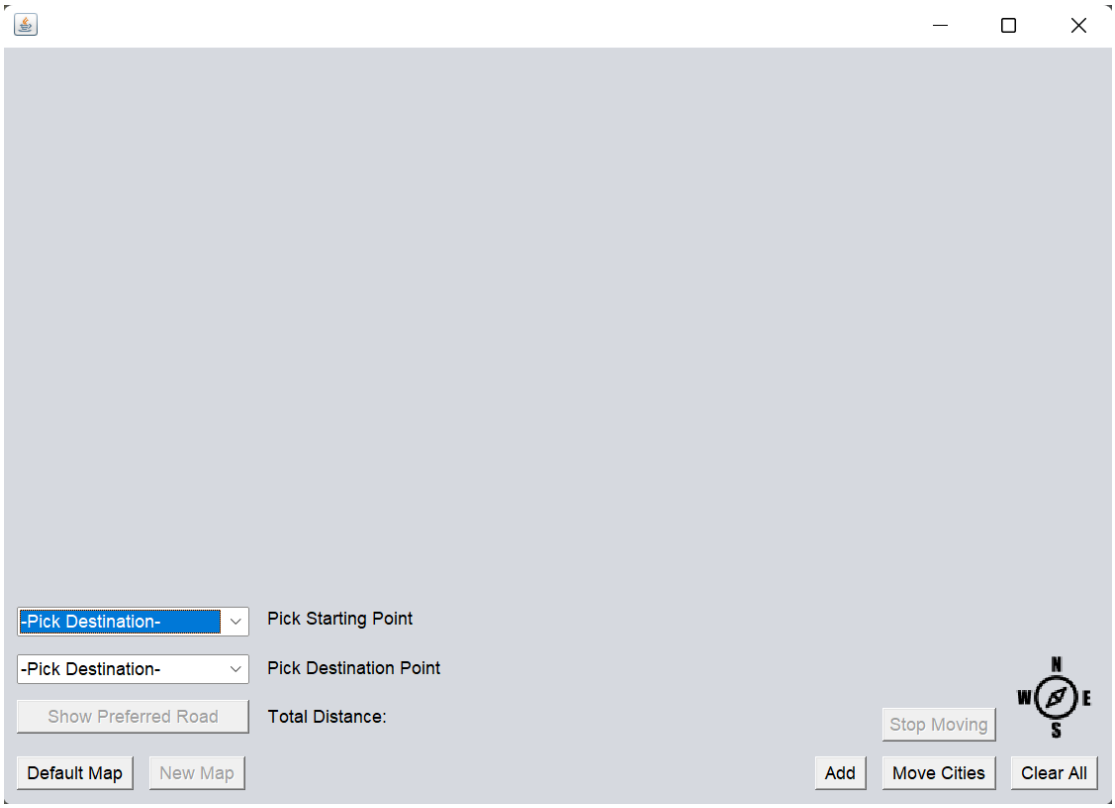
Όπως παρατηρείτε στον πίνακα έχουμε στην πρώτη οριζόντια στήλη όλους τους κόμβους. Και, στην καθετή την σειρά με την οποία τους επισκεπτόμαστε. Οι κίτρινοι αριθμοί είναι οι ελάχιστες αποστάσεις που χρειάζεται για να μεταβεί κανείς στον κόμβο αναζήτησης . Και, τα μικρά κίτρινα νούμερα χρησιμεύουν ώστε να πάρουμε την διαδρομή με την οποία θα πάμε στο κάθε κόμβο. Απλά, διαλέγουμε τον κόμβο που θέλουμε και πάμε προς τα πίσω για να δούμε την διαδρομή που κάναμε από το 3.

Παράδειγμα, θέλουμε, να δούμε την διαδρομή από το 3 στο 4. Βλέπουμε, ότι, το 4 έχει κιτρινίσει στο 11_2 αρά θα πάμε να δούμε που σταμάτησε ο κόμβος 2 στο 3_3 και το 3 είναι ο κόμβος στον οποίο προήλθαμε αρά το συντομότερο μονοπάτι για το 4 είναι το 3->2->4 με ελάχιστη απόσταση το 11.

4ο Κεφάλαιο: Η χρήση της εφαρμογής

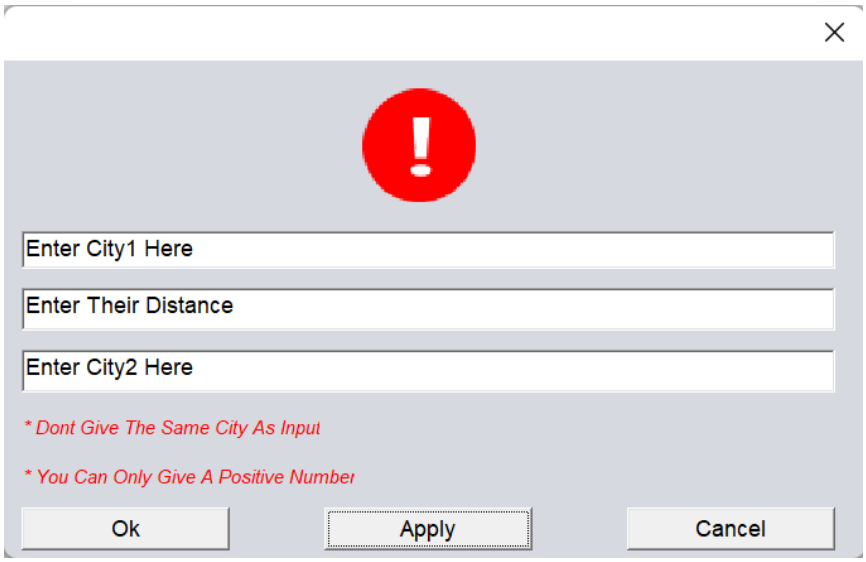
4.1 Τρόπος Λειτουργίας Εφαρμογής

Στο αρχικό παράθυρο της εφαρμογής δίνεται στον χρήστη οι εξής επιλογές εικόνα 4.1

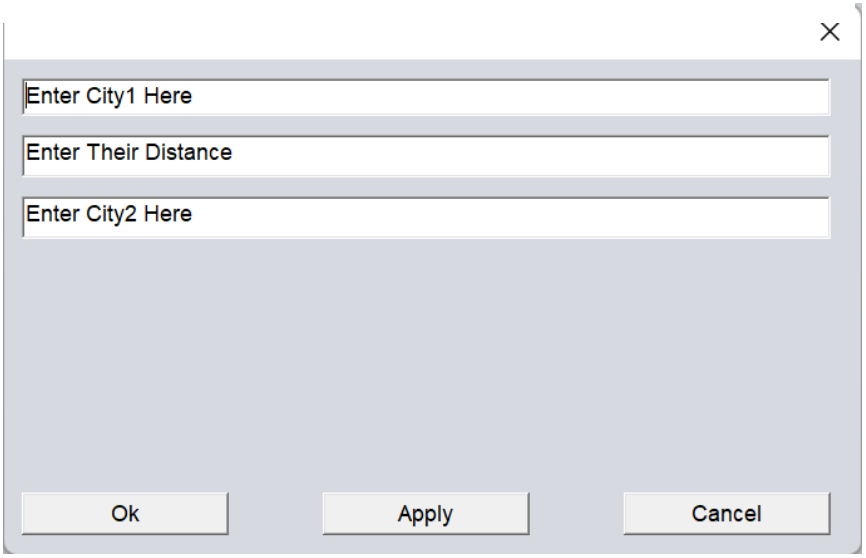


Εικόνα 4. 1

Η πρώτη είναι να προσθέσει στο παράθυρο τις πόλεις με τους δρόμους πατώντας το κουμπί με την ονομασία **Add**. Το οποίο θα του ανοίξει ένα καινούριο παράθυρο, το οποίο έχει τρία κουτιά κειμένου στα οποία μπορεί να εισάγει δυο πόλεις, και τον δρόμο που ενώνονται πάνω στο χάρτη - παράθυρο. Το καινούριο παράθυρο έχει τρεις επιλογές την **Ok**, **Apply**, και την **Cancel** στις οποίες μόλις πατηθούν γίνεται η διαδικασία εισαγωγής στον χάρτη, εισαγωγή πολλών πόλεων, και ακύρωσης αντιστοίχως. Επιπλέον, δεν επιτρέπει στον χρήστη να έχει πρόσβαση στο αρχικό παράθυρο, και στην περίπτωση που βάλει αρνητικές τιμές για τους δρόμους ή δώσει την ίδια πόλη τότε, δεν δέχεται την εισαγωγή και εμφανίζει ανάλογο μήνυμα στον χρήστη με κόκκινα γράμματα. Μόλις επιλέξει την επιλογή **Ok**, τότε ο χρήστης τοποθετεί τις πόλεις πάνω στο κομμάτι του παραθύρου που του επιτρέπεται εικόνες 4.2 και 4.3.



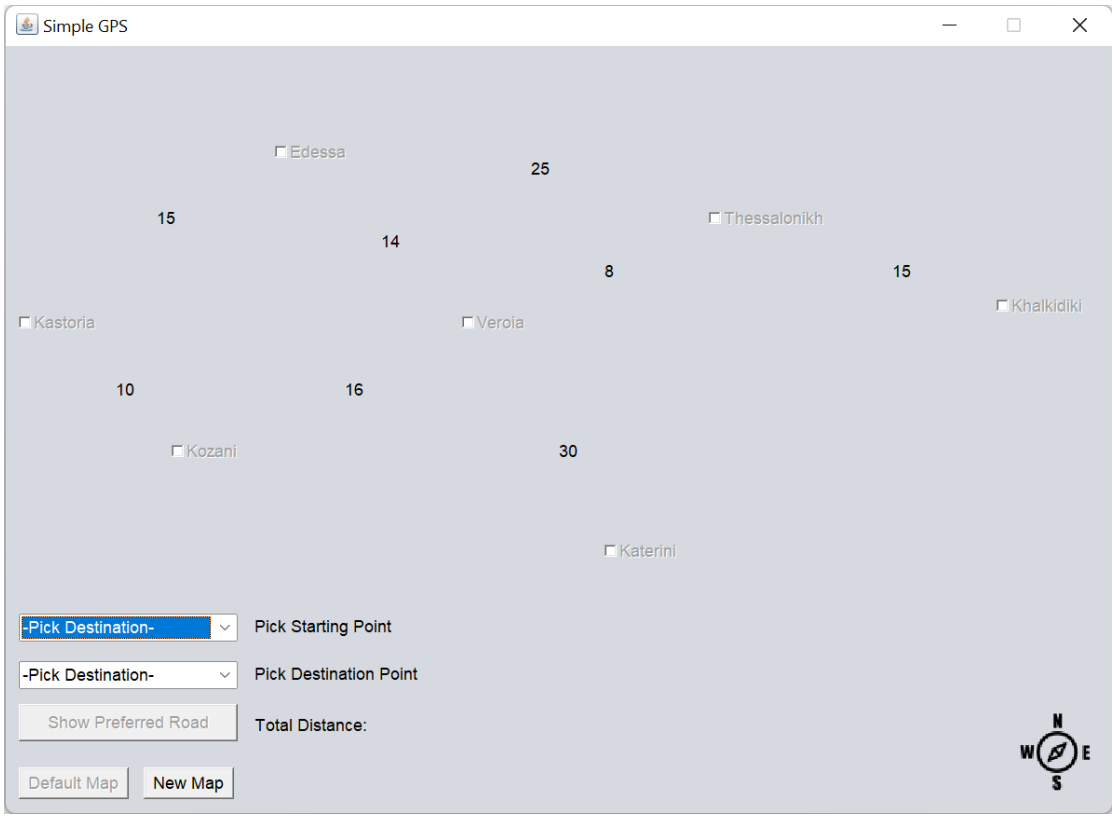
Εικόνα 4. 2



Εικόνα 4. 3

Τη στιγμή που θα τοποθετηθούν οι πόλεις τότε όλες οι αποστάσεις τους εμφανίζονται σε μορφή ενός αριθμού αναμεσά στις κάθε αντίστοιχες πόλεις. Αφότου εισαχθούν οι πόλεις ο χρήστης έχει τις επιλογές να διαλέξει την πόλη αφετηρίας και την πόλη προορισμού οι οποίες παίρνουν μια πράσινη και κίτρινη απόχρωση αντιστοίχως. Έπειτα η επιλογή **Show Preferred Road** γίνεται ενεργή και ο χρήστης βλέπει την βέλτιστη διαδρομή χάρη στον αλγόριθμο του **Dijkstra**, μαζί με την συνολική απόσταση που θα χρειαστεί να διανύσει. Ο χρήστης έχει τις επιλογές επιπλέον, να μετακινήσει τις πόλεις πάνω στο χάρτη με την επιλογή **Move Cities**, και να σταματήσει αυτή τη λειτουργία με την Stop Moving. Η επιλογή **Clear All** διαγραφεί όλες τις πόλεις για να αρχίσει ο χρήστης από το μηδέν. Τέλος, ο χρήστης μπορεί να μεταβεί σε έναν προκαθορισμένο χάρτη με την επιλογή **Default Map** και να επιστρέψει με την επιλογή **New Map**. Στον προκαθορισμένο χάρτη ο χρήστης έχει μόνο τις επιλογές ώστε να διαλέξει πόλεις και να δει την βέλτιστη διαδρομή εικόνα

4.4.



Εικόνα 4. 4

4.2 Βασικά κομμάτια κώδικα

Ο παρακάτω κώδικας με την ονομασία DijkstraAlgo είναι ο αλγόριθμος του Dijkstra και είναι ο κώδικας που χρησιμοποιείται για να βρεθεί η βέλτιστη διαδρομή πάνω σε γράφους δηλαδή στις πόλεις της εφαρμογής. Χρησιμοποιεί λίστες γειτνίασης και μια ουρά προτεραιότητας για να επιτύχει τον σκοπό του. Για να λειτουργήσει πρέπει να υπάρχει ένας σταθμισμένος γράφος και οι κόμβοι να είναι ενωμένοι μεταξύ τους[6][7].

```
//This Function Finds The Shortest Route Possible At All Possible Destinations
        void DijkstraAlgo(final E src)
        {
            if(graph.isEmpty())
            {
                System.out.println("Initialize a graph");
                return;
            }
```

```
data = new ArrayList<>(); //Showing the minimum distance
datadest = new ArrayList<>();//Showing the road


for(int itr = 0; itr < graph.size(); itr++)
{
        ArrayList<E> init = new ArrayList<>();
        datadest.add(itr, init);
        if(itr == graph.getfromindexcounter(src))
        {
                data.add(itr ,0); //Src = 0
        }else{
                data.add(itr, Integer.MAX_VALUE);
        }
}

PriorityQueue<Pair<Integer>> mypq = new PriorityQueue<>(new Comparator_Pair());
mypq.add(new Pair(graph.getfromindexcounter(src), 0));
Integer u, v, weight, i;
E vconvert;

while(!mypq.isEmpty())
{
        u = mypq.poll().getLeft();

        for(i = 0; i < graph.get(u).size(); i++)
        {
                vconvert = graph.get(u).getindex(i);
                v = graph.getfromindexcounter(vconvert);
                weight = graph.get(u).get(i);
```

```
                    if(data.get(v) > data.get(u) + weight)
                    {
                        Object newpath = datadest.get(u).clone();
                        ArrayList<E> newpathobj = (ArrayList<E>) newpath; //deep copy
                        newpathobj.add(vconvert);
                        datadest.set(v, newpathobj); // The Destination path

                        data.set(v, data.get(u) + weight); //The Destination total distance
                        mypq.add(new Pair(v, data.get(v)));
                    }

                }
            }
        }
```

Ο παρακάτω κώδικας προσθέτει τους κόμβους του γράφου ανά μία κατεύθυνση και έχει την ονομασία AddOnedirectional. Επιπλέον συγκρίνει αν τα δεδομένα είναι αποδεκτά για να υλοποιηθεί σωστά ο αλγόριθμος. Η εφαρμογή καλεί δύο φορές την παρακάτω μέθοδο ώστε να μπορεί ο χρήστης να πηγαίνει και να επιστρέφει από οπουδήποτε [5].

```
//This Function Adds A Road To Two Cities But At Only One Direction
        void AddOnedirectional(final E Start, final E End, Integer dist)
        {
            if(dist.compareTo(0) <= 0 || Start.equals(End))
            {return;}

            if(!graph.containsIndex(Start))
            {
                DyArray<Integer, E> Arrobj = new DyArray<>();
                Arrobj.addElementAt(End, dist);
                graph.addElementAt(Start, Arrobj);
```

```
                }else{
                        if(graph.getfromindex(Start).containsIndex(End))
                        {

        graph.getfromindex(Start).setElementAt(End, dist);
                        }else
                        {
                                DyArray<Integer, E> Arrobj =
graph.getfromindex(Start);
                                Arrobj.addElementAt(End, dist);
                                graph.addElementAt(Start, Arrobj);
                        }
                }
        }
```

Η μέθοδος TikCheckBoxes καλείται κάθε φορά που χρήστης επιλέγει μια πόλη και εγγυείται ότι οι σωστές πόλεις θα είναι επιλεγμένες στο παραθυρικό περιβάλλον [5][1][3].

```
//This Function Gets Called With Every Choice Selection Thats Happening
        //To Show Which Checkbox Is Selected To The User
        public void TikCheckBoxes(java.awt.Choice cho)
        {
                /**/
                if(cho.getItem(0).equals("-Pick Destination-"))
                {cho.remove(0);}

                ArrayList<java.awt.Checkbox> city; //= getcitydata(cho);
                java.awt.Choice othcho;
                int i1, i2;
                Color c1, c2;
                if(cho == choice1 || cho == choice2)
                {
                        if(cho == choice1)
                        {
                                i1 = 0;
                                i2 = 1;
```

```
                othcho = choice2;
        }else
        {
                i1 = 1;
                i2 = 0;
                othcho = choice1;
        }
        city = citydata;
}else
{
        if(cho == choice3)
        {
                i1 = 2;
                i2 = 3;
                othcho = choice4;
        }else
        {
                i1 = 3;
                i2 = 2;
                othcho = choice3;
        }
        city = citydata2;
}
if(i1 % 2 == 0)
{
        c1 = Color.green;
        c2 = Color.yellow;
}else
{
        c1 = Color.yellow;
        c2 = Color.green;
}

int a = cho.getSelectedIndex();
```

```
//If choices are equals
if (cho.getSelectedItem().equals(othcho.getSelectedItem()))
{
	othcho.removeAll();
	othcho.add("-Pick Destination-");
	othcho.select(0);
	if(i1 < 2)
	{
		button5.setEnabled(false);
	}else
	{
		button6.setEnabled(false);
	}
	for(int y = 0; y < city.size(); y++)
	{
		othcho.add(city.get(y).getLabel());
	}
	states2.get(i1).setState(false);
	states2.get(i1).setBackground(this.getBackground());
	states2.set(i1, city.get(a));
	states2.get(i1).setState(true);
	states2.get(i1).setBackground(c1);

	states2.set(i2, new Checkbox());
	return;
}
```

Η παρακάτω μέθοδος showroad βοηθάει στο να εμφανίσει την σωστή συνολική απόσταση μαζί και να βάλει τα κατάλληλα χρώματα στις πόλεις αφετηρίας και προορισμού [3].

```
//This Function Shows The Road At The Default And New Mapspublic void showroad(java.awt.event.ActionEvent evt)
	{
		Graph dij;
```

```
int st1, st2;
java.awt.Label leb;

if(this.button5 == evt.getSource())
{
        dij = obj;
        st1 = 0;
        st2 = 1;
        leb = label12;
}else
{
        dij = obj2;
        st1 = 2;
        st2 = 3;
        leb = label15;
}

dij.DijkstraAlgo(states2.get(st1));
Pair<ArrayList<java.awt.Checkbox>> data = dij.GetData(states2.get(st2));
leb.setText(data.getRight().toString());

if(!data.getRight().equals(0))
{
        for(int i = 0; i < data.getLeft().size() -1; i++)
        {
                data.getLeft().get(i).setState(true);
                data.getLeft().get(i).setBackground(Color.orange);

                try{
                TimeUnit.SECONDS.sleep(1);
                }
                catch(InterruptedException e){
                        System.err.println(e);
                }
```

```
                    data.getLeft().get(i).setState(false);
                    data.getLeft().get(i).setBackground(this.getBackground());
                }

                states2.get(st2).setBackground(Color.orange);
                try{
                    TimeUnit.SECONDS.sleep(1);
                    }
                    catch(InterruptedException e){
                        System.err.println(e);
                    }
                states2.get(st2).setState(true);
                states2.get(st2).setBackground(Color.yellow);
            }
        }
```

Η επόμενη μέθοδος ελέγχει τις τιμές που λήφθηκαν από το μικρότερο παράθυρο της εφαρμογής, που εμφανίζεται όταν ο χρήστης προσθέτει καινούριες πόλεις. Και βοηθάει ώστε η κλάση Graph να έχει αποδεκτές τιμές.

```
//This Function Helps At Apply And Ok Button To Check If The Values
 //Are Correct And Display The Appropriate Error Messages To jDialog1
    public void applychecksfromtextfields()
    {

        String t1, t2;
        Integer d;
        t1 = textField1.getText();
        t2 = textField2.getText();

        if(t1.equals(t2))
        {
            jLabel1.setVisible(true);
            jLabel2.setVisible(true);
            textField1.setText("Enter City1 Here");
```

```
            textField2.setText("Enter City2 Here");
            distancecond();
            return;
}

d = distancecond();
if (d.equals(-1))
{
            return;
}



java.awt.Checkbox check1 = new java.awt.Checkbox();
java.awt.Checkbox check2 = new java.awt.Checkbox();

for(int i = 0; i < obj2.SizeOfNodes(); i++)
{
            Checkbox t = (Checkbox) obj2.getIndex(i);
            if(t.getLabel().equals(t1))
            {
                        check1 = t;
            }
            if(t.getLabel().equals(t2))
            {
                        check2 = t;
            }
}

for(int i = 0; i < objt.SizeOfNodes(); i++)
{
            Checkbox t = (Checkbox) objt.getIndex(i);
            if(t.getLabel().equals(t1))
            {
```

```
                check1 = t;
            }
            if(t.getLabel().equals(t2))
            {
                check2 = t;
            }
        }

        if(check1.getLabel().isEmpty())
        {
            check1 = new java.awt.Checkbox(t1);
            check1.setEnabled(false);
            check1.setState(false);
            check1.setSize(check1.getLabel().length() * 8 + 10, 20);
        }
        if(check2.getLabel().isEmpty())
        {
            check2 = new java.awt.Checkbox(t2);
            check2.setEnabled(false);
            check2.setState(false);
            check2.setSize(check2.getLabel().length() * 8 + 10, 20);
        }

        objt.AddBidirectional(check1, check2, d);
        if(jLabel1.isVisible())
        {
            jLabel1.setVisible(false);
            jLabel2.setVisible(false);
            jLabel5.setVisible(false);

            textField1.setText("Enter City1 Here");
            textField2.setText("Enter City2 Here");
            textField3.setText("Enter Their Distance");
        }
```

```
}
```

Η επόμενη μέθοδος με την ονομασία gethalfdist βοηθάει ώστε να μπουν ακριβώς ανάμεσα στις δύο πόλεις η ταμπέλα της συνολικής απόστασης του δρόμου.

```
//This Function Helps To Add The Roads Of Cities
    Point gethalfdist(Point p1, Point p2)
    {
        int x, y;
        x = (p1.x - p2.x)/2;
        if(x >= 0)
        {
            x = x + p2.x;
        }else
        {
            x = p1.x - x;
        }
        y = (p1.y - p2.y)/2;
        if(y >= 0)
        {
            y = y + p2.y;
        }else
        {
            y = p1.y - y;
        }
        Point p = new Point(x, y);
        return p;
    }
```

Συμπεράσματα και μελλοντική εργασία

Το αντικείμενο της παρούσας πτυχιακής είναι η υλοποίηση gps εφαρμογής σε παραθυρικό περιβάλλον με χρήση της αντικειμενοστραφής γλώσσας προγραμματισμού Java. Η εκπόνηση της πτυχιακής διεκπεραιώθηκε χάρης στο προγραμματιστικό περιβάλλον Netbeans το οποίο κατείχε χρήσιμα εργαλεία για την γρήγορη υλοποίηση της εφαρμογής τα οποία έχουν την ονομασία Swing και Awt, και χάρη στις παρατηρήσεις του επιβλέποντα καθηγητή, κύριο Παναγιώτη Μπάτο.

Γενικός η χρήση του συστήματος μιας gps εφαρμογής παρέχει μια σειρά από σημαντικά πλεονεκτήματα ως προς τον χρήστη, που του παρέχουν αυτοματοποιημένες λύσεις, τα οποία είναι η εύρεση της βέλτιστης διαδρομής ανάμεσα σε δύο σημεία, η εύρεση της παγκόσμιας θέσης του χρήστη πάνω στο χάρτη, και πολλές γεωγραφικές πληροφορίες όπως αξιοθέατα, μαγαζιά, και κτήρια έκτακτης ανάγκης.

Υπάρχουν κι όμως, μερικά μικρά μειονεκτήματα ωστόσο που συμπέρανα στα gps που είναι ότι ορισμένες φορές υπάρχει αδυναμία στην ακρίβεια εντοπισμού ενός οχήματος μέσω gps, και ορισμένοι υπάλληλοι ίσως να νοιώθουν ότι προσβάλλεται η ελευθερία τους διότι υπάρχει παρακολούθηση προσωπικών δεδομένων. Ωστόσο στην πραγματικότητα η ασφάλεια, η αυτοματοποίηση και η καθοδήγηση των gps είναι ο κύριος σκοπός τους και όχι η παρακολούθηση των πελατών τους. Επιπλέον, πρέπει η διεπαφή να καλύπτει ένα μεγάλο γεωγραφικό φάσμα ώστε να δίνεται η δυνατότητα στο όχημα να δίνει στίγμα σε κάθε πιθανό σημείο ανεξάρτητα της χιλιομετρικής απόσταση ή την γεωγραφική του θέση. Το πρόβλημα αυτό μπορεί να αντιμετωπιστεί καταχωρώντας σε μια βάση δεδομένων κάθε φορά το γεωγραφικό μήκος και πλάτος ως συντεταγμένες οι οποίες θα καλούνται και θα τοποθετούνται πάνω στο χάρτη που θα δημιουργηθεί. Με αυτόν τον τρόπο, θα είμαστε σε θέση να λαμβάνουμε το στίγμα του οχήματος κάθε στιγμή σε πραγματικό χρόνο.

Η πτυχιακή με βοήθησε στο να κατανοήσω καλύτερα τον αλγόριθμο του Dijsktra, ο οποίος είναι ένας άπληστος αλγόριθμος, και γενικός πως αναπαρίστανται οι γράφοι σε γλώσσες προγραμματισμού, με τη χρήση μιας δυάδας και δύο πινάκων, καθώς και παραθυρικές εφαρμογές που μπορούν να αναπτυχθούν στην Java με την χρήση των Swing και Awt. Πλέον, χάρη στην εκπόνηση της πτυχιακής είμαι ικανός

να διεκπεραιώσω εφαρμογές με τη χρήση διαφόρων αντικειμενοστραφών γλωσσών προγραμματισμού με μεγαλύτερη αυτοπεποίθηση, και ταχύτητα.

Ως προς την μελλοντική ανάπτυξη της παρούσας εργασίας, θα μπορούσε να βελτιστοποιηθεί ο αλγόριθμος αναζήτησης της διαδρομής, με το να υπάρχει αποθηκευμένα στη μνήμη μια βάσης δεδομένων. Με όλους τους κόμβους, και η εφαρμογή να επιλέγει στο παρασκήνιο συγκεκριμένους κόμβους που βρίσκονται προσεγγιστικά σε μια γραμμή η αφετηρία και η πόλη προορισμού. Επιπλέον, θα μπορούσε η εφαρμογή να κάνει χρήση ενός χάρτη που χρησιμοποιεί μπάρες μετακίνησης, ώστε να χωρέσουν περισσότερες πόλεις μαζί με τους δρόμους τους. Με αυτό τον τρόπο γίνεται πιο ευρείς ο χάρτης και μπορούμε να συμπεριλάβουμε ολόκληρο τον κόσμο. Τέλος, υπάρχει η δυνατότητα να τοποθετηθούν δύο κουμπιά μεγέθυνσης και συρρίκνωσης τα οποία δείχνουν τον χάρτη σε διαφορετική κλίμακα. Δηλαδή θα φαίνεται διαφορετικά η λεπτομέρεια, αναλόγως το πόσο μακριά ή το πόσο κοντά βρίσκεται το φάσμα του χάρτη σε συνάρτηση με την κάμερα [7][3].

Βιβλιογραφία

[1] Walter Savitch “Απόλυτη Java” μετάφραση: Δημήτρης Ιακωβίδης https://www.politeianet.gr/books/9789604116454-savitch-walter-ion-apoluti-java-periechei-cd-rom-148948

[2] Paul Deitel, Harvey Deitel “Java Προγραμματισμός” Δέκατη Έκδοση μετάφραση: Γκλάβα Μαρία https://www.politeianet.gr/books/9789605126810-deitel-m-harvey-gkiourdas-m-java-programmatismos-109290

[3] Walter Savitch “Java μια εισαγωγή στην επίλυση προβλημάτων και στον προγραμματισμό” έβδομη έκδοση μετάφραση Αλέξανδρος Χατζηγεωργίου https://www.politeianet.gr/books/9789604185016-savitch-walter-tziolas-java-308745

[4] Alan Dennis, Barbara Wixom, David Tegarden “Ανάλυση & σχεδιασμός συστημάτων με την uml 2.0 μια αντικειμενοστραφής προσέγγιση” μετάφραση Καρτσακλής Δημήτρης https://biblionet.gr/titleinfo/?titleid=159858

[5] Λιακέας Γιώργος “Εισαγωγή στη Java” δεύτερη έκδοση https://www.politeianet.gr/books/9789602094310-liakeas-giorgos-kleidarithmos-eisagogi-stin-java-70493

[6] Robert Sedgewick “Αλγόριθμοι σε Java Μέρη 1-4 θεμελιώδεις έννοιες: Δομές Δεδομένων: Ταξινόμηση: Αναζήτηση ” https://www.public.gr/product/books/greek-books/computer-science/programming/algorithmoi-se-java/1607465

[7] Κλεάνθης Θραμπουλίδης “Αντικειμενοστραφής Προγραμματισμός στη -Java” Τρίτη Έκδοση https://metabook.gr/books/java-antikeimenostrafis-proghrammatismos-3i-ekdosi-kleanthis-thrampoylidis-299094

[8] Αλέξανδρος Χατζηγεωργίου “Αντικειμενοστραφής Σχεδίαση UML, Αρχές, Πρότυπα και Ερευνητικοί Κανόνες” https://www.politeianet.gr/books/9789602098820-chatzigeorgiou-alexandros-kleidarithmos-antikeimenostrefis-schediasi-148790

Παράρτημα Κώδικα

Simple_GPS Class

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */
package com.mycompany.gps_project;

import java.awt.Checkbox;
import java.util.ArrayList;
import java.awt.Color;
import java.awt.Font;
import java.awt.Image;
import java.util.concurrent.TimeUnit;
import java.awt.Point;
import java.awt.Rectangle;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;
import javax.swing.ImageIcon;

/**
* @author George Kousidis
 */
public class Simple_GPS extends javax.swing.JFrame {

        /**
```

```
 * Creates new form Simple_GPS
 */
//The Constructor Where We Initialise
//And Set All The Data At The Start Of The Program
public Simple_GPS() {
initComponents();
citydata = getCheckBoxes();
//Integer.parseInt(s);
//Integer.valueOf(s);
obj.AddBidirectional(citydata.get(0), citydata.get(1),
Integer.valueOf(label6.getText())); // Veroia - Thess 8
obj.AddBidirectional(citydata.get(0), citydata.get(3),
Integer.valueOf(label9.getText())); // Veroia - Katerini 30
obj.AddBidirectional(citydata.get(0), citydata.get(4),
Integer.valueOf(label3.getText())); // Veroia - Kozani 16
obj.AddBidirectional(citydata.get(0), citydata.get(6),
Integer.valueOf(label8.getText())); // Veroia - Edessa 14
obj.AddBidirectional(citydata.get(1), citydata.get(5),
Integer.valueOf(label10.getText()));// Thessaloniki - Khalkidiki 15
obj.AddBidirectional(citydata.get(1), citydata.get(6),
Integer.valueOf(label7.getText())); // Thessaloniki - Edessa 25
obj.AddBidirectional(citydata.get(2), citydata.get(4),
Integer.valueOf(label4.getText())); // Kastoria - Kozani 10
obj.AddBidirectional(citydata.get(2), citydata.get(6),
Integer.valueOf(label5.getText())); // Kastoria - Edessa 15


choice1.add("-Pick Destination-");
choice2.add("-Pick Destination-");
choice3.add("-Pick Destination-");
```

```
choice4.add("-Pick Destination-");

choice1.select(0);
choice2.select(0);
choice3.select(0);
choice4.select(0);

for(int i = 0; i < citydata.size(); i++)
{
	choice1.add(citydata.get(i).getLabel());
	choice2.add(citydata.get(i).getLabel());
}

for(int i = 0; i < 4; i++)
{
	states2.add(i, new java.awt.Checkbox());
}

try{
	System.out.println(new File(".").getCanonicalPath());
}catch(IOException e)
{
	e.printStackTrace();
}


//jLabels
//Icon For Compass
BufferedImage img = null;
try {
```

```
        img = ImageIO.read(new
File("./src/main/java/com/mycompany/gps_project/Compass Icon.png"));
} catch (IOException e) {
        e.printStackTrace();
}

Image dimg = img.getScaledInstance(60, 60, Image.SCALE_SMOOTH);

ImageIcon imageIcon = new ImageIcon(dimg);

jLabel3.setIcon(imageIcon);
jLabel4.setIcon(imageIcon);



try {
        img = ImageIO.read(new
File("./src/main/java/com/mycompany/gps_project/Wrong Input Icon.png"));
} catch (IOException e) {
        e.printStackTrace();
}

dimg = img.getScaledInstance(60, 60,
Image.SCALE_SMOOTH);

ImageIcon imageIcon2 = new ImageIcon(dimg);
jLabel1.setIcon(imageIcon2);
jLabel1.setBounds(jLabel1.getBounds());
jLabel1.setVisible(false);

Font f = new Font("Calibri (Body)", Font.ITALIC, 10);
```

```
jLabel2.setFont(f);
jLabel2.setForeground(Color.red);
jLabel2.setVisible(false);
jLabel5.setFont(f);
jLabel5.setForeground(Color.red);
jLabel5.setVisible(false);
}

//This Function Adds The Default Cities In A Container That The Program
Uses
public ArrayList<java.awt.Checkbox> getCheckBoxes()
{
ArrayList<java.awt.Checkbox> list = new ArrayList<>();
list.add(checkbox1); //veroia
list.add(checkbox2); //thessaloniki
list.add(checkbox3); //kastoria
list.add(checkbox4); //katerini
list.add(checkbox5); //kozani
list.add(checkbox6); //khalkidiki
list.add(checkbox7); //edessa

return list;
}

//This Function Gets Called With Every Choice Selection Thats Happening
//To Show Which Checkbox Is Selected To The User
public void TikCheckBoxes(java.awt.Choice cho)
{
/**/
if(cho.getItem(0).equals("-Pick Destination-"))
{cho.remove(0);}
```

```
ArrayList<java.awt.Checkbox> city; //= getcitydata(cho);
java.awt.Choice othcho;
int i1, i2;
Color c1, c2;
if(cho == choice1 || cho == choice2)
{
        if(cho == choice1)
        {
        i1 = 0;
        i2 = 1;
        othcho = choice2;
        }else
        {
        i1 = 1;
        i2 = 0;
        othcho = choice1;
        }
        city = citydata;
}else
{
        if(cho == choice3)
        {
        i1 = 2;
        i2 = 3;
        othcho = choice4;
        }else
        {
        i1 = 3;
        i2 = 2;
```

```
            othcho = choice3;
            }
            city = citydata2;
}
if(i1 % 2 == 0)
{
            c1 = Color.green;
            c2 = Color.yellow;
}else
{
            c1 = Color.yellow;
            c2 = Color.green;
}

int a = cho.getSelectedIndex();

//If choices are equals
if (cho.getSelectedItem().equals(othcho.getSelectedItem()))
{
            othcho.removeAll();
            othcho.add("-Pick Destination-");
            othcho.select(0);
            if(i1 < 2)
            {
            button5.setEnabled(false);
            }else
            {
            button6.setEnabled(false);
            }
            for(int y = 0; y < city.size(); y++)
```

```
        {
        othcho.add(city.get(y).getLabel());
        }
        states2.get(i1).setState(false);
        states2.get(i1).setBackground(this.getBackground());
        states2.set(i1, city.get(a));
        states2.get(i1).setState(true);
        states2.get(i1).setBackground(c1);

        states2.set(i2, new Checkbox());
        return;
}


int b = othcho.getSelectedIndex();

//Not equals from here on
states2.get(i1).setState(false);
states2.get(i1).setBackground(this.getBackground());
states2.set(i1, city.get(a));
states2.get(i1).setState(true);
states2.get(i1).setBackground(c1);

if(!othcho.getItem(b).equals("-Pick Destination-"))
{
        states2.get(i2).setState(false);
        states2.get(i2).setBackground(this.getBackground());
        states2.set(i2, city.get(b));
        states2.get(i2).setState(true);
        states2.get(i2).setBackground(c2);
```

```
}

ifcho(cho);
}

//This Function Enables And Disables The Show Road
//Buttons From The TickCheckBoxes Function
public void ifcho(java.awt.Choice cho)
{
if (cho == choice1 || cho == choice2)
{
        if(cho == choice1)
        {
        if(choice2.getItem(0) != "-Pick Destination-")
        {
                button5.setEnabled(true);
        }
        }else
        {
        if(choice1.getItem(0) != "-Pick Destination-")
        {
                button5.setEnabled(true);
        }
        }
}else
{
        if(cho == choice3)
        {
        if(choice4.getItem(0) != "-Pick Destination-")
```

```
        {
                button6.setEnabled(true);
        }
        }else
        {
        if(choice3.getItem(0) != "-Pick Destination-")
        {
                button6.setEnabled(true);
        }
        }
}

}

//This Function Shows The Road At The Default And New Maps
public void showroad(java.awt.event.ActionEvent evt)
{
Graph dij;
int st1, st2;
java.awt.Label leb;

if(this.button5 == evt.getSource())
{
        dij = obj;
        st1 = 0;
        st2 = 1;
        leb = label12;
}else
{
        dij = obj2;
```

```
        st1 = 2;
        st2 = 3;
        leb = label15;
}

dij.DijkstraAlgo(states2.get(st1));
Pair<ArrayList<java.awt.Checkbox>> data = dij.GetData(states2.get(st2));
leb.setText(data.getRight().toString());

if(!data.getRight().equals(0))
{
        for(int i = 0; i < data.getLeft().size() -1; i++)
        {
        data.getLeft().get(i).setState(true);
        data.getLeft().get(i).setBackground(Color.orange);

        try{
        TimeUnit.SECONDS.sleep(1);
        }
        catch(InterruptedException e){
                System.err.println(e);
        }

        data.getLeft().get(i).setState(false);
        data.getLeft().get(i).setBackground(this.getBackground());
        }

        states2.get(st2).setBackground(Color.orange);
        try{
        TimeUnit.SECONDS.sleep(1);
```

```
        }
        catch(InterruptedException e){
                System.err.println(e);
        }
        states2.get(st2).setState(true);
        states2.get(st2).setBackground(Color.yellow);
}
}

//This Function Is Inside The applychecksfromtextfields Function And It
//Checks Whether The Distance Is An Integer Or Not
public Integer distancecond()
{
Integer d;
String temp = textField3.getText();

for(int i = 0; i < temp.length(); i++)
{
        char ch = temp.charAt(i);
        if(ch != '0' && ch != '1'
        && ch != '2' && ch != '3'
        && ch != '4' && ch != '5'
        && ch != '6' && ch != '7'
        && ch != '8' && ch != '9')
        {
        jLabel1.setVisible(true);
        jLabel5.setVisible(true);
        textField3.setText("Enter Their Distance");
        return -1;
        }
```

```
}

d = Integer.valueOf(temp);

if( d.equals(0))
{
        jLabel1.setVisible(true);
        jLabel5.setVisible(true);
        textField3.setText("Enter Their Distance");
        return -1;
}

return d;
}

//This Function Helps At Apply And Ok Button To Check If The Values
//Are Correct And Display The Appropriate Error Messages To jDialog1
public void applychecksfromtextfields()
{

String t1, t2;
Integer d;
t1 = textField1.getText();
t2 = textField2.getText();

if(t1.equals(t2))
{
        jLabel1.setVisible(true);
        jLabel2.setVisible(true);
        textField1.setText("Enter City1 Here");
```

```
        textField2.setText("Enter City2 Here");
        distancecond();
        return;
}

d = distancecond();
if (d.equals(-1))
{
        return;
}



java.awt.Checkbox check1 = new java.awt.Checkbox();
java.awt.Checkbox check2 = new java.awt.Checkbox();

for(int i = 0; i < obj2.SizeOfNodes(); i++)
{
        Checkbox t = (Checkbox) obj2.getIndex(i);
        if(t.getLabel().equals(t1))
        {
        check1 = t;
        }
        if(t.getLabel().equals(t2))
        {
        check2 = t;
        }
}

for(int i = 0; i < objt.SizeOfNodes(); i++)
```

```
{
        Checkbox t = (Checkbox) objt.getIndex(i);
        if(t.getLabel().equals(t1))
        {
        check1 = t;
        }
        if(t.getLabel().equals(t2))
        {
        check2 = t;
        }
}

if(check1.getLabel().isEmpty())
{
        check1 = new java.awt.Checkbox(t1);
        check1.setEnabled(false);
        check1.setState(false);
        check1.setSize(check1.getLabel().length() * 8 + 10, 20);
}
if(check2.getLabel().isEmpty())
{
        check2 = new java.awt.Checkbox(t2);
        check2.setEnabled(false);
        check2.setState(false);
        check2.setSize(check2.getLabel().length() * 8 + 10, 20);
}

objt.AddBidirectional(check1, check2, d);
if(jLabel1.isVisible())
{
```

```
        jLabel1.setVisible(false);
        jLabel2.setVisible(false);
        jLabel5.setVisible(false);

        textField1.setText("Enter City1 Here");
        textField2.setText("Enter City2 Here");
        textField3.setText("Enter Their Distance");
}
}

//This Function Helps To Checks If The Cursor Is On Top Of A City
Boolean iscollided(Point p)
{
if(p.y + 20 >= choice3.getLocation().y)
{
        return true;
}
for(int i = 0; i < collide.size(); i++)
{
        int x, y, width, height;
        x = collide.get(i).x;
        y = collide.get(i).y;
        width = collide.get(i).width;
        height = collide.get(i).height;

        if(p.x >= x && p.x <= width + x && p.y >= y && p.y <= height + y)
        {
        return true;
        }
}
```

```
return false;
}

//This Function Helps To Prevent Cities From Being On Top Of The Other
int getcollided(Point p)
{
for(int i = 0; i < collide.size(); i++)
{
        int x, y, width, height;
        x = collide.get(i).x;
        y = collide.get(i).y;
        width = collide.get(i).width;
        height = collide.get(i).height;

        if(p.x >= x && p.x <= width + x && p.y >= y && p.y <= height + y)
        {
        return i;
        }
}
return -1;
}

//This Function Helps To Add The Cities On The jFrame2's Map
void movecheck(Checkbox thischeck, Point p)
{
if(!iscollided(p))
        {
        thischeck.setLocation(p);
        this.jFrame2.add(thischeck);
        }
```

```
}

//This Function Repositions The Labels On The jFrame2's Map Accordingly
void movelabel()
{
//Label
int ind = obj2.getfromElement(temp.get(0));
Point p1 = temp.get(0).getLocation();
for(int itr = 0;itr < Arrlabel.get(ind).size(); itr++)
{
        Checkbox ch = (Checkbox) obj2.get(ind).getindex(itr);
        Point p = gethalfdist(p1, ch.getLocation());
        Arrlabel.get(ind).get(itr).setLocation(p);
}//Label
temp.remove(0);
movetemp = false;
}

//This Function Helps To Add The Road's Of Cities
Point gethalfdist(Point p1, Point p2)
{
int x, y;
x = (p1.x - p2.x)/2;
if(x >= 0)
{
        x = x + p2.x;
}else
{
        x = p1.x - x;
}
```

```
        y = (p1.y - p2.y)/2;
        if(y >= 0)
        {
                y = y + p2.y;
        }else
        {
                y = p1.y - y;
        }
        Point p = new Point(x, y);
        return p;
        }


/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
        @SuppressWarnings("unchecked")
        // <editor-fold defaultstate="collapsed" desc="Generated Code">

        private void initComponents() {

        jFrame2 = new javax.swing.JFrame();
        choice3 = new java.awt.Choice();
        choice4 = new java.awt.Choice();
        label13 = new java.awt.Label();
        label14 = new java.awt.Label();
        label15 = new java.awt.Label();
        label16 = new java.awt.Label();
        button3 = new java.awt.Button();
```

```
button4 = new java.awt.Button();
button6 = new java.awt.Button();
button7 = new java.awt.Button();
button8 = new java.awt.Button();
button9 = new java.awt.Button();
button10 = new java.awt.Button();
jLabel3 = new javax.swing.JLabel();
jDialog1 = new javax.swing.JDialog();
jPanel1 = new javax.swing.JPanel();
textField1 = new java.awt.TextField();
textField2 = new java.awt.TextField();
textField3 = new java.awt.TextField();
button11 = new java.awt.Button();
button12 = new java.awt.Button();
button13 = new java.awt.Button();
jLabel1 = new javax.swing.JLabel();
jLabel2 = new javax.swing.JLabel();
jLabel5 = new javax.swing.JLabel();
button1 = new java.awt.Button();
button2 = new java.awt.Button();
button5 = new java.awt.Button();
checkbox1 = new java.awt.Checkbox();
checkbox2 = new java.awt.Checkbox();
checkbox3 = new java.awt.Checkbox();
checkbox4 = new java.awt.Checkbox();
checkbox5 = new java.awt.Checkbox();
checkbox6 = new java.awt.Checkbox();
checkbox7 = new java.awt.Checkbox();
choice1 = new java.awt.Choice();
choice2 = new java.awt.Choice();
```

```
        label1 = new java.awt.Label();
        label2 = new java.awt.Label();
        label3 = new java.awt.Label();
        label4 = new java.awt.Label();
        label5 = new java.awt.Label();
        label6 = new java.awt.Label();
        label7 = new java.awt.Label();
        label8 = new java.awt.Label();
        label9 = new java.awt.Label();
        label10 = new java.awt.Label();
        label11 = new java.awt.Label();
        label12 = new java.awt.Label();
        jLabel4 = new javax.swing.JLabel();

        jFrame2.setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON
_CLOSE);
        jFrame2.setMinimumSize(new java.awt.Dimension(763, 547));
        jFrame2.addMouseMotionListener(new
java.awt.event.MouseMotionAdapter() {
                public void mouseMoved(java.awt.event.MouseEvent evt) {
                jFrame2MouseMoved(evt);
                }
        });
        jFrame2.addMouseListener(new java.awt.event.MouseAdapter() {
                public void mouseClicked(java.awt.event.MouseEvent evt) {
                jFrame2MouseClicked(evt);
                }
        });

        choice3.addItemListener(new java.awt.event.ItemListener() {
                public void itemStateChanged(java.awt.event.ItemEvent evt) {
```

```
        choice3ItemStateChanged(evt);
        }
});

choice4.addItemListener(new java.awt.event.ItemListener() {
        public void itemStateChanged(java.awt.event.ItemEvent evt) {
        choice4ItemStateChanged(evt);
        }
});

label13.setText("Total Distance:");

label14.setText("Pick Starting Point");

label16.setText("Pick Destination Point");

button3.setLabel("Default Map");
button3.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
        button3ActionPerformed(evt);
        }
});

button4.setEnabled(false);
button4.setLabel("New Map");

button6.setEnabled(false);
button6.setLabel("Show Preferred Road");
button6.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
        button6ActionPerformed(evt);
        }
});

button7.setLabel("Add");
button7.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
        button7ActionPerformed(evt);
        }
});

button8.setLabel("Move Cities");
button8.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
        button8ActionPerformed(evt);
        }
});

button9.setLabel("Clear All");
button9.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
        button9ActionPerformed(evt);
        }
});

button10.setEnabled(false);
button10.setLabel("Stop Moving");
button10.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
        button10ActionPerformed(evt);
```

```
            }
        });

        javax.swing.GroupLayout jFrame2Layout = new
javax.swing.GroupLayout(jFrame2.getContentPane());
        jFrame2.getContentPane().setLayout(jFrame2Layout);
        jFrame2Layout.setHorizontalGroup(

        jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addGroup(jFrame2Layout.createSequentialGroup()
                .addContainerGap()

        .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.TRAILING)
                        .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
jFrame2Layout.createSequentialGroup()
                        .addComponent(choice3,
javax.swing.GroupLayout.PREFERRED_SIZE, 157,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(label14,
javax.swing.GroupLayout.PREFERRED_SIZE, 121,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGroup(jFrame2Layout.createSequentialGroup()

        .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.TRAILING)

        .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
jFrame2Layout.createSequentialGroup()
                                .addComponent(button3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(button4,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGroup(jFrame2Layout.createSequentialGroup()

        .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.TRAILING, false)
                                .addComponent(button6,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(choice4,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.PREFERRED_SIZE, 157,
javax.swing.GroupLayout.PREFERRED_SIZE))

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

        .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.LEADING)
                                .addComponent(label16,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addGroup(jFrame2Layout.createSequentialGroup()
                                .addComponent(label13,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(label15,
javax.swing.GroupLayout.PREFERRED_SIZE, 45,
javax.swing.GroupLayout.PREFERRED_SIZE)))))
                        .addGap(15, 15, 15)))
```

```
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
224, Short.MAX_VALUE)

                .addComponent(button7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


        .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.LEADING, false)

                        .addComponent(button10,
javax.swing.GroupLayout.PREFERRED_SIZE, 0, Short.MAX_VALUE)

                        .addComponent(button8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


        .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.LEADING)

                        .addComponent(button9,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

                        .addComponent(jLabel3,
javax.swing.GroupLayout.PREFERRED_SIZE, 60,
javax.swing.GroupLayout.PREFERRED_SIZE))

                .addContainerGap())

        );

        jFrame2Layout.setVerticalGroup(


        jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)

                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jFrame2Layout.createSequentialGroup()
```

```
                .addContainerGap(414, Short.MAX_VALUE)

            .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.TRAILING)

                .addComponent(button7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

                .addGroup(jFrame2Layout.createSequentialGroup()

            .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.TRAILING)

                .addComponent(jLabel3,
javax.swing.GroupLayout.PREFERRED_SIZE, 60,
javax.swing.GroupLayout.PREFERRED_SIZE)

                .addComponent(button10,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

            .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.LEADING)

                .addComponent(button8,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

                .addComponent(button9,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

                .addGroup(jFrame2Layout.createSequentialGroup()

            .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.TRAILING, false)
```

```
                .addComponent(choice3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                .addComponent(label14,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


        .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.LEADING)

                .addComponent(choice4,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

                .addComponent(label16,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


        .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.LEADING, false)

                .addComponent(button6,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                .addComponent(label13,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                .addComponent(label15,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

                .addGap(15, 15, 15)


        .addGroup(jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.LEADING)

                .addComponent(button3,
javax.swing.GroupLayout.Alignment.TRAILING,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(button4,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addContainerGap())
        );

        jDialog1.setMinimumSize(new java.awt.Dimension(470, 300));
        jDialog1.setModal(true);
        jDialog1.setResizable(false);

        jPanel1.setMaximumSize(new java.awt.Dimension(450, 235));
        jPanel1.setMinimumSize(new java.awt.Dimension(450, 235));

        textField1.setText("Enter City1 Here");
        textField1.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
            textField1MouseClicked(evt);
            }
        });

        textField2.setText("Enter City2 Here");
        textField2.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
            textField2MouseClicked(evt);
            }
        });
```

```
textField3.setText("Enter Their Distance");
textField3.addMouseListener(new java.awt.event.MouseAdapter() {
        public void mouseClicked(java.awt.event.MouseEvent evt) {
        textField3MouseClicked(evt);
        }
});

button11.setLabel("Apply");
button11.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
        button11ActionPerformed(evt);
        }
});

button12.setLabel("Cancel");
button12.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
        button12ActionPerformed(evt);
        }
});

button13.setLabel("Ok");
button13.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
        button13ActionPerformed(evt);
        }
});

jLabel1.setToolTipText("");
jLabel1.setMinimumSize(new java.awt.Dimension(60, 60));
```

```
        jLabel1.setPreferredSize(new java.awt.Dimension(60, 60));


        jLabel2.setText("* Dont Give The Same City As Input");


        jLabel5.setText("* You Can Only Give A Positive Number");


        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(


        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                .addContainerGap(190, Short.MAX_VALUE)
                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 70,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(190, 190, 190))
                .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()


        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                        .addComponent(textField3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(textField1,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(textField2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addGroup(jPanel1Layout.createSequentialGroup()
```

```
                .addComponent(button13,
javax.swing.GroupLayout.PREFERRED_SIZE, 110,
javax.swing.GroupLayout.PREFERRED_SIZE)

                .addGap(50, 50, 50)

                .addComponent(button11,
javax.swing.GroupLayout.PREFERRED_SIZE, 110,
javax.swing.GroupLayout.PREFERRED_SIZE)


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                .addComponent(button12,
javax.swing.GroupLayout.PREFERRED_SIZE, 110,
javax.swing.GroupLayout.PREFERRED_SIZE))

                .addGroup(jPanel1Layout.createSequentialGroup()


        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)

                        .addComponent(jLabel5)

                        .addComponent(jLabel2))

                .addGap(0, 0, Short.MAX_VALUE)))

            .addContainerGap())

        );

        jPanel1Layout.setVerticalGroup(


        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)

            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()

            .addContainerGap()


        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.TRAILING)

                .addGroup(jPanel1Layout.createSequentialGroup()

                .addGap(0, 0, Short.MAX_VALUE)

                .addComponent(button12,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

                .addGroup(jPanel1Layout.createSequentialGroup()

                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 70,
javax.swing.GroupLayout.PREFERRED_SIZE)


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

                .addComponent(textField1,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

                .addComponent(textField3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

                .addComponent(textField2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

                .addComponent(jLabel2)


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)

                .addComponent(jLabel5)


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
106, Short.MAX_VALUE)


        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING, false)
```

```
                                .addComponent(button13,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                                .addComponent(button11,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))))

                .addContainerGap())

        );


        javax.swing.GroupLayout jDialog1Layout = new
javax.swing.GroupLayout(jDialog1.getContentPane());

        jDialog1.getContentPane().setLayout(jDialog1Layout);

        jDialog1Layout.setHorizontalGroup(


        jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)

                .addComponent(jPanel1,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

        );

        jDialog1Layout.setVerticalGroup(


        jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)

                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

        );


        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE
);

        setTitle("Simple GPS");

        setMaximumSize(new java.awt.Dimension(763, 547));

        setMinimumSize(new java.awt.Dimension(763, 547));

        setResizable(false);
```

```
button1.setEnabled(false);
button1.setLabel("Default Map");

button2.setLabel("New Map");
button2.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
        button2ActionPerformed(evt);
        }
});

button5.setEnabled(false);
button5.setLabel("Show Preferred Road");
button5.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
        button5ActionPerformed(evt);
        }
});

checkbox1.setEnabled(false);
checkbox1.setLabel("Veroia");

checkbox2.setEnabled(false);
checkbox2.setLabel("Thessalonikh");

checkbox3.setEnabled(false);
checkbox3.setLabel("Kastoria");

checkbox4.setEnabled(false);
checkbox4.setLabel("Katerini");
```

```
checkbox5.setEnabled(false);
checkbox5.setLabel("Kozani");

checkbox6.setEnabled(false);
checkbox6.setLabel("Khalkidiki");

checkbox7.setEnabled(false);
checkbox7.setLabel("Edessa");

choice1.addItemListener(new java.awt.event.ItemListener() {
        public void itemStateChanged(java.awt.event.ItemEvent evt) {
        choice1ItemStateChanged(evt);
        }
});

choice2.addItemListener(new java.awt.event.ItemListener() {
        public void itemStateChanged(java.awt.event.ItemEvent evt) {
        choice2ItemStateChanged(evt);
        }
});

label1.setText("Pick Starting Point");

label2.setText("Pick Destination Point");

label3.setText("16");

label4.setText("10");
```

```
        label5.setText("15");

        label6.setText("8");

        label7.setText("25");

        label8.setText("14");

        label9.setText("30");

        label10.setCursor(new
java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));
        label10.setText("15");

        label11.setText("Total Distance:");

        jLabel4.setBackground(new java.awt.Color(60, 60, 65));

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                .addContainerGap()

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
                        .addGroup(layout.createSequentialGroup()
                        .addComponent(button1,
javax.swing.GroupLayout.DEFAULT_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

                .addComponent(button2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

                .addGap(575, 575, 575))

                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.T
RAILING)

                        .addGroup(layout.createSequentialGroup()

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING, false)

                                .addComponent(button5,
javax.swing.GroupLayout.DEFAULT_SIZE, 157, Short.MAX_VALUE)

                                .addComponent(choice2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)

                                .addGroup(layout.createSequentialGroup()

                                .addComponent(label11,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

                                .addComponent(label12,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)

                                .addGap(311, 311, 311)
```

```
                                .addComponent(jLabel4,
javax.swing.GroupLayout.PREFERRED_SIZE, 60,
javax.swing.GroupLayout.PREFERRED_SIZE))


        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()


        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.T
RAILING)

                                .addComponent(label1,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                                .addComponent(label2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

                                .addGap(455, 455, 455))))

                        .addGroup(layout.createSequentialGroup()

                        .addGap(419, 419, 419)

                        .addComponent(checkbox4,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                        .addGap(257, 257, 257))

                        .addGroup(layout.createSequentialGroup()


        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)

                                .addGroup(layout.createSequentialGroup()

                                .addGap(109, 109, 109)

                                .addComponent(checkbox5,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                                .addGap(208, 208, 208)

                                .addComponent(label9,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
```

```
                .addComponent(choice1,
javax.swing.GroupLayout.PREFERRED_SIZE, 157,
javax.swing.GroupLayout.PREFERRED_SIZE))

                .addGap(347, 347, 347))

                .addGroup(layout.createSequentialGroup()

                .addGap(97, 97, 97)

                .addComponent(label5,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                .addGap(129, 129, 129)

                .addComponent(label8,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                .addGap(202, 202, 202)

                .addComponent(checkbox2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                .addGap(184, 184, 184))

                .addGroup(layout.createSequentialGroup()


        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.T
RAILING)

                .addGroup(layout.createSequentialGroup()

                .addGap(68, 68, 68)

                .addComponent(label4,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                .addGap(87, 87, 87)


        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.T
RAILING)

                .addComponent(checkbox7,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)


        .addGroup(layout.createSequentialGroup()
```

```
                                        .addGap(49, 49, 49)
                                        .addComponent(label3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                                    .addGap(105, 105, 105)
                                    .addComponent(label7,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                                    .addGroup(layout.createSequentialGroup()
                                    .addComponent(checkbox3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                    .addGap(230, 230, 230)
                                    .addComponent(checkbox1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                                .addGap(306, 306, 306)
                                .addComponent(checkbox6,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                            .addContainerGap())))
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addGap(428, 428, 428)
                .addComponent(label6, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGap(185, 185, 185)
                .addComponent(label10, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGap(132, 132, 132))
        );
        layout.setVerticalGroup(

        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
```

```
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
                        .addGroup(layout.createSequentialGroup()
                        .addGap(65, 65, 65)
                        .addComponent(checkbox7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(22, 22, 22)

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
                                .addComponent(checkbox2,
javax.swing.GroupLayout.DEFAULT_SIZE, 26, Short.MAX_VALUE)
                                .addComponent(label5,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
                                .addGroup(layout.createSequentialGroup()
                                .addGap(14, 14, 14)

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.T
RAILING)
                                        .addComponent(label6,
javax.swing.GroupLayout.DEFAULT_SIZE, 23, Short.MAX_VALUE)
                                        .addComponent(label10,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                                .addGap(14, 14, 14)

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.T
RAILING)
                                        .addComponent(checkbox3,
javax.swing.GroupLayout.DEFAULT_SIZE, 23, Short.MAX_VALUE)
```

```
                                .addComponent(checkbox1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))

                            .addGroup(layout.createSequentialGroup()

                            .addGap(37, 37, 37)

                            .addComponent(checkbox6,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                            .addGap(11, 11, 11)))

                    .addGap(23, 23, 23))

                    .addGroup(layout.createSequentialGroup()

                    .addGap(77, 77, 77)

                    .addComponent(label7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

                    .addGap(24, 24, 24)

                    .addComponent(label8,
javax.swing.GroupLayout.PREFERRED_SIZE, 0, Short.MAX_VALUE)

                    .addGap(77, 77, 77)))


        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)

                    .addComponent(label3,
javax.swing.GroupLayout.DEFAULT_SIZE, 26, Short.MAX_VALUE)

                    .addComponent(label4,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

                .addGap(18, 18, 18)


        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)

                    .addComponent(checkbox5,
javax.swing.GroupLayout.DEFAULT_SIZE, 26, Short.MAX_VALUE)

                    .addComponent(label9,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
```

```
                .addGap(47, 47, 47)

                .addComponent(checkbox4,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

                .addGap(29, 29, 29)


        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.T
RAILING)

                        .addComponent(choice1,
javax.swing.GroupLayout.PREFERRED_SIZE, 26,
javax.swing.GroupLayout.PREFERRED_SIZE)

                        .addComponent(label1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)

                        .addComponent(choice2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

                        .addComponent(label2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))


        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)

                        .addGroup(layout.createSequentialGroup()


        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)

                                .addComponent(button5,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                                .addGroup(layout.createSequentialGroup()
                                .addComponent(label12,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addGap(4, 4, 4))
                                .addComponent(label11,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 24,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING, false)
                                .addComponent(button2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(button1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                        .addComponent(jLabel4,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap())
        );

        getAccessibleContext().setAccessibleName("JFrameDefault");

        pack();
        }// </editor-fold>

        //Default's Map Starting Point Choice Got Picked So The Proper City Must Be
        //Selected Accordingly
        private void choice1ItemStateChanged(java.awt.event.ItemEvent evt) {

        // Starting Point Got An Input From User
```

```
TikCheckBoxes(choice1);
}

//Default's Map Destination Choice Got Picked So The Proper City Must Be
//Selected Accordingly
private void choice2ItemStateChanged(java.awt.event.ItemEvent evt) {

// Destination Point Got An Input From User
TikCheckBoxes(choice2);
}

//The Default Show Road Got Pressed So That The Road Will Be Show With
The
//Total Distance
private void button5ActionPerformed(java.awt.event.ActionEvent evt) {

// Show Prefered Road Button Pressed
showroad(evt);
}

//The New Graph Button Got Pressed Where The User Will Be Able To
Create A
//Custom Map
private void button2ActionPerformed(java.awt.event.ActionEvent evt) {

// New Graph
this.setVisible(false);
jFrame2.setVisible(true);

}
```

```
        //jFrame2's Show Road Got Pressed So That The Road Will Be Show With
The
        //Total Distance
        private void button6ActionPerformed(java.awt.event.ActionEvent evt) {

        // Show Prefered Road Button Pressed
        showroad(evt);
        }

        //jFrame2's Starting Location Choice Got Picked So The Proper City Must Be
        //Selected Accordingly
        private void choice3ItemStateChanged(java.awt.event.ItemEvent evt) {

        // Starting Point Got An Input From User
        TikCheckBoxes(choice3);
        }

        //jFrame2's Destination Choice Got Picked So The Proper City Must Be
        //Selected Accordingly
        private void choice4ItemStateChanged(java.awt.event.ItemEvent evt) {

        // Destination Point Got An Input From User
        TikCheckBoxes(choice4);
        }

        //The Default Graph Button Got Pressed And A Predesigned Map Will Be
Given
        //To The User
        private void button3ActionPerformed(java.awt.event.ActionEvent evt) {

        // Default Graph Requested
        jFrame2.setVisible(false);
```

```
this.setVisible(true);
}

//The Add Button Got Pressed So jDialog1 Opens And Insert Cities In The
Map
//Of jFrame2
private void button7ActionPerformed(java.awt.event.ActionEvent evt) {

// Add Button Pressed
jDialog1.setVisible(true);
}

//Move Button Got Pressed And Any City Can Be Moved In The jFrame2
//Designated Area
private void button8ActionPerformed(java.awt.event.ActionEvent evt) {

// Move Cities Pressed
if(!citydata2.isEmpty())
{
        button10.setEnabled(true); //stop moving

        choice3.setEnabled(false);
        choice4.setEnabled(false);
        button3.setEnabled(false);
        button6.setEnabled(false);
        button7.setEnabled(false);
        button8.setEnabled(false);
        button9.setEnabled(false);
        moveflag = true;
}
}
```

```
//Clear Button Got Pressed Where All Cities
//And Distances From jFrame2 Gets Deleted
private void button9ActionPerformed(java.awt.event.ActionEvent evt) {

// Clear All Cities
for(int i = 0; i < obj2.SizeOfNodes();i++)
{
        for(int y = 0; y < obj2.get(i).size(); y++)
        {
        this.jFrame2.remove((Checkbox)obj2.get(i).getindex(y));
        }
}
obj2.Clear();

for(int i = 0; i < Arrlabel.size(); i++)
{
        for(int y = 0; y < Arrlabel.get(i).size(); y++)
        {
        this.jFrame2.remove(Arrlabel.get(i).get(y));
        }
        Arrlabel.get(i).clear();
}
Arrlabel.clear();
citydata2.clear();
collide.clear();
states2.remove(3);
states2.add(3, new java.awt.Checkbox());
states2.remove(2);
states2.add(2, new java.awt.Checkbox());
choice3.removeAll();
```

```
        choice3.add("-Pick Destination-");
        choice3.select(0);
        choice4.removeAll();
        choice4.add("-Pick Destination-");
        choice4.select(0);
        button6.setEnabled(false);
        }

        //The Stop Moving Button Got Pressed So That
        //We Can Use The Other Capabilities Of The GPS
        private void button10ActionPerformed(java.awt.event.ActionEvent evt) {

        // Stop Moving Button
        if(movetemp)
        {
                movelabel();
        }
        moveflag = false;
        choice3.setEnabled(true);
        choice4.setEnabled(true);
        button3.setEnabled(true);
        button6.setEnabled(true);
        button7.setEnabled(true);
        button8.setEnabled(true);
        button9.setEnabled(true);
        if(choice3.getItem(0).equals("-Pick Destination-") ||
choice4.getItem(0).equals("-Pick Destination-"))
        {
                button6.setEnabled(false);
        }
        button10.setEnabled(false);
```

```
    }

    //Cursor Clicked On jFrame2 Where It Only Gets Used When Ok Button Gets
    //Pressed Or Move Cities Button And It Chooses Where The Cities Will Be
Placed
    private void jFrame2MouseClicked(java.awt.event.MouseEvent evt) {

    // Mouse Clicked

    Point point = evt.getPoint();
    point.setLocation(point.x -7, point.y - 35);
    if(addflag)
    {          // 1 = left button click
        if(evt.getButton() == 1 && !iscollided(point))
        {
        Checkbox check;

        if(!indexes.isEmpty())
        {
                check = (Checkbox) objt.getIndex((int)indexes.get(0));
                indexes.remove(0);

                citydata2.add(check);

                //check.setLocation(point);
                Rectangle rect =
                new Rectangle(point.x, point.y, check.getSize().width, 20);
                collide.add(rect);

                movecheck(check, point);
```

```
                    if(indexes.isEmpty())
                    {
                    addflag = false;
                    obj2.AddGraphBi(objt);

                    for(int add = Arrlabel.size(); add < obj2.SizeOfNodes();
add++)
                    {
                            Arrlabel.add(new ArrayList<>());
                    }

                    //Labels here
                    for(int i = 0; i < obj2.SizeOfNodes(); i++)
                    {
                            Checkbox tempch = (Checkbox) obj2.getIndex(i);

                            Point p1 = tempch.getLocation();

                            ArrayList<java.awt.Label> lList;

                            int index = objt.getfromElement(tempch);

                            if(index > -1)//objt has something to add or update
                            {
                            for(int a = 0; a < obj2.get(i).size();a++)
                            {

                                    Checkbox tempch2 = (Checkbox)
obj2.get(i).getindex(a);

                                    Point p2 = tempch2.getLocation();
```

```
                                        Point p = gethalfdist(p1, p2);

                                        int ind =
objt.get(index).getfromindexcounter(tempch2);

                                        //we found the second index so either update or
add from obj2
                                        if(ind > -1)
                                        {
                                        int compfir = obj2.getfromElement(tempch2);
                                        int compsec =
obj2.get(compfir).getfromindexcounter(tempch);

                                        java.awt.Label tlab;
                                        //We get proper labels here
                                        if(Arrlabel.get(compfir).size() > compsec)
                                        {
                                                tlab =
Arrlabel.get(compfir).get(compsec);
                                        }else
                                        {
                                                Integer text = (Integer)
objt.get(index).get(ind);
                                                tlab = new
java.awt.Label(Integer.toString(text));
                                                tlab.setBounds(p.x, p.y,
tlab.getText().length() * 8, 20);
                                        }

                                        lList = Arrlabel.get(i);
                                        //We remove here
                                        if(lList.size() > a)
                                        {
```

```
                        tlab = lList.get(a);
                        tlab.setText(tempch2.getLabel());
                        tlab.setLocation(p);
                        lList.set(a, tlab);
                }else
                {
                        int siz = lList.size();
                        if(siz >= obj2.get(i).size())
                        {
                        java.awt.Label del = lList.get(siz);
                        this.jFrame2.remove(del);
                        lList.remove(siz);
                        }
                        lList.add(tlab);
                        this.jFrame2.add(tlab);
                }
                }
        }
        }
}//Labels here

choice3.removeAll();
choice4.removeAll();

choice3.add("-Pick Destination-");
choice4.add("-Pick Destination-");

for(int i = 0; i < obj2.SizeOfNodes();i++)
{
        Checkbox s = (Checkbox) obj2.getIndex(i);
```

```
                    choice3.add(s.getLabel());
                    choice4.add(s.getLabel());
                }

                choice3.select(0);
                choice4.select(0);
                states2.remove(3);
                states2.remove(2);
                states2.add(new Checkbox());
                states2.add(new Checkbox());
                button6.setEnabled(false);

                objt.Removeall();
                }
        }
        }
}else if(moveflag)
{
        // 1 = left button click
        if(evt.getButton() == 1 && iscollided(point) && !movetemp)
        {
        colindex = getcollided(point);
        temp.add((Checkbox) obj2.getIndex(colindex));
        movetemp = true;

        }else if(evt.getButton() == 1 && !iscollided(point) && movetemp)
        {
        movecheck(temp.get(0), point);
        collide.set(colindex, temp.get(0).getBounds());
```

```
        movelabel();
        }
}
}

//Cursor Moved On jFrame2 Where It Only Gets
//Used When Ok Button Got Pressed Or Move Cities Button
private void jFrame2MouseMoved(java.awt.event.MouseEvent evt) {

// Mouse Moved On Screen
try{
        TimeUnit.MILLISECONDS.sleep(100);
}
catch(InterruptedException e){
        System.err.println(e);
}

Point point = evt.getPoint();
point.setLocation(point.x -7, point.y - 35);

if(addflag && !iscollided(point))
{
        Checkbox check = (Checkbox) objt.getIndex((int) indexes.get(0));
        movecheck(check, point);
}else if(movetemp && !iscollided(point) && !temp.isEmpty())
{
        movecheck(temp.get(0), point);
}
}

//The Ok Button Got Pressed Where It Allows
```

```
//The User To Added His Desired Cities To The jFrame2 Window
private void button13ActionPerformed(java.awt.event.ActionEvent evt) {

// Ok Button Pressed
applychecksfromtextfields();

jLabel2.setVisible(false);
jLabel5.setVisible(false);

jLabel1.setVisible(false);

textField1.setText("Enter City1 Here");
textField2.setText("Enter City2 Here");
textField3.setText("Enter Their Distance");

jDialog1.setVisible(false);

for(int i = 0; i < objt.SizeOfNodes(); i++)
{
        if(!obj2.containsindex(objt.getIndex(i)))
        {
        indexes.add(i);//we add the indexes we want to parse
        }
}

if(!indexes.isEmpty())
{
        addflag = true;
}else
{
        for(int i = 0; i < obj2.SizeOfNodes(); i++)
```

```
{
Checkbox ch1 = (Checkbox) obj2.getIndex(i);
int tind = objt.getfromElement(ch1);
if(tind > -1)
{
        for(int y = 0; y < obj2.get(i).size(); y++)
        {
        Checkbox ch2 = (Checkbox) obj2.get(i).getindex(y);
        int tind2 = objt.get(tind).getfromindexcounter(ch2);
        if(tind2 > -1)
        {
                java.awt.Label l1, l2;
                Integer temp = (Integer) objt.get(tind).get(tind2);
                l1 = Arrlabel.get(i).get(y);

                //Temporary label
                l2 = new  java.awt.Label(Integer.toString(temp));

                if(l2.getText().equals(l1.getText()))
                {
                return;
                }else
                {
                Point p1, p2, p;
                p1 = ch1.getLocation();
                p2 = ch2.getLocation();
                p = gethalfdist(p1, p2);

                l2.setBounds(p.x, p.y,
                l2.getText().length() * 8, 20);
```

```
                        Arrlabel.get(i).set(tind2, l2);
                        obj2.AddBidirectional(ch1, ch2, temp);
                        this.jFrame2.remove(l1);
                        this.jFrame2.add(l2);
                        }
                    }
                    }
                }
                }
            }
            }

        //The Cansel Button Got Pressed So Any Cities And Distances That Where
        //Saved While jDialog1 Was Open Get Deleted
        private void button12ActionPerformed(java.awt.event.ActionEvent evt) {

        // Cancel Button Pressed
        objt.Removeall();
        jDialog1.setVisible(false);
        }

        //The Apply Button Got Pressed From jDialog1 Where When Ok Button
        //Gets Pressed All The Cities With Their Distances Get Added To The
jFrame2
        private void button11ActionPerformed(java.awt.event.ActionEvent evt) {

        // Apply Button Pressed
        applychecksfromtextfields();
        }
```

```
//This Is The Second TextField Where The
//Distance Of The Two Cities Will Be Added From jDialog1
private void textField3MouseClicked(java.awt.event.MouseEvent evt) {

// Enter Their Distance
if(textField3.getText().equals("Enter Their Distance"))
{textField3.setText("");}
}

//This Is The Third TextField Where
//The Second City Will Be Added From jDialog1
private void textField2MouseClicked(java.awt.event.MouseEvent evt) {

// Enter City2 Here
if(textField2.getText().equals("Enter City2 Here"))
{textField2.setText("");}
}

//This Is The First TextField Where The
//First City Will Be Added From jDialog1
private void textField1MouseClicked(java.awt.event.MouseEvent evt) {

// Enter City1 Here
if(textField1.getText().equals("Enter City1 Here"))
{textField1.setText("");}
}



/**
 * @param args the command line arguments
 */
```

```
//The Main Fuction Where Our Program Starts
public static void main(String args[]) {
/* Set the Nimbus look and feel */
//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
/* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
 * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
 */
try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
        if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
        }
        }
} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Simple_GPS.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Simple_GPS.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Simple_GPS.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Simple_GPS.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
}
//</editor-fold>

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
        Simple_GPS mainframe = new Simple_GPS();

        mainframe.setVisible(false);
        mainframe.jFrame2.setVisible(true);

        }
});
}

//Glovbal Variables That Are Going To Be Used In Our Program
private ArrayList<Checkbox> temp = new ArrayList<>();
private ArrayList<Integer> indexes = new ArrayList<>();
//private int indexforevt = 0;
private int colindex;
private ArrayList<Rectangle> collide = new ArrayList<>();
//private ArrayList<Rectangle> collabel = new ArrayList<>();
private ArrayList<ArrayList<java.awt.Label>> Arrlabel = new ArrayList<>();
private ArrayList<java.awt.Checkbox> states2   = new ArrayList<>();
private ArrayList<java.awt.Checkbox> citydata  = new ArrayList<>();
private ArrayList<java.awt.Checkbox> citydata2 = new ArrayList<>();
private Boolean addflag = false, moveflag = false, movetemp = false;
private Graph obj  = new Graph();
private Graph obj2 = new Graph();
private Graph objt = new Graph();
```

```
// Variables declaration - do not modify
private java.awt.Button button1;
private java.awt.Button button10;
private java.awt.Button button11;
private java.awt.Button button12;
private java.awt.Button button13;
private java.awt.Button button2;
private java.awt.Button button3;
private java.awt.Button button4;
private java.awt.Button button5;
private java.awt.Button button6;
private java.awt.Button button7;
private java.awt.Button button8;
private java.awt.Button button9;
private java.awt.Checkbox checkbox1;
private java.awt.Checkbox checkbox2;
private java.awt.Checkbox checkbox3;
private java.awt.Checkbox checkbox4;
private java.awt.Checkbox checkbox5;
private java.awt.Checkbox checkbox6;
private java.awt.Checkbox checkbox7;
private java.awt.Choice choice1;
private java.awt.Choice choice2;
private java.awt.Choice choice3;
private java.awt.Choice choice4;
private javax.swing.JDialog jDialog1;
private javax.swing.JFrame jFrame2;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
```

```
        private javax.swing.JLabel jLabel4;
        private javax.swing.JLabel jLabel5;
        private javax.swing.JPanel jPanel1;
        private java.awt.Label label1;
        private java.awt.Label label10;
        private java.awt.Label label11;
        private java.awt.Label label12;
        private java.awt.Label label13;
        private java.awt.Label label14;
        private java.awt.Label label15;
        private java.awt.Label label16;
        private java.awt.Label label2;
        private java.awt.Label label3;
        private java.awt.Label label4;
        private java.awt.Label label5;
        private java.awt.Label label6;
        private java.awt.Label label7;
        private java.awt.Label label8;
        private java.awt.Label label9;
        private java.awt.TextField textField1;
        private java.awt.TextField textField2;
        private java.awt.TextField textField3;
        // End of variables declaration
}
```

Graph Class

```
package com.mycompany.gps_project;

import java.util.ArrayList;
import java.util.PriorityQueue;
/**
```

```
 * @author George Kousidis
 */

//Graph Class Uses DyArray And Pair Class For Finding Shortest Possible Route And Also Showing The Data
public class Graph<E> {
        private  //elements         ,index
        DyArray<DyArray<Integer, E>, E> graph = new DyArray<>(); //From Pair<E> to Integer

        private ArrayList<Integer> data = new ArrayList<>();//Showing the distance
        private ArrayList<ArrayList<E>> datadest = new ArrayList<>();//Showing the road

        public
        //The Default Constructor It Initializes The Private Data
        Graph() //default
        {
                graph = new DyArray<>();
                data = new ArrayList<>();
                datadest = new ArrayList<>();
        }

        //This Function Adds A New Graph To The Graph That Calls This Function
        void AddGraphBi(Graph newgraph)
        {
                for(int i = 0; i < newgraph.graph.size(); i++)
                {
                DyArray<Integer, E> tempitem = (DyArray<Integer, E>) newgraph.graph.get(i);
                E Start = (E) newgraph.graph.getindex(i);
                for(int y = 0; y < tempitem.size(); y++)
```

```
        {
                Integer tempP = tempitem.get(y);
                E End = tempitem.getindex(y);
                this.AddBidirectional(Start, End, tempP);
        }
        }
}

//This Function Adds A Road To Two Cities But At Only One Direction
void AddOnedirectional(final E Start, final E End, Integer dist)
{
        if(dist.compareTo(0) <= 0 || Start.equals(End))
        {return;}

        if(!graph.containsIndex(Start))
        {

        DyArray<Integer, E> Arrobj = new DyArray<>();
        Arrobj.addElementAt(End, dist);
        graph.addElementAt(Start, Arrobj);
        }else{
        if(graph.getfromindex(Start).containsIndex(End))
        {
                graph.getfromindex(Start).setElementAt(End, dist);
        }else
        {
                DyArray<Integer, E> Arrobj = graph.getfromindex(Start);
                Arrobj.addElementAt(End, dist);
                graph.addElementAt(Start, Arrobj);
        }
```

```
        }
}

//This Function Adds A Road To Two Cities Back And Forth
void AddBidirectional(final E Start, final E End, Integer dist)
{
        AddOnedirectional(Start, End, dist);
        AddOnedirectional(End, Start, dist);
}

//This Function Returns The Number Of Cities
int SizeOfNodes()
{
        return graph.size();
}

//This Function Removes The Cities Of A Graph Without Deleting The Cities
void Removeall()
{
        for(int i = 0; i < graph.size(); i++)
        {
        for(int y = 0; y < graph.get(i).size(); y++)
        {
                graph.get(i).remove(y);
        }
        graph.get(i).Clear();
        //graph.remove(i);
        }
        graph.Clear();
}
```

```
//This Function Deletes All The Cities Of The Graph That Calls This Function
void Clear()
{
        for(int i = 0; i < graph.size(); i++)
        {
        graph.get(i).Clear();
        }
        graph.Clear();
}

//This Function Finds The Shortest Route Possible At All Possible Destinations
void DijkstraAlgo(final E src)
{
        if(graph.isEmpty())
        {
        System.out.println("Initialize a graph");
        return;
        }

        data = new ArrayList<>(); //Showing the minimum distance
        datadest = new ArrayList<>();//Showing the road


        for(int itr = 0; itr < graph.size(); itr++)
        {
        ArrayList<E> init = new ArrayList<>();
        datadest.add(itr, init);
        if(itr == graph.getfromindexcounter(src))
        {
```

```
                data.add(itr ,0); //Src = 0
            }else{
                data.add(itr, Integer.MAX_VALUE);
            }
            }

            PriorityQueue<Pair<Integer>> mypq = new PriorityQueue<>(new
Comparator_Pair());
            mypq.add(new Pair(graph.getfromindexcounter(src), 0));
            Integer u, v, weight, i;
            E vconvert;

            while(!mypq.isEmpty())
            {
            u = mypq.poll().getLeft();

            for(i = 0; i < graph.get(u).size(); i++)
            {
                vconvert = graph.get(u).getindex(i);
                v        = graph.getfromindexcounter(vconvert);
                weight   = graph.get(u).get(i);

                if(data.get(v) > data.get(u) + weight)
                {
                Object newpath = datadest.get(u).clone();
                ArrayList<E> newpathobj = (ArrayList<E>) newpath; //deep
copy
                newpathobj.add(vconvert);
                datadest.set(v, newpathobj); // The Destination path
```

```
                    data.set(v, data.get(u) + weight); //The Destination total
distance
                    mypq.add(new Pair(v, data.get(v)));
                    }

            }
            }
        }

        //This Function Returns The Route To The Destination And The Total
Distance
        //It Will Take To Reach It
        Pair<ArrayList<E>> GetData(E Dest)
        {
            int i = graph.getfromindexcounter(Dest);
            ArrayList<E> Route = new ArrayList<>();
            Pair<ArrayList<E>> RouteData;
            if(!datadest.get(i).isEmpty())
            {
            Route = getMinRoute(i);
            i = getMinDistance(i);
            }else
            {
            i = 0;
            Route.add(Dest);
            //RouteData = new Pair<>(Route, i);
            }
            RouteData = new Pair<>(Route, i);
            return RouteData;
        }
```

```
	//This Function Returns The Shortest Route's Total Distance
	ArrayList<E> getMinRoute(int i)
	{
		return datadest.get(i);
	}

	//This Function Returns The Shortest Routes
	int getMinDistance(int i)
	{
		return data.get(i);
	}

	//This Function Returns Every Road's Distance From Beginning To Destination
	ArrayList<Integer> GetRawDistData(int index)
	{
		ArrayList<Integer> Route = new ArrayList<>();
		for(int i = 0; i < graph.get(index).size(); i++)
		{
		Route.add(graph.get(index).get(i));
		}
		return Route;
	}

	//This Function Returns The Integer Location Of The DyArray With Input The City
	int getfromElement(E element)
	{
		return graph.getfromindexcounter(element);
	}
```

```
        //This Function Returns The City Location Of The DyArray With Input The
Integer
        E getIndex(int index)
        {
                return graph.getindex(index);
        }

        //This Function Returns The All The Connected Cities
        //Between The index City And Their Distance
        DyArray<Integer, E> get(int index)
        {
                return graph.get(index);
        }

        //This Function Checks If The index Exist In The Current Object
        Boolean containsindex(int index)
        {
                return graph.containsIndex(index);
        }

        //This Function Checks If The index Exist In The Current Object
        Boolean containsindex(E index)
        {
                return graph.containsIndex(index);
        }
}
```

DyArray Class

```
package com.mycompany.gps_project;
/**
 *
 * @author Kousidis George
```

```
 */
import java.util.ArrayList;

public class DyArray<E, T>{
	private ArrayList<E> array;
	private ArrayList<T> indexcount;
	private int indexcounter = 0;

	//The Default Constructor It Initializes The Private Data
	public DyArray()
	{
	indexcounter = 0;
	this.array	= new ArrayList<>();
	this.indexcount = new ArrayList<>();
	}

	//Constructor With Given Desired Data
	public DyArray(T index, E element)
	{
	indexcounter = 0;
	indexcount.add(indexcounter, index);
	array.add(indexcounter, element);
	indexcounter++;
	}

	//The Method That Adds To Our Data Structure
	public void addElementAt(T index, E a)
	{
	int checkindex = this.getfromindexcounter(index);
```

```
if(checkindex == -1)
{
//arbitrary index location
indexcount.add(indexcounter, index);
//inserts an element at the specified counter
array.add(indexcounter, a);
indexcounter++;
return;
}
array.set(checkindex, a); //.add(checkindex, a);
}

//The Method That Sets To Our Data Structure
public void setElementAt(T index, E element)
{
for(int i = 0; i < indexcounter; i++)
{
        if(indexcount.get(i) == index)
        {
        array.set(i, element);
        }
}
}

//The Method That Gives The Element Based On Its Arbitrary Index
public E getfromindex(T index)
{
for(int i = 0; i < indexcounter; i++)
{
        if(indexcount.get(i) == index)
```

```
            {
            return array.get(i);
            }
        }
        System.out.println("Index was not found program exception incoming");
        E e = (E)null;
        return e;
        }

        //The Method That Gives The Integer Of The Array Based On Its Arbitrary
Index
        public int getfromindexcounter(T index)
        {
        for(int i = 0; i < indexcounter; i++)
        {
            if(indexcount.get(i) == index)
            {
            return i;
            }
        }
        //System.out.println("Index was not found returning -1");
        return -1;
        }

        //The Method That Gives The Arbitrary Index Based On The Integer index
        public T getindex(int index)
        {
        if(index <= indexcounter)
        {
            return indexcount.get(index);
        }
```

```
System.out.println("Index was not found returning null arbitrary index");
return (T)null;
}

//The Method That Gives The Element Based On Integer index
public E get(int index)
{
if(index <= indexcounter)
{
        return array.get(index);
}
System.out.println("Index was not found returning null element");
return (E)null;
}

//The Method That Returns The Total Size Of The DyArray
public int size()
{
return indexcounter;
}

//The Method That Removes The Element And Arbitrary Index At The Integer
index
public void remove(int i)
{
array.remove(i);
indexcount.remove(i);
indexcounter--;
}

//The Method That Clears The Whole Instance Of The Class
```

```
public void Clear()
{
array.clear();
indexcount.clear();
indexcounter = 0;
}

//The Method That Checks If It Contains The Particular Element
public Boolean containsElement(E element)
{
return array.contains(element);
}

//The Method That Checks If It Contains The Particular Arbitrary Index
public Boolean containsIndex(T index)
{
//System.out.println("Element was not found returning false");
return indexcount.contains(index);
}

//The Method That Checks If It Contains The Particular Integer index
public Boolean containsIndex(int index)
{
if(index <= indexcounter)
{
        return true;
}else
{
        return false;
}
```

```
        }

        //The Method That Checks An Instance Has Data In It
        public Boolean isEmpty()
        {
        return indexcounter == 0;
        }
}
```

Pair Class

```
package com.mycompany.gps_project;

/**
 *
 * @author George Kousidis
 */
//The Pair Class Where We Have
//An Arbitrary Element On The Left And An Integer On The Right
public class Pair<E>{

  private E left;
  private Integer right;

  //Constructor With Desired Data
  public Pair(E left, Integer right) {
            assert left != null;
            assert right != null;

            this.left = left;
            this.right = right;
```

```
  }

  //The Method That Modifies The Element Of The Class's Instance
  public void setLeft(E left)
  {
            this.left = left;
  }

  //The Method That Modifies The Integer Of The Class's Instance
  public void setRight(Integer right)
  {
            this.right = right;
  }

  //The Method That Returns The Element Of The Class's Instance
  public E getLeft()  { return left; }
  //The Method That Returns The Integer Of The Class's Instance
  public Integer getRight() { return right; }
}
```

Comparator_Pair Class

```
package com.mycompany.gps_project;

import java.util.Comparator;

/**
 *
 * @author George Kousidis
 */
//The Class That Defines How The Pair Class Will Compare Its Data
public class Comparator_Pair<T extends Comparable<T>> implements
Comparator<Pair<T>>{
```

```
        //The Overrideable Method That Does The Comparison Of The Data
        @Override
        public int compare(Pair<T> o1, Pair<T> o2) {
        if (o1.getRight() < o2.getRight()) {
            return 1;
        }
        else
        {
            return -1;
        }
        }
}
```