ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Δημιουργία εφαρμογής επεξεργασίας εικόνας με Java

Ιωάννης Γκοσδής

Επιβλέπων: Γλαβάς Ευριπίδης,
βαθμίδα: καθηγητής

Άρτα, Μάρτιος, 2023

Java image processing application

Εγκρίθηκε από τριμελή εξεταστική επιτροπή

Τόπος, Ημερομηνία

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Επιβλέπων καθηγητής

   Ευριπίδης Γλαβάς,

2. Μέλος επιτροπής

   Όνομα Επίθετο,

3. Μέλος επιτροπής

   Όνομα Επίθετο,

Δήλωση μη λογοκλοπής

Δηλώνω υπεύθυνα και γνωρίζοντας τις κυρώσεις του Ν. 2121/1993 περί Πνευματικής Ιδιοκτησίας, ότι η παρούσα πτυχιακή εργασία είναι εξ ολοκλήρου αποτέλεσμα δικής μου ερευνητικής εργασίας, δεν αποτελεί προϊόν αντιγραφής ούτε προέρχεται από ανάθεση σε τρίτους. Όλες οι πηγές που χρησιμοποιήθηκαν (κάθε είδους, μορφής και προέλευσης) για τη συγγραφή της περιλαμβάνονται στη βιβλιογραφία.

Γκοσδής Ιωάννης

Υπογραφή

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία με θέμα «Δημιουργία εφαρμογής επεξεργασίας εικόνας με java» πραγματοποιήθηκε στο πλαίσιο της πτυχιακής εργασίας του τμήματος πληροφορικής και τηλεπικοινωνιών του πανεπιστημίου Ιωαννίνων με έδρα την Άρτα το έτος 2023. Ευχαριστώ θερμά τον επιβλέπων καθηγητή μου, κύριο Ευριπίδη Γλαβά για την καθοδήγηση του στο θέμα της πτυχιακής εργασίας μου, για την βοήθεια που μου πρόσφερε με τις υποδείξεις του , και την επιμονή του. Θερμές ευχαριστίες αποδίδω σε όλους τους καθηγητές που είχα στην ακαδημαϊκή μου ζωή, καθώς και για τις γνώσεις που μου πρόσφεραν και με εξέλιξαν ως άνθρωπο. Τέλος θα ήθελα να ευχαριστήσω την οικογένεια μου που με στήριξαν όλα αυτά τα χρόνια καθώς και για την υπομονή που έδειξαν.

ΠΕΡΙΛΗΨΗ

Στην παρούσα πτυχιακή εργασία θα κατασκευάσουμε μια εφαρμογή επεξεργασίας εικόνων με χρήση της γλώσσας προγραμματισμού Java. Αρχικά θα γίνει αναφορά στις γενικές έννοιες για την ψηφιακή φωτογραφία και πως αυτή αναγνωρίζεται και αναπαρίσταται στον υπολογιστή. Στη συνέχεια θα γίνει αναφορά στη Java και τις δομές που χρησιμοποιεί για την αναπαράσταση εικόνας. Έπειτα θα συνδυάσουμε τις έννοιες αυτές για να δημιουργήσουμε μια εφαρμογή επεξεργασίας ψηφιακών εικόνων.

ABSTRACT

The purpose of this thesis is to develop a stand alone application by using Java programming language for digital image editing and apply transformations to a digital image. Firstly we are going to present basic principals of digital image processing and how the computer stores and displays an RGB image. After that we are going to present the Java basics and the data structures that Java uses for digital images. Finally we are going to combine this knowledge and make an application for image processing.

***Keywords*:**: array, pixel, interface, frame, RGB, jar

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

Πίνακας εικόνων

1. Ψηφιακή εικόνα

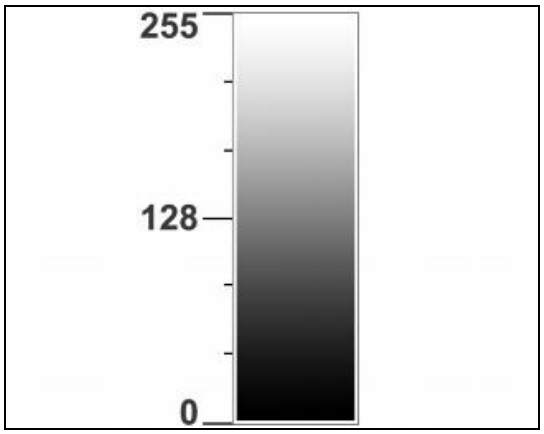
1.1 Χρωματικά μοντέλα

Τα χρωματικά μοντέλα είναι ένας τρόπος που συμφωνείται για να γίνεται η αναπαράσταση των χρωμάτων στον υπολογιστή. Κάθε χρωματικό μοντέλο δημιουργεί συνθέσεις από χρωματικές συνιστώσες. Υπάρχει μια μεγάλη γκάμα χρωματικών μοντέλων ανάλογα με τον τύπο του γραφικού που θα χρησιμοποιηθεί.

Για να γίνουν πιο κατανοητά τα παραπάνω θα παραθέσουμε τέσσερα από τα πιο γνωστά και ευρέως διαδεδομένα χρωματικά μοντέλα τα οποία αποτέλεσαν τη βάση για να δημιουργηθούν κι άλλα μοντέλα. Τα χρωματικά μοντέλα είναι : το Grayscale, το RGB, το HSB και το CMYK. Στο σημείο αυτό θα πρέπει να αναφέρουμε ότι τα χρωματικά μοντέλα είναι πολύ περισσότερα και πολλές εταιρείες που δραστηριοποιούνται στο χώρο των γραφικών και της φωτογραφίας έχουν δημιουργήσει δικά τους χρωματικά μοντέλα που είναι συμβατά με τις δικές τους εφαρμογές και συσκευές.

1.1.1 Grayscale model

Όπως λέει και το όνομα του πρόκειται για ένα χρωματικό μοντέλο που αποτελείται από χρωματικές συνιστώσες του γκρι. Εδώ έχουμε μόνο μία χρωματική συνιστώσα που ονομάζεται φωτεινότητα. Οι τιμές της φωτεινότητας μπορεί να έχουν την τιμή 0 που είναι το μαύρο και να φτάσουν ως την τιμή 255 που είναι το λευκό. Προφανώς οι ενδιάμεσες τιμές είναι οι αποχρώσεις του γκρι. Εμπειρικά μία ασπρόμαυρη φωτογραφία που βλέπουμε είναι μια grayscale image. Στο σημείο αυτό θα πρέπει να διευκρινίσουμε ότι οι ασπρόμαυρες φωτογραφίες που γνωρίζουμε εμπειρικά δεν θα πρέπει να συγχέονται με τις δυαδικές εικόνες (binary images) που έχουν μόνο το άσπρο ή το μαύρο (Understanding color models, 2013).

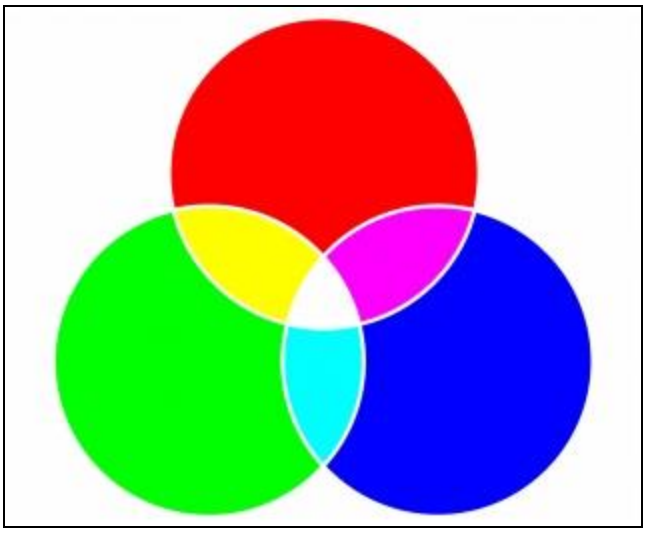


Εικόνα 1-Grayscale image

1.1.2 RGB model

Το χρωματικό μοντέλο RGB χρησιμοποιεί τις τρεις χρωματικές συνιστώσες κόκκινο (RED), πράσινο (GREEN), μπλε (BLUE) για να ορίσει ένα χρώμα. Σε μία κλασσική εικόνα βάθους 24bits κάθε χρωματική συνιστώσα μπορεί να πάρει την τιμή από 0..255 ενώ σε εικόνες με μεγαλύτερο βάθος μπορούμε να πάρουμε περισσότερες τιμές για μια χρωματική συνιστώσα επιτυγχάνοντας περισσότερες αποχρώσεις. Η κάθε μία χρωματική συνιστώσα αναπαρίσταται από 8bits όπου με 8bits μπορούμε να έχουμε 256 τιμές. Ακριβώς επειδή χρησιμοποιούμε τρεις συνιστώσες λέμε ότι δημιουργούμε μια εικόνα με 24bits βάθος.

Το χρωματικό μοντέλο RGB είναι ένα προσθετικό μοντέλο. Το χρώμα δημιουργείται από το φως που μεταδίδεται. Για το λόγο αυτό είναι ιδανικό για χρήση σε οθόνες όπου τα τρία βασικά χρώματα αναμειγνύονται για να παραχθούν οι διάφορες αποχρώσεις. Όταν τα τρία αυτά χρώματα δοθούν στη μέγιστη τιμή τους (255,255,255) το μάτι αντιλαμβάνεται το συνδυασμό αυτό ως λευκό χρώμα. Αντίθετα όταν οι χρωματικές συνιστώσες γίνουν (0,0,0) που υπονοεί απουσία φωτός τότε το ανθρώπινο μάτι λαμβάνει το μαύρο χρώμα. (Doughty, 2009).

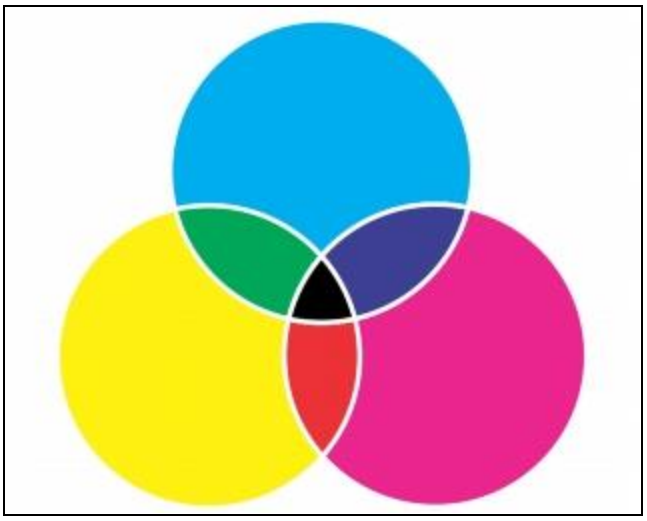


Εικόνα 2- Rgb model

1.1.3 CMYK color model

Το χρωματικό μοντέλο αυτό είναι ένα αφαιρετικό μοντέλο. Οι χρωματικές αποχρώσεις εδώ δημιουργούνται αφαιρώντας τιμές από τις χρωματικές συνιστώσες. Το μοντέλο αυτό δημιουργείται με χρήση των τεσσάρων χρωμάτων : κυανό (Cyan), μαντζέντα (Magenta) που είναι ένα κόκκινο-μοβ χρώμα, κίτρινο (Yellow) και μαύρο (Black). Η ποσότητα κάθε χρωματικής συνιστώσας συμμετέχει με ένα ποσοστό στο τελικό χρώμα. Όταν όλα τα χρώματα συνυπάρχουν τότε δημιουργείται το μαύρο χρώμα. Αν αφαιρέσουμε ποσοστά από όλα τα χρώματα τότε δημιουργείται το λευκό χρώμα.

Το χρωματικό μοντέλο αυτό υιοθετήθηκε κυρίως για χρήση από τους εκτυπωτές μιας και το αυτούσιο μαύρο χρώμα είναι πιο καθαρό από το να δημιουργηθεί από τη χρήση των άλλων τριών συνιστωσών και επίσης είναι πιο φθηνό από τα έγχρωμα μελάνια (Doughty, 2009).

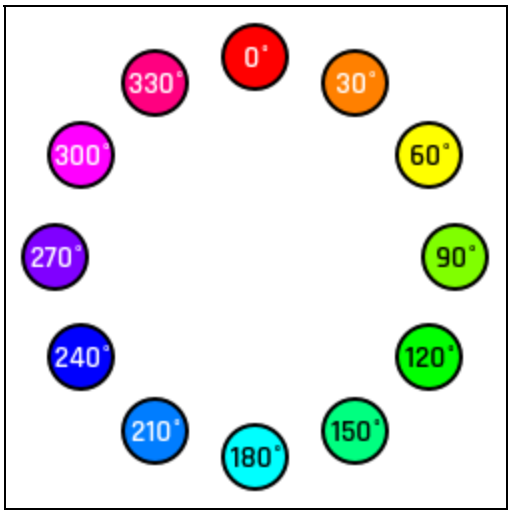


Εικόνα 3-CMYK model

1.1.4 HSB color model

Το HSB χρωματικό μοντέλο χρησιμοποιεί τρεις συνιστώσες για να δημιουργήσει τις χρωματικές αποχρώσεις. Αυτές είναι η απόχρωση (Hue), κορεσμός (Saturation) και φωτεινότητα (Brightness). Η τελευταία συνιστώσα αναφέρεται και ως τιμή (Value) και για αυτό το λόγο συναντάμε το ίδιο μοντέλο με το όνομα HSV. Το χρωματικό αυτό μοντέλο έχει εφαρμογή στο πεδίο των γραφικών όπως είναι για παράδειγμα το Photoshop όπου το μοντέλο αυτό χρησιμοποιείται για να επιλέξουμε ένα χρώμα με το εργαλείο sample.

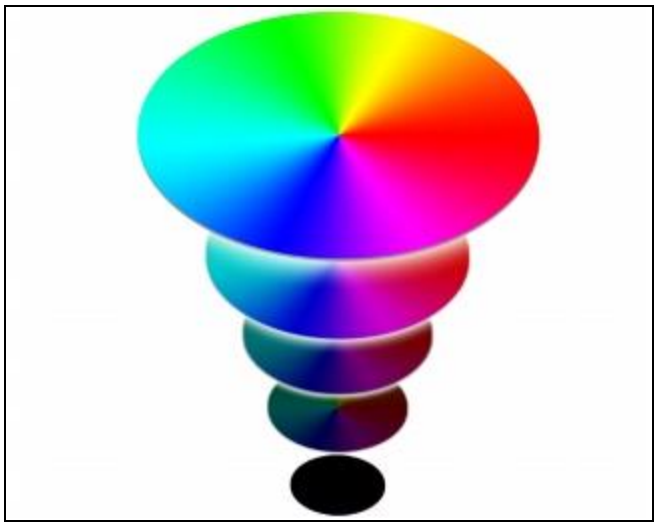
Η απόχρωση (Hue) αναπαριστά την τιμή του χρώματος στον χρωματικό κύκλο. Για παράδειγμα στις 0^o (μοίρες) έχουμε την τιμή για το κόκκινο, στις 60^o έχουμε την τιμή για το κίτρινο, στις 120^o έχουμε την τιμή για το πράσινο , στις 180^o έχουμε την τιμή για το κυανό, στις 240^o έχουμε την τιμή για το μπλε και στις 300^o είναι το μαντζέντα (Kennedy, 2020).



Εικόνα 4-Χρωματικός κύκλος

Ο κορεσμός (Saturation) αναφέρεται με ποσοστό (%) , παίρνει τιμές από 0 έως 100 και σημαίνει τη «ζωντάνια» με την οποία συμμετέχει η συνιστώσα. Προφανώς 100% σημαίνει ότι το χρώμα συμμετέχει με πλήρη κορεσμό.

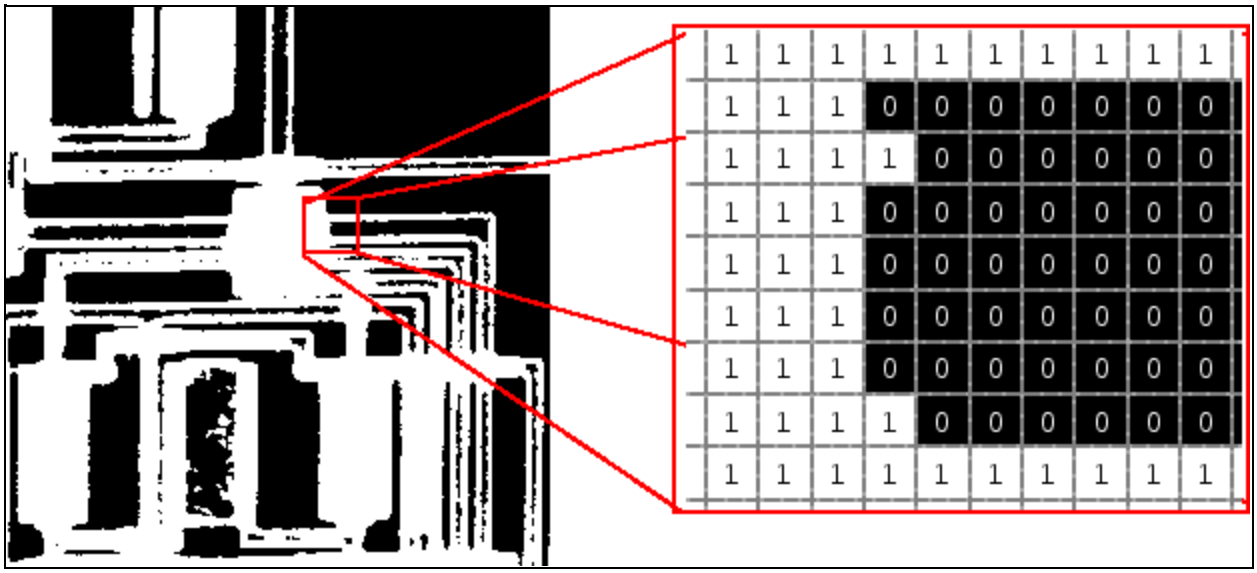
Η ίδια ποσοστιαία αναπαράσταση ακολουθείται και για την φωτεινότητα όπου όσο μεγαλύτερο είναι το ποσοστό τόσο μεγαλύτερη φωτεινότητα έχουμε. (Understanding color models, 2013).



Εικόνα 5-HSB model

1.2 Δυαδικές εικόνες

Οι δυαδικές εικόνες (binary images) χρησιμοποιούν 1 bit για την αναπαράσταση ενός εικονοστοιχείου. Επειδή λοιπόν έχουμε 1 bit απευθείας έχουμε μόνο δύο δυνατές τιμές: το 1 και το 0. Έτσι λοιπόν οι δυαδικές εικόνες έχουν μόνο άσπρο ή μαύρο και δεν πρέπει να τις συγχέουμε με αυτό που λέμε στην καθομιλουμένη για ασπρόμαυρες εικόνες έναντι των έγχρωμων , γιατί εκεί εννοούμε τις grayscale images που θα αναφέρουμε ευθύς αμέσως. Οι δυαδικές εικόνες δημιουργούνται από εικόνες που λαμβάνουμε γύρω μας χρησιμοποιώντας ένα κατώφλι (threshold) όπως λέγεται. Οποιαδήποτε τιμή πάνω από την τιμή της κατωφλίωσης θα πάρει την τιμή 1 , δηλαδή λευκό και οποιαδήποτε τιμή κάτω από την κατωφλίωση θα πάρει την τιμή 0 , δηλαδή μαύρο (Types of Images, 2021).



Εικόνα 6-(Image Types in the ToolBox, 2022)

1.3Εικόνες grayscale

Οι εικόνες grayscale είναι οι εικόνες αποχρώσεων του γκρι. Κάθε εικονοστοιχείο χρησιμοποιεί 8bits για την αναπαράστασή του, πράγμα που σημαίνει ότι οι δυνατές τιμές που μπορεί να πάρει είναι μέσα στο διάστημα 0..255. Θεωρούμε ως μαύρο την τιμή 0, ως λευκό την τιμή 255 και οι ενδιάμεσες τιμές είναι οι αποχρώσεις του γκρι. Αυτές δηλαδή είναι οι χρωματικές συνιστώσες μια τέτοιας εικόνας. Το επόμενο παράδειγμα είναι κατατοπιστικό όπου έχουμε μια έγχρωμη εικόνα και δίπλα την αντίστοιχη σε gray scale

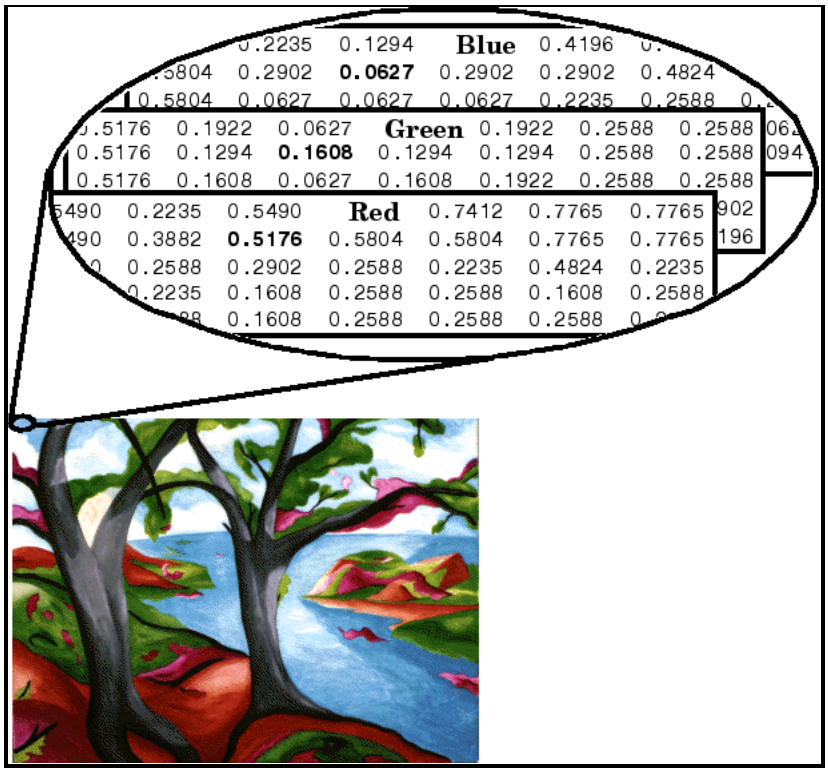
Εικόνα 7-(Convert a color photo to Grayscale mode, 2022)

1.4 Εικόνες RGB

Οι εικόνες RGB, λέγονται και true color είναι αυτές με τις οποίες θα ασχοληθούμε περισσότερο στην εργασία μας μιας και είναι αυτές που χρησιμοποιούμε καθημερινά. Οι εικόνες αυτές χρησιμοποιούν το RGB χρωματικό μοντέλο και έχουν βάθος χρώματος 24bits. Αυτό προκύπτει από το γεγονός ότι κάθε χρωματική συνιστώσα χρειάζεται 8bits και επειδή για την αναπαράσταση κάθε εικονοστοιχείου θέλουμε 3 χρωματικές συνιστώσες που θα συνδυαστούν έχουμε 3*8bits=24bits. Κάθε χρωματική συνιστώσα μπορεί να πάρει τιμές ανάμεσα στο 0..255. Για να γίνει αυτή η αναπαράσταση στον υπολογιστή μας θα πρέπει να χρησιμοποιηθεί η κατάλληλη δομή που δεν είναι άλλη από έναν τρισδιάστατο πίνακα όπου έχουμε width X height X layers όπου τα τρία layers είναι τα red , green και blue.

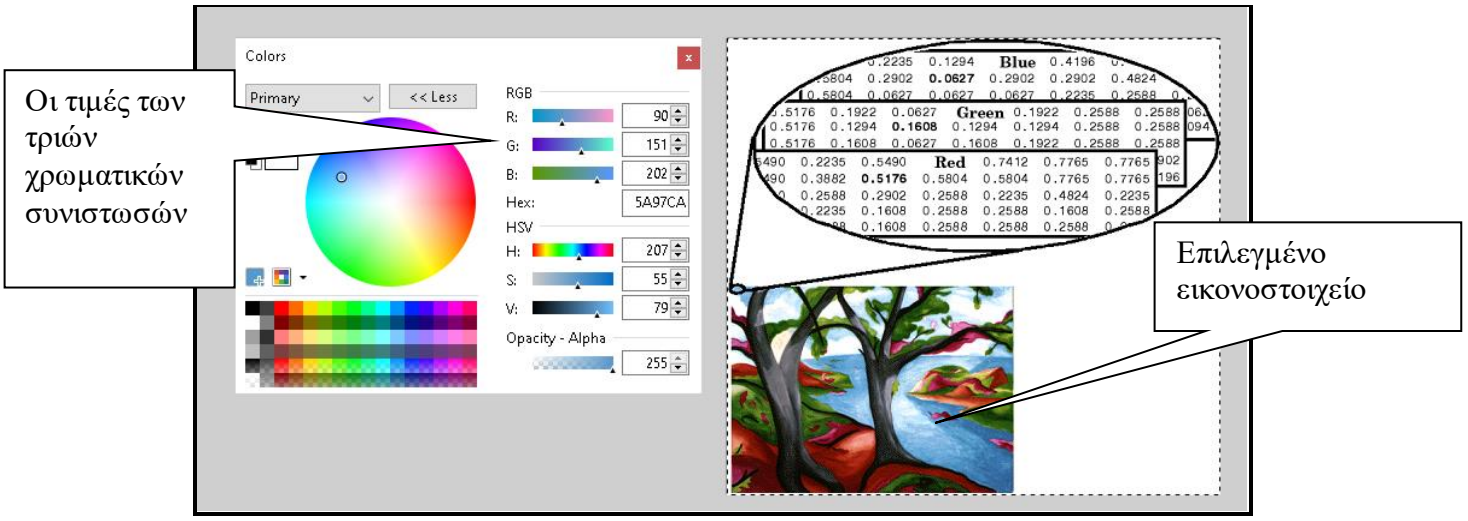
Για να γίνει πιο κατανοητή η υπόσταση μιας τέτοια εικόνας δείχνουμε την αναπαράστασή της στην επόμενη εικόνα με μια επισήμανση ότι στην επόμενη εικόνα

κάθε συνιστώσα μπορεί να πάρει περισσότερες από 256 τιμές, αλλά η φιλοσοφία του τρισδιάστατου πίνακα είναι ακριβώς η ίδια.



Εικόνα 8-(Image Types, 2022)

Επιπροσθέτως αντιγράψαμε την εικόνα και την επικολλήσαμε σε ένα καινούριο αρχείο της δωρεάν εφαρμογής paint.NET που είναι πρόγραμμα επεξεργασίας φωτογραφίας. Επιλέξαμε το εργαλείο color picker και εμφανίσαμε τις 3 χρωματικές συνιστώσες και τις τιμές αυτών στο RGB μοντέλο για να γίνει πιο κατανοητή η υπόσταση μιας true color εικόνας .



Εικόνα 9-Άνοιγμα εικόνας με paint.NET

Παρατηρούμε ότι επειδή έχουμε επιλέξει ένα εικονοστοιχείο μέσα στο νερό της εικόνας η τιμή για το μπλε είναι 202 και υπερισχύει των άλλων δύο χρωματικών συνιστωσών.

2. Η γλώσσα προγραμματισμού Java

2.1 Ιστορία της Java

Η ιστορία δημιουργίας της Java είναι κάτι που έχει αρκετό ενδιαφέρον. Η γλώσσα αυτή αρχικά σχεδιάστηκε για τη λειτουργία τηλεοράσεων που θα είχαν κάποια διαδραστικότητα καθώς και για άλλες συσκευές όπου θα χρησιμοποιούνταν για τη λειτουργία ενσωματωμένων συσκευών σε αυτές όπως για παράδειγμα το πλυντήριο ρούχων. Η ιστορία της ξεκινά από μία ομάδα που λεγόταν Green Team που ξεκίνησε ένα έργο δημιουργίας μιας γλώσσας προγραμματισμού για ψηφιακές συσκευές. Στην πορεία είδαν ότι αυτή η γλώσσα ταίριαζε καλύτερα για διαδικτυακό προγραμματισμό και για το λόγο αυτό η τεχνολογία αυτή υιοθετήθηκε από την Netscape.

Η νέα γλώσσα έπρεπε να έχει τα εξής χαρακτηριστικά:

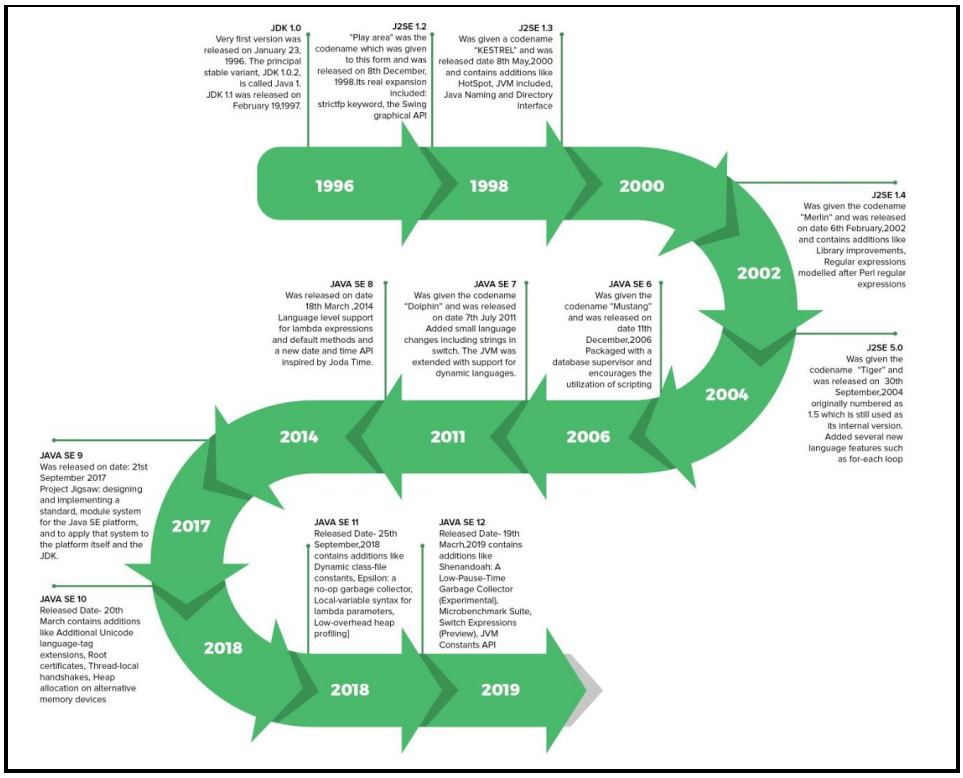
- Απλή
- Στιβαρή
- Μεταφέρσιμη
- Ανεξάρτητη από πλατφόρμα
- Ασφαλής
- Υψηλής απόδοσης
- Πολυνηματική
- Ανεξάρτητη αρχιτεκτονικής
- Αντικειμενοστρεφής
- Δυναμική

Η Java δημιουργήθηκε από τον James Gosling που θεωρείται ο πατέρας της γλώσσας. Το έργο δημιουργίας της γλώσσας από αυτόν και την ομάδα του ξεκίνησε στις αρχές του 1990 και πιο συγκεκριμένα μαζί με τον Mike Sheridan και τον Patrick Naughton τον Ιούνιο του 1991 θεωρείται ότι έγινε η απαρχή της γλώσσας.

Η γλώσσα αρχικά ονομάστηκε “Greentalk” από τον James Gosling εξαιτίας της επέκτασης των αρχείων που ήταν “.gt”. Έπειτα ονομάστηκε “Oak” (βελανιδιά) μιας και το δέντρο αυτό είναι σύμβολο δύναμης και ανθεκτικότητας και θεωρείται και εθνικό δέντρο από πολλά κράτη (History of Java, 2021).

Το τελικό όνομα της γλώσσας ήταν Java και αποφασίστηκε μετά από αρκετές συζητήσεις και προέρχεται από μια ποικιλία καρπού καφέ . Το περιοδικό TIME MAGAZINE την κατέταξε το 1995 στα δέκα καλύτερα προϊόντα της χρονιάς.

Το 1995 ήταν που παρουσιάστηκε το JDK 1.0 που είναι ουσιαστικά μια βιβλιοθήκη από κλάσεις αν μπορούμε να το πούμε όσο πιο απλά γίνεται. Σήμερα η γλώσσα προγραμματισμού χρησιμοποιείται για διαδικτυακό προγραμματισμό , συσκευές κινητών και για πακέτα επιχειρηματικών εφαρμογών. Στην παρακάτω εικόνα βλέπουμε την πορεία της γλώσσας (Bhatnagar, 2022).



Εικόνα 10- Εκδόσεις Java (Batnagar, 2022)

2.2 Κλάσεις

Η Java είναι μια αντικειμενοστρεφής γλώσσα και ο,τιδήποτε γράφουμε στη γλώσσα αυτή συσχετίζεται με κλάσεις και αντικείμενα. Η κλάση (Class) είναι το στοιχειώδες δομικό στοιχείο και μπορεί να οριστεί και ως ένα γενικό πρότυπο για την κατασκευή στιγμιοτύπων (instances) τα οποία ονομάζονται αντικείμενα (Class Definition in Java, 2021). Κάθε κλάση χαρακτηρίζεται από ιδιότητες (attributes) και μεθόδους

(methods) Για παράδειγμα στην πραγματική ζωή βλέπουμε πολλά μεμονωμένα αντικείμενα που ανήκουν στην ίδια κατηγορία. Υπάρχουν για παράδειγμα πολλά ποδήλατα που είναι της ίδιας εταιρείας και είναι ακριβώς το ίδιο μοντέλο. Με όρους αντικειμενοστραφούς προγραμματισμού κάθε ποδήλατο είναι ένα στιγμιότυπο (instance) της κλάσης των ποδηλάτων. Αυτό το στιγμιότυπο λέμε ότι είναι ένα αντικείμενο της κλάσης.

Για να γίνουν λίγο πιο κατανοητά τα παραπάνω παραθέτουμε τον ορισμό της κλάσης Bicycle (What is a Class, 2022).

```
class Bicycle {

    int cadence = 0;
    int speed = 0;
    int gear = 1;

    void changeCadence(int newValue) {
        cadence = newValue;
    }

    void changeGear(int newValue) {
        gear = newValue;
    }

    void speedUp(int increment) {
        speed = speed + increment;
    }

    void applyBrakes(int decrement) {
        speed = speed - decrement;
    }

    void printStates() {
```

```
        System.out.println("cadence:" +
            cadence + " speed:" +
            speed + " gear:" + gear);
    }
}
```

Εδώ πρέπει να προσθέσουμε ότι δεν είναι μια πλήρης εφαρμογή καθότι λείπει η κύρια συνάρτηση (main) που θα δημιουργήσει αντικείμενα της κλάσης Bicycle.

2.3 Κληρονομικότητα

Η έννοια της κληρονομικότητας είναι ένα από τα σημαντικότερα στοιχεία της γλώσσας Java. Με την έννοια αυτή μπορούμε να έχουμε κλάσεις που είναι επέκταση από άλλες κλάσεις , είναι απορρέουσες (derived) ,δηλαδή αυτές οι κλάσεις κληρονομούν πεδία και μεθόδους από άλλη κλάση. Η κλάση γονέας που κληροδοτεί ονομάζεται superclass (Java Inheritance, 2022). Μια κλάση που κληρονομεί από κάποια άλλη ονομάζεται απορρέουσα (derived), επεκτάσιμη κλάση (extended) ή κλάση παιδί (child class) ή υποκλάση (subclass)

Το στοιχειώδες αυτό εργαλείο που μας παρέχει η java μας δίνει την δυνατότητα να επεκτείνουμε τις δυνατότητες από κάποια κλάση κάνοντάς την να κληρονομεί από κάποια άλλη κλάση γνωρίσματα με μια απλή δήλωση. Με τον τρόπο αυτό δεν επαναλαμβάνουμε ίδιο κώδικα , αλλά επαναπροσδιορίζουμε νέα χαρακτηριστικά και ιδιότητες στην απορρέουσα κλάση. Στο παράδειγμα που ακολουθεί θα ορίσουμε μια κλάση με όνομα Bicycle που υλοποιεί ένα ποδήλατο και στη συνέχεια θα ορίσουμε μια κλάση που λέγεται MountainBike και θα είναι η κλάση παιδί (child). Η υποκλάση αυτή ουσιαστικά κληρονομεί και επεκτείνει (extends) όλα τα χαρακτηριστικά της κλάσης γονέα (Bicycle) και επιπλέον προσθέτει και κάποια άλλα δικά της μαζί με τις απαραίτητες μεθόδους –συναρτήσεις που χρειάζεται. Παρατηρούμε τη λέξη extends στη δήλωση της κλάσης (Inheritance, 2022).

```
public class Bicycle {

    // the Bicycle class has three fields
    public int cadence;
    public int gear;
    public int speed;
```

```
    // the Bicycle class has one constructor
    public Bicycle(int startCadence, int startSpeed, int startGear) {
        gear = startGear;
        cadence = startCadence;
        speed = startSpeed;
    }

    // the Bicycle class has four methods
    public void setCadence(int newValue) {
        cadence = newValue;
    }

    public void setGear(int newValue) {
        gear = newValue;
    }

    public void applyBrake(int decrement) {
        speed -= decrement;
    }

    public void speedUp(int increment) {
        speed += increment;
    }

}
```

Ακολουθεί η δήλωση της υποκλάσης MountainBike

```
public class MountainBike extends Bicycle {

    // the MountainBike subclass adds one field
    public int seatHeight;

    // the MountainBike subclass has one constructor
    public MountainBike(int startHeight,
                        int startCadence,
                        int startSpeed,
                        int startGear) {
        super(startCadence, startSpeed, startGear);
        seatHeight = startHeight;
    }

    // the MountainBike subclass adds one method
    public void setHeight(int newValue) {
        seatHeight = newValue;
    }
}
```

Στο σημείο αυτό αξίζει να αναφέρουμε ότι η απορρέουσα κλάση μπορεί να ξαναδηλώσει μέσα στο σώμα της μια συνάρτηση με το ίδιο όνομα που έχει κάποια άλλη που κληρονόμησε από τον γονέα της, αλλά να περιέχει διαφορετικό σώμα εντολών. Μπορεί όπως λέμε να κάνει υπέρβαση (override) και να την εξειδικεύσει στις ανάγκες της και να την χρησιμοποιήσει σε ένα άλλο προηγμένο χαρακτηριστικό που είναι ο

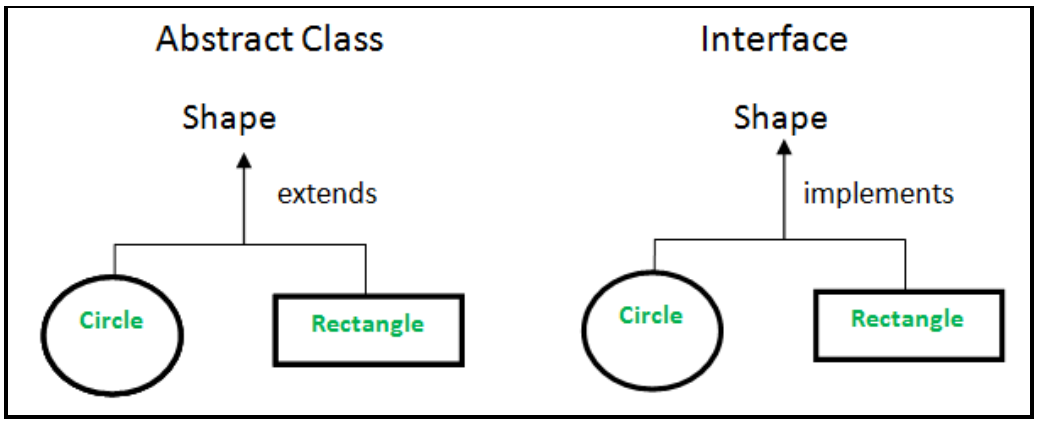
πολυμορφισμός κατά το χρόνο εκτέλεσης (runtime polymorphism) (Method Overriding in Java, 2021).

2.4 Διεπαφές (interfaces)

Ένα άλλο σημαντικό στοιχείο στη γλώσσα προγραμματισμού της java είναι η έννοια της διεπαφής (interface). Θα προσπαθήσουμε να αποφύγουμε τυπικούς ορισμούς που δεν βοηθάνε στην εύκολη κατανόηση του αναγνώστη μιας και κύριος σκοπός μας είναι να γίνονται κατανοητές οι έννοιες και όχι να παραθέτουμε αυστηρές έννοιες. Η διεπαφή είναι κατά μία έννοια σαν μια ειδική κλάση που μοιάζει με την αφηρημένη κλάση (abstract) της java. Η java υποστηρίζει την κληρονομικότητα , αλλά δεν υποστηρίζει την πολλαπλή κληρονομικότητα όπως για παράδειγμα η C++. Ωστόσο δεν δημιουργεί πρόβλημα στις ικανότητες της γλώσσας , γιατί κατά μία άλλη άποψη το κενό αυτό καλύπτεται από τα interfaces.

Η διεπαφή (interface) είναι ουσιαστικά σαν μία αφηρημένη κλάση όπου περιέχει μόνο αφηρημένες μεθόδους τις οποίες οφείλει η κλάση που υλοποιεί την διεπαφή να την ορίσει μέσα στο σώμα της. Σαν να λέμε ότι την «κληρονομεί»,αν και είναι λάθος σαν έννοια μιας και δεν την κληρονομεί. Μια κλάση μπορεί να κληρονομεί (extends) μόνο από μία κλάση , αλλά μπορεί να υλοποιεί πολλά interfaces. Με τον τρόπο αυτό μπορούμε να επιτύχουμε έμμεσα πολλαπλή κληρονομικότητα. (Σφέτσος, χ.η)

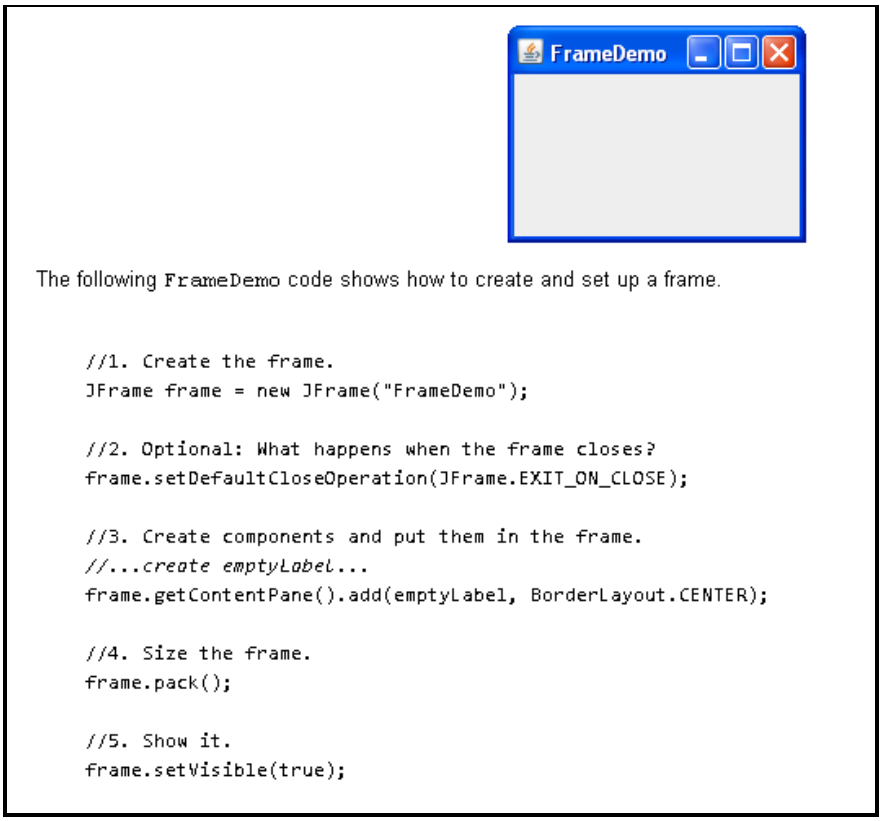
Στην επόμενη εικόνα δείχνουμε μία κλάση που κληρονομεί μία αφηρημένη κλάση και δίπλα μια κλάση που υλοποιεί ένα interface.



Εικόνα 11-(Dheerendra, 2022)

2.5 Frames

Τα Frames χρησιμοποιούνται από τη java για να μας εισάγουν στο παραθυρικό περιβάλλον. Το Frame είναι το ανώτερο ιεραρχικά παράθυρο όπου έχει τουλάχιστον έναν τίτλο και ένα περίγραμμα. Το Frame υλοποιείται ως ένα στιγμιότυπο της κλάσης JFrame. Αν δούμε τον κώδικα που μας παρέχει το επίσημο site της Oracle και πιο συγκεκριμένα τον κώδικα FrameDemo.java θα εντοπίσουμε εύκολα το σημείο όπου καλείται η συνάρτηση δημιουργίας , όπως λέγεται , που δίνει υπόσταση στο παράθυρό μας. Παραθέτουμε τις πιο σημαντικές γραμμές του κώδικα και το αποτέλεσμα που εμφανίζεται.



Εικόνα 12-(How to Make Frames, 2022)

Στην παρούσα εργασία θα εκκινήσουμε τη δημιουργία ενός τέτοιου παραθύρου όπου θα προσθέσουμε και άλλες λειτουργικότητες όπως είναι τα στοιχεία μενού καθώς και αλληλεπίδραση με το χρήστη, την οποία θα επιτύχουμε με τους ακροατές γεγονότων (action listeners) όπως λέγονται.

2.6 Java και διαχείριση ψηφιακών εικόνων

Η java ως γλώσσα προγραμματισμού υψηλού επιπέδου μπορεί να διαχειριστεί ψηφιακή επεξεργασία εικόνων με χρήση πολλών διαφορετικών συναρτήσεων. Η κύρια λειτουργικότητα μας παρέχεται μέσω της κλάσης BufferedImage Class. Η κλάση αυτή είναι επέκταση της βασικής κλάσης Image (Java BufferedImage Class, 2022). Η κλάση BufferedImage υποστηρίζει τρεις τύπους συναρτήσεων δημιουργίας (constructor) και ένα πλήθος άλλων μεθόδων όπως φαίνεται στη συνέχεια. Στις συναρτήσεις δημιουργίας μπορούμε να ορίσουμε διαστάσεις, τον τύπο της εικόνας ή αν θέλουμε να έχουμε δυαδική ή εικόνα δεικτών. Παραθέτουμε στις επόμενες δύο εικόνες τις συναρτήσεις δημιουργίας και τις υπόλοιπες συναρτήσεις (methods) της κλάσης.

Constructors

Constructor and Description
BufferedImage(**ColorModel** cm, **WritableRaster** raster, boolean isRasterPremultiplied, **Hashtable**<?,?> properties) Constructs a new BufferedImage with a specified ColorModel and Raster.
BufferedImage(int width, int height, int imageType) Constructs a BufferedImage of one of the predefined image types.
BufferedImage(int width, int height, int imageType, **IndexColorModel** cm) Constructs a BufferedImage of one of the predefined image types: TYPE_BYTE_BINARY or TYPE_BYTE_INDEXED.

Εικόνα 13-BufferedImage Constructors (Class BufferedImage, 2020)

1	**copyData(WritableRaster outRaster)** It computes an arbitrary rectangular region of the BufferedImage and copies it into a specified WritableRaster.
2	**getColorModel()** It returns object of class ColorModel of an image.
3	**getData()** It returns the image as one large tile.
4	**getData(Rectangle rect)** It computes and returns an arbitrary region of the BufferedImage.
5	**getGraphics()** This method returns a Graphics2D, retains backwards compatibility.
6	**getHeight()** It returns the height of the BufferedImage.
7	**getMinX()** It returns the minimum x coordinate of this BufferedImage.
8	**getMinY()** It returns the minimum y coordinate of this BufferedImage.
9	**getRGB(int x, int y)** It returns an integer pixel in the default RGB color model (TYPE_INT_ARGB) and default sRGB colorspace.
10	**getType()** It returns the image type.

Εικόνα 14-BufferedImage methods (Java BufferedImage Class, 2022)

Στην εργασία μας αυτή θα είναι η κύρια συνάρτηση που μας δίνει τα πιο χρήσιμα εργαλεία. Ωστόσο όπως θα δείξουμε και στον κώδικα της εφαρμογής μας θα χρειαστούμε και άλλες συναρτήσεις για το άνοιγμα και το κλείσιμο αρχείων στο δίσκο (java.io.File), τη διαχείριση σφαλμάτων εισόδου-εξόδου αρχείων (java.io.IOException)

καθώς και θα υλοποιήσουμε το interface ActionListener για να ανταποκρίνεται η εφαρμογής μας σε επιλογές του χρήστη με χρήση της συσκευής κατάδειξης (mouse).

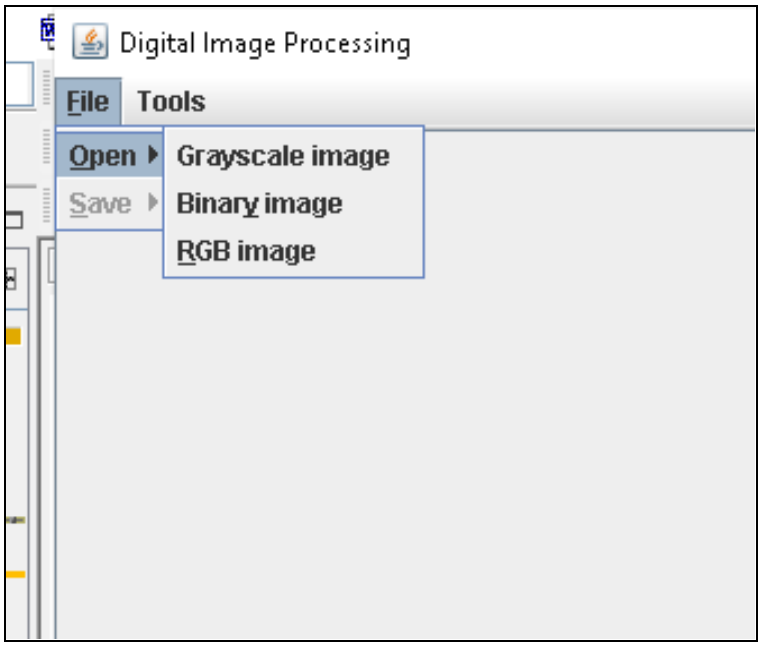
3. Κατασκευή εφαρμογής επεξεργασίας εικόνας

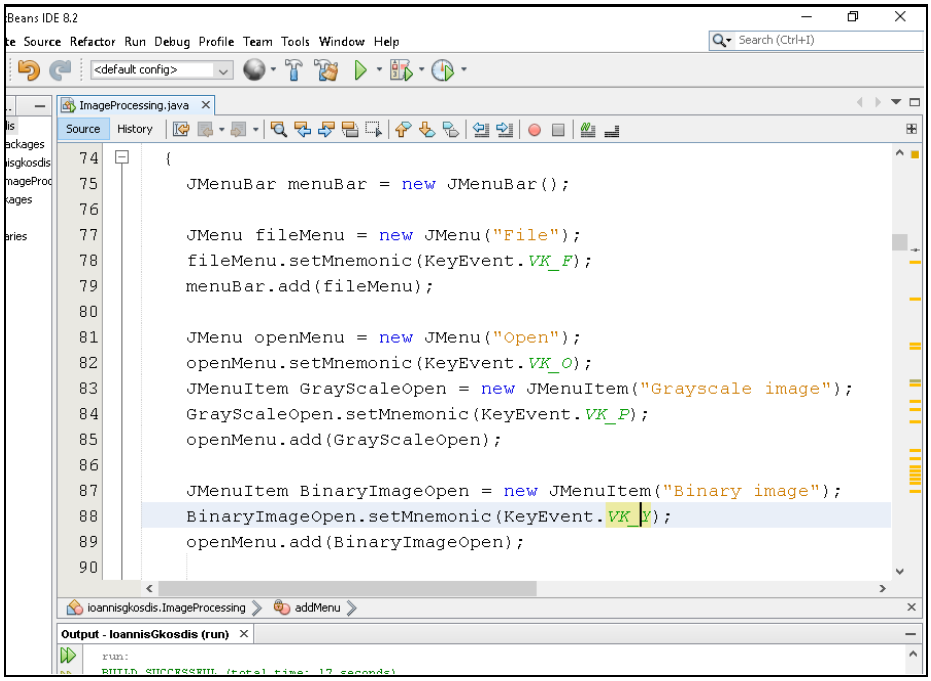
3.1 Δημιουργία μενού εντολών

Αφού δείξαμε πως φτιάχνουμε ένα στοιχειώδες παράθυρο στη Java θα αρχίσουμε να προσθέτουμε όλα εκείνα τα απαραίτητα στοιχεία για την εφαρμογή μας. Η εφαρμογή μας θα πρέπει να οργανωθεί με μία γραμμή μενού εντολών όπου κάθε μία εντολή μενού θα εμφανίζει ένα υπομενού με άλλες εντολές. Η κάθε εντολή υπομενού θα συνδέεται με ένα ακροατή γεγονότων και θα καλεί ουσιαστικά στο παρασκήνιο κάποια άλλη συνάρτηση.

Η λογική είναι απλή και θα ακολουθηθεί με την ίδια λογική για να προσθέσουμε όλη τη λειτουργικότητα στην εφαρμογή μας. Δημιουργούμε αρχικά ένα νέο αντικείμενο της κλάσης JMenuBar το οποίο θα τοποθετήσουμε μέσα στο JFrame που φτιάξαμε. Στη συνέχεια θα δημιουργήσουμε την πρώτη εντολή από το μενού που θα την ονομάσουμε “File”. Θα δώσουμε επίσης τη δυνατότητα να ανοίγει αυτή η εντολή όχι μόνο μετά από επιλογή με τη συσκευή κατάδειξης (mouse), αλλά και με χρήση συνδιασμού των πλήκτρων Alt + χαρακτήρας της επιλογής μας. Αυτό γίνεται με την συνάρτηση setMnemonic(keyevent) Με παρόμοιο τρόπο θα κατασκευάσουμε και άλλα JMenu αντικείμενα τα οποία θα τα προσθέσουμε στην κύρια εντολή μενού. Αφού τελειώσουμε με τις εντολές μενού θα πάμε να δημιουργήσουμε τις εντολές υπομενού τις οποίες θα κατασκευάσουμε με τη βοήθεια της κλάσης JMenuItem . Θα κατασκευάσουμε τέτοια αντικείμενα τα οποία θα τα προσθέσουμε ουσιαστικά σε κάθε αντικείμενο JMenu. Τα αντικείμενα JMenuItem είναι πολύ σημαντικά καθότι αυτά θα περιέχουν και έναν ακροατή γεγονότων για να ανταποκρίνονται στην επιλογή του χρήστη.

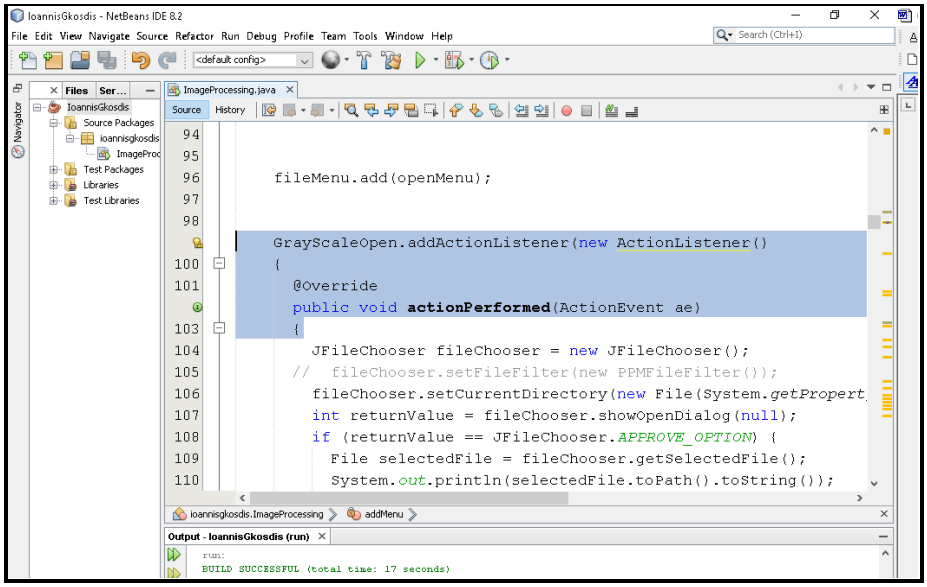
Για να γίνουν λίγο πιο κατανοητά τα παραπάνω θα εμφανίσουμε μία στοιχειώδη γραμμή εντολών με κάποιες εντολές καθώς και τον κώδικα από τον οποίο προήλθε. Στην τελική εφαρμογή μας ίσως δεν χρησιμοποιήσουμε ακριβώς αυτές τις εντολές ωστόσο για την ώρα θα δείξουμε τον γενικότερο τρόπο.





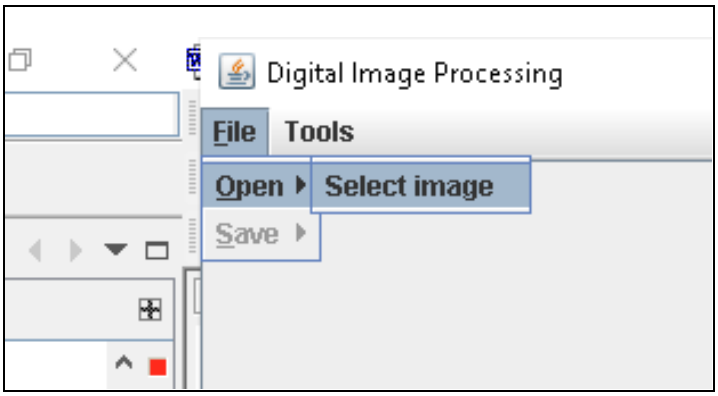
Στην επόμενη εικόνα κρατήσαμε ένα στιγμιότυπο από τη συνάρτηση που θα καλεί το αντικείμενο JMenuItem με όνομα GrayScaleOpen μετά από επιλογή της συγκεκριμένης εντολής. Το ίδιο μοτίβο θα χρησιμοποιηθεί σε όλες τις εντολές της εφαρμογής μας. Οπότε το μόνο που μένει είναι να προσθέσουμε τον κατάλληλο κώδικα

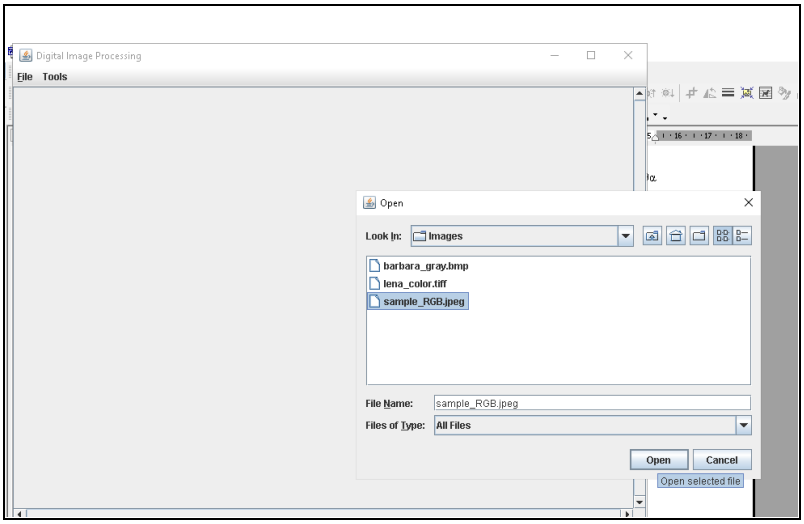
σε κάθε κατάλληλη συνάρτηση που θα δημιουργήσουμε και να τη συνδέσουμε με τη σωστή εντολή μενού.

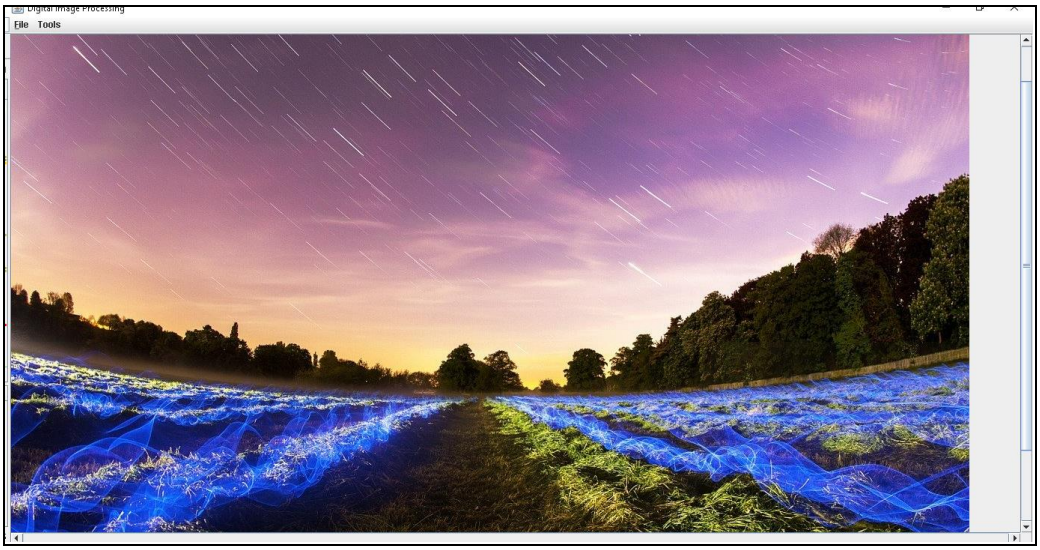


3.2 Άνοιγμα εικόνας

Θα πάμε στο φάκελο του έργου μας που δημιουργεί το NetBeans όπου θα δημιουργήσουμε έναν φάκελο που θα τον ονομάσουμε Images και μέσα σε αυτόν θα αποθηκεύσουμε κάποιες εικόνες ενδεικτικές για να επιδείξουμε τις λειτουργίες της εφαρμογής μας. Για τη λήψη κάποιων εικόνων χρησιμοποιήσαμε τον ιστότοπο filesamples.com (Sample JPEG Files Download, 2022).


Digital Image Processing
File
Tools
Open
Select image
Save


Digital Image Processing
File Tools
Open
Look In:
Images
barbara_gray.bmp
lena_color.tiff
sample_RGB.jpeg
File Name:
sample_RGB.jpeg
Files of Type:
All Files
Open
Cancel
Open selected file

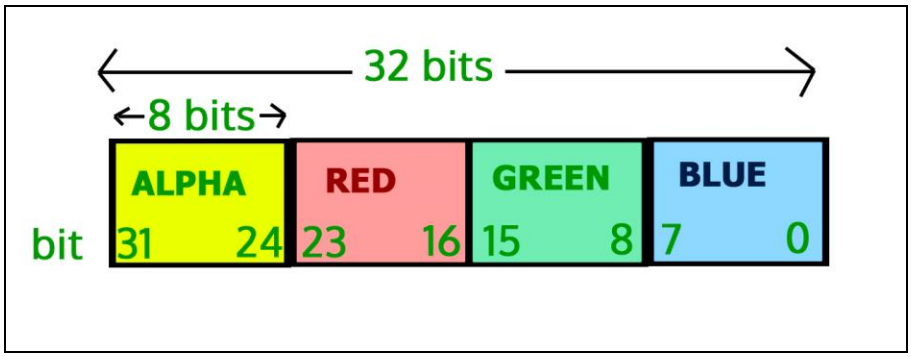

File
Tools

Παρατηρούμε ότι επειδή η εικόνα μας είναι μεγαλύτερη σε διαστάσεις από τις διαστάσεις που έχουμε για το παράθυρό μας εμφανίζεται δεξιά η μπάρα κύλισης για να μετακινηθούμε σε όλη την εικόνα. Αυτό η Java το καταφέρνει με χρήση του εργαλείου JScrollPane (Class JScrollPane, 2020).

3.3 Μετατροπή έγχρωμης εικόνας σε grayscale

Για να μετατρέψουμε μία εικόνα true color RGB σε εικόνα διαβαθμίσεων του γκρι (gray scale) θα διασχίσουμε όλη την εικόνα , ανά εικονοστοιχείο και θα βρίσκουμε το μέσο όρο από τις τρεις χρωματικές συνιστώσες. Ο μέσος όρος θα είναι μία τιμή από [0..255]. Βρίσκουμε αυτή την τιμή και αντικαθιστούμε τις τρεις συνιστώσες του εικονοστοιχείου με μία που θα είναι ο μέσος όρος τους.

Η συνάρτηση getRGB(x,y) μας δίνει έναν ακέραιο αριθμό από 32bits. Τα 8 MSB (most significant bits) είναι αυτά που αφορούν την συνιστώσα A που αφορά την διαφάνεια (transparency) της εικόνας. Στην πραγματικότητα αυτό αναφέρεται ως ARGB μοντέλο μιας και υπάρχει και η Alpha συνιστώσα. Η επόμενη εικόνα είναι κατατοπιστική:



Εικόνα 15-(Agarwal, 2021)

Εμείς θα διασχίσουμε την εικόνα μας όπως ξέρουμε από τη χρήση των πινάκων , δηλαδή με μία διπλή επανάληψη και θα στοχεύουμε σε κάθε εικονοστοιχείου που μας υποδεικνύουν οι συντεταγμένες x και y. Μόλις αντλήσουμε τον ακέραιο αυτό από 32 bits θα προσπαθήσουμε με πράξεις bitwise να απομονώσουμε τη σωστή πλειάδα από 8 bits κάθε φορά για κάθε συνιστώσα και στο τέλος θα βρούμε την μέση τιμή από τις 3 χρωματικές συνιστώσες. Η συνιστώσα Alpha δεν θα πειραχτεί και θα χρησιμοποιηθεί πάλι πίσω για την κατασκευή της gray scale image.

Οι δύο πράξεις που θα χρειαστούμε είναι η ολίσθηση δεξιά (Logical Shift Right) και ολίσθηση αριστερά (Logical Shift Left). Να υπενθυμίσουμε ότι γενικά οι πράξεις αυτές ταυτίζονται με τη διαίρεση και τον πολλαπλασιασμό αντίστοιχα πάντα στη βάση του συστήματος στο οποίο δουλεύουμε. Αρχικά θα κάνουμε LSR με τον αριθμό OxFF του δεκαεξαδικού που είναι ο αριθμός 255_{10} . Με τον τρόπο αυτό όταν θα κάνουμε λογικό «ΚΑΙ» (Bitwise and - &) αυτό θα εκτελείται μόνο μεταξύ των σωστών αντίστοιχων «οχτάδων» από bits. Όταν κρατήσουμε τις συνιστώσες και βρούμε το μέσο όρο από τις χρωματικές τότε ξανασυναρμολογούμε την ακέραια αναπαράσταση των 32 bits του εικονοστοιχείου με διαδοχικές ολισθήσεις και την πράξη τον λογικού Ή (bitwise OR-|)(Bitwise and Bit Shift Operators, 2022).

```
            int p = image.getRGB(x, y);
            int a = (p >> 24) & 0xff;// 24 bits Logical Shift
            int r = (p >> 16) & 0xff;
            int g = (p >> 8) & 0xff;
            int b = p & 0xff;

           // Ypologismos mesou orou
            int avg = (r + g + b) / 3;

            /* epistrofi olon ton timon
             ton 4 synistoson gia
             th dhmioyrgia tou neou pixel
           me gray scale times*/
            p = (a << 24) | (avg << 16) | (avg << 8)| avg;
            image.setRGB(x, y, p);
            imgLabel.setIcon(new ImageIcon(image));
        }
```

3.4 Εμφάνιση του αρνητικού της εικόνας

Στη συνέχεια θα προσπαθήσουμε να εμφανίσουμε το αρνητικό μίας εικόνας. Θα φορτώσουμε μία έγχρωμη εικόνα αρχικά και στη συνέχεια θα κάνουμε αντιστροφή (invertion) σε όλες τις τιμές των εικονοστοιχείων της . Με την αντιστροφή των τιμών σε κάθε εικονοστοιχείο αυτό που βλέπουμε λευκό θα εμφανιστεί ως μαύρο. Αυτό που εμφανίζεται ως κόκκινο (255,0,0) θα εμφανιστεί ως κυανό (0,255,255) και το ίδιο θα εφαρμόσουμε σε όλα τα εικονοστοιχεία.

Για να πάρουμε το συμπλήρωμα από κάθε μία τιμή μπορούμε να χρησιμοποιήσουμε τη λογική πράξη NOT ή πολύ απλά να αφαιρέσουμε κάθε τιμή από κάθε χρωματική συνιστώσα από το 255.

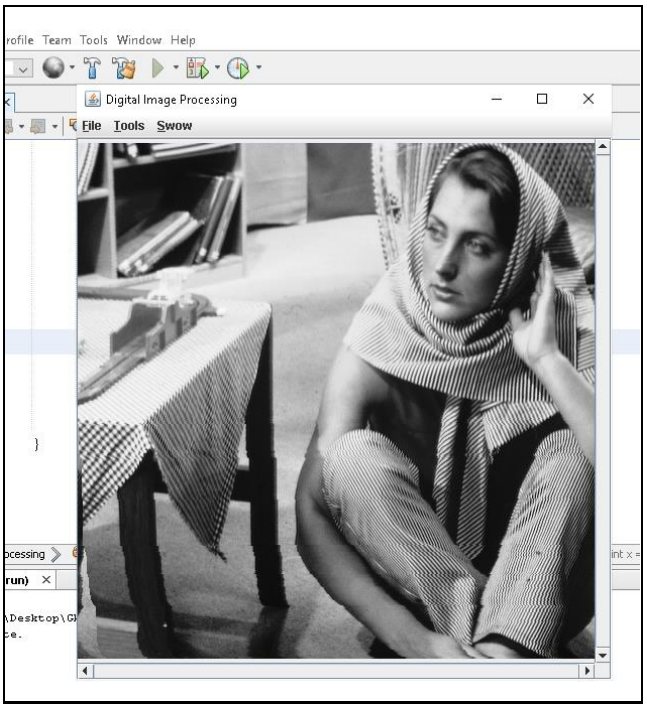
Για παράδειγμα αν έχουμε τον αριθμό 163_{10} και τον μετατρέψουμε στο δυαδικό σύστημα θα πάρουμε τον αριθμό 10100011_2 . Για να βρούμε το συμπλήρωμα αυτού του αριθμού θα κάνουμε αντιστροφή (NOT) σε κάθε ένα bit ή θα κάνουμε αφαίρεση από τον αριθμό 255_{10} . Εδώ αν κάνουμε αντιστροφή με την πράξη NOT θα πάρουμε τον δυαδικό αριθμό 01011100 που είναι ο 92_{10} . Ακόμα και αν κάναμε αφαίρεση από τον αριθμό 255 θα παίρναμε πάλι την ίδια τιμή. Στον κώδικα μας θα κάνουμε αφαίρεση από τον αριθμό 255 που είναι η πιο απλή λύση.

Άρα ο αλγόριθμος που θα ακολουθήσουμε είναι ο εξής:

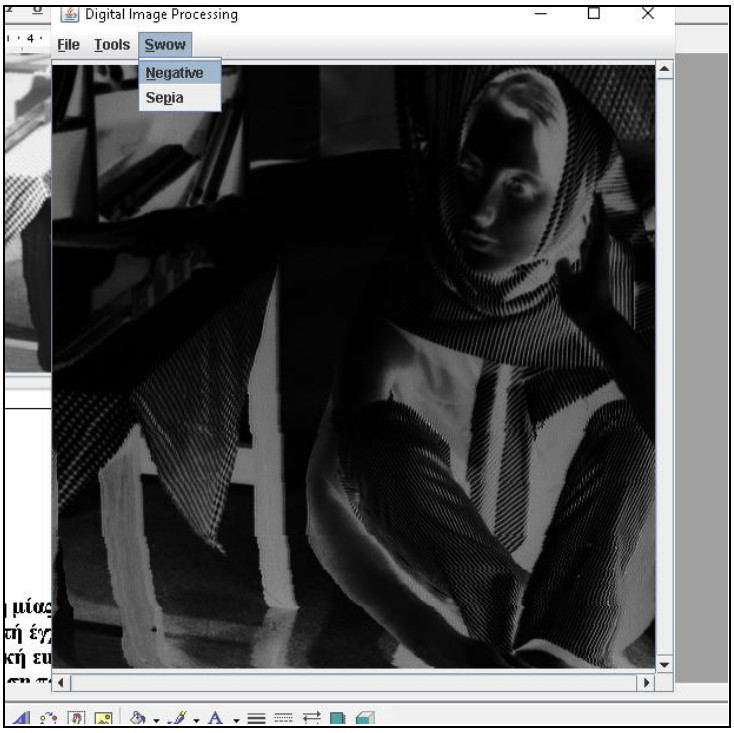
1. παίρνω την τιμή ενός εικονοστοιχείου RGB
2. υπολογίζω τις νέες τιμές για τις τρεις συνιστώσες Rnew, Gnew, Bnew
3. Αντικαθιστώ τις νέες τιμές των τριών συνιστωσών στο εικονοστοιχείο που επέλεξα
4. Επαναλαμβάνω για όλα τα εικονοστοιχεία της εικόνας μου από το βήμα 1 και σταματάω μόνο αν έχω διασχίσει όλη την εικόνα μου.

Επειδή η Java αντιμετωπίζει τις εικόνες με το μοντέλο ARGB θυμόμαστε ότι δεν θα αλλοιώσουμε την πρώτη συνιστώσα που αφορά την διαφάνεια (transparency) της εικόνας παρά θα επικεντρωθούμε στις τρεις χρωματικές συνιστώσες Red, Green και Blue. Θα προσθέσουμε λοιπόν μία νέα εντολή στη γραμμή μενού της εφαρμογής μας και θα τη συνδέσουμε με τον κώδικα που θα κάνει την εμφάνιση του αρνητικού της εικόνας μας.

Ανοίξαμε μία gray scale image για να φανεί καλύτερα η λειτουργία της εφαρμογής μας .



Εικόνα 16-Εμφάνιση εικόνας gray scale



Εικόνα 17-Εμφάνιση αρνητικού εικόνας

3.5 Μετατροπή έγχρωμης σε Sepia

Οι εικόνες sepia έχουν ταυτιστεί με εικόνες αντίκες. Οι φωτογραφίες αυτές είναι άμεσα αναγνωρίσιμες . Η φωτογράφος Ellen Fisch αναφέρει ότι η sepia μοιάζει με την ασπρόμαυρη στο ότι είναι μονόχρωμη, με τη διαφορά ότι είναι πιο καφέ ή πιο μαύρη. Ενώ οι ασπρόμαυρες εικόνες (εννοούμε τις gray scale images) χρησιμοποιούν αποχρώσεις τους του γκρι οι sepia εικόνες χρησιμοποιούν έναν κοκκινοκαφέ τόνο. Αυτό τους δίνει μία πιο απαλή και ονειρική αίσθηση (Fisch & Ballos, 2022).

Για να υπολογίσουμε την νέα τιμή sepia για κάθε εικονοστοιχείο θα χρησιμοποιήσουμε την επόμενη φόρμουλα (Parada, 2015).

newRed = 0.393*R + 0.769*G + 0.189*B

newGreen = 0.349*R + 0.686*G + 0.168*B

newBlue = 0.272*R + 0.534*G + 0.131*B

Σε κάθε νέα τιμή που βρίσκουμε θα κρατάμε το ακέραιο μέρος. Αν κάποια στιγμή οι πράξεις μας οδηγήσουν σε τιμή πάνω από 255 τότε ως τιμή κρατάμε το 255.

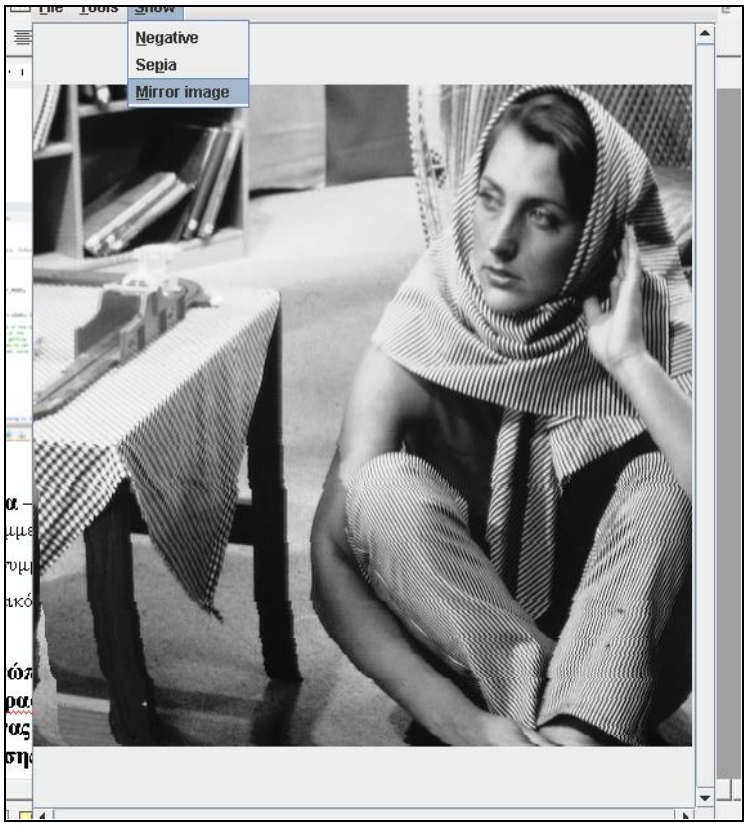
Οπότε η δουλειά μας τώρα είναι να διασχίσουμε όλο τον πίνακα με την εικόνα μας και να επικεντρωθούμε στις τρεις χρωματικές συνιστώσες της εικόνας μας τις οποίες θα αλλάξουμε. Την πρώτη συνιστώσα που αφορά τη διαφάνεια δεν θα την αλλάξουμε. Στη συνέχεια θα εμφανίσουμε την sepia εικόνα που δημιουργήσαμε στο παράθυρο της εφαρμογής μας.

Εικόνα 18-sepia(1)

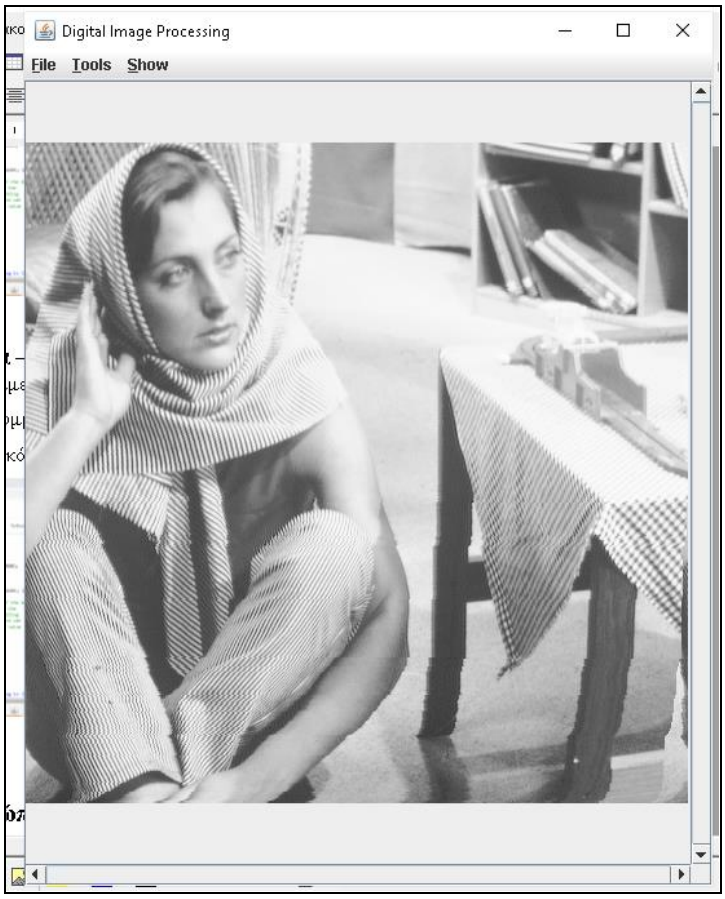
Εικόνα 19-sepia(2)

3.6 Συμμετρική εικόνα –καθρεπτισμός

Για να κάνουμε τη συμμετρική εικόνα θα μετακινήσουμε τα εικονοστοιχεία από τη μία θέση στην αντίστοιχη συμμετρική τους. Είναι το ίδιο αποτέλεσμα με το να κάναμε οριζόντια περιστροφή σε μία εικόνα.



Εικόνα 20-εικόνα πριν τη περιστροφή

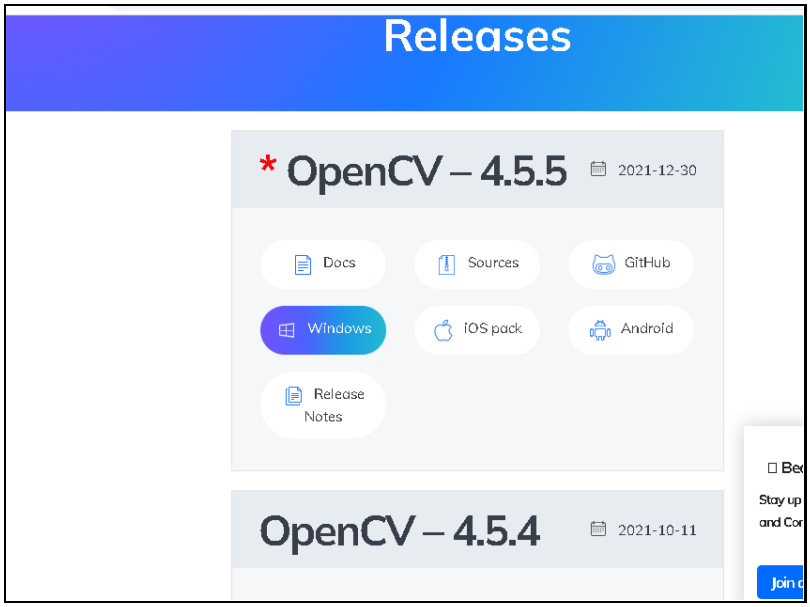


Εικόνα 21-εικόνα μετά την περιστροφή

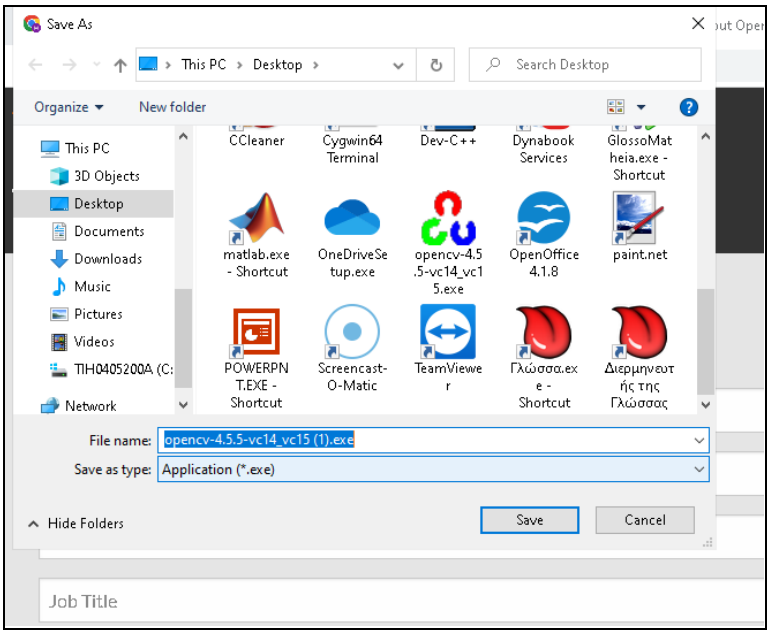
3.7 Αναγνώριση προσώπου σε εικόνα (facedetection)

Στη συνέχεια θα προσθέσουμε μία επιπλέον λειτουργικότητα στην εφαρμογή μας, αυτή της αναγνώρισης προσώπου (face detection). Για να επιτύχουμε αυτή τη λειτουργικότητα θα πρέπει πρώτα να εγκαταστήσουμε την OpenCV βιβλιοθήκη στο NetBeans για να μπορούμε να χρησιμοποιήσουμε ένα πλήθος από κλάσεις και συναρτήσεις. Το όνομα OpenCV προέρχεται από τις λέξεις Open Source Computer Vision . Η βιβλιοθήκη αυτή περιέχει ένα πλήθος μερικές εκατοντάδες αλγορίθμους για εικόνες και βίντεο όπως επίσης απαραίτητα αρχεία όπως είναι ένα σύνολο εικόνων από πρόσωπα που βοηθάνε το πρόγραμμά μας να κάνει τον εντοπισμό. Μόλις γίνει ο εντοπισμός τότε θα σχεδιάσουμε ένα ορθογώνιο που θα περικλείει το πρόσωπο που εντοπίστηκε (Open Source Computer Vision, 2022).

Για να κάνουμε λήψη της βιβλιοθήκης επισκεφθήκαμε τον ιστότοπο https://opencv.org/releases/

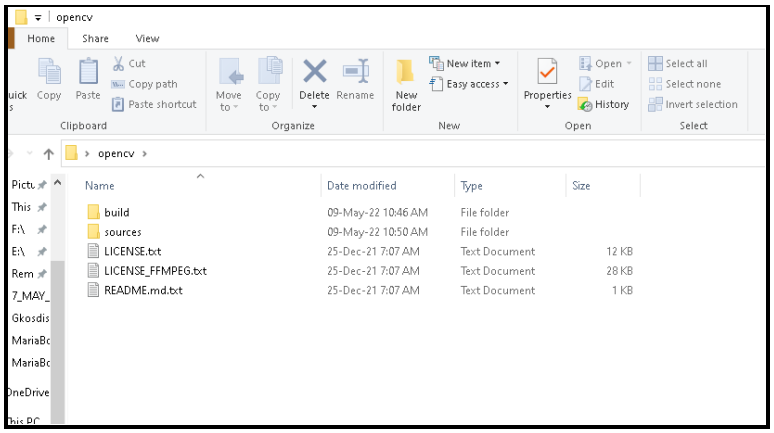


Εικόνα 22-OpenCV



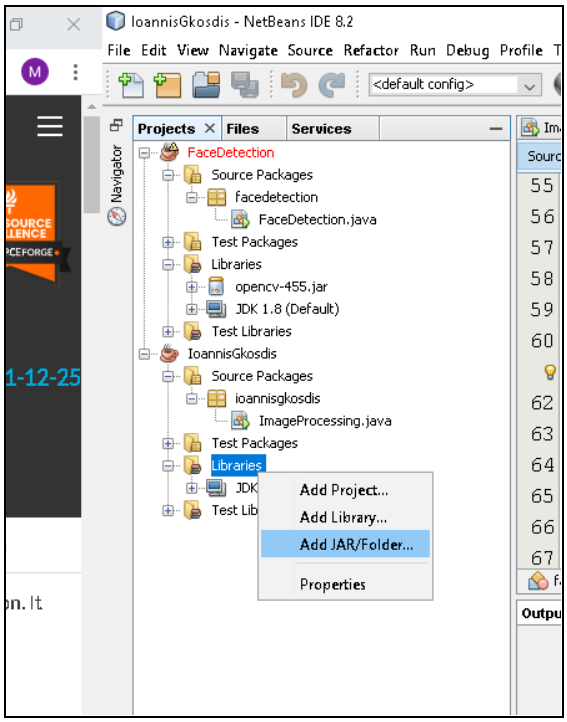
Εικόνα 23-Αρχείο εγκατάστασης OpenCV

Θα εκτελέσουμε το αρχείο αυτό και θα αποσυμπιέσουμε τα αρχεία στην επιφάνεια εργασίας. Παραλείπουμε για λόγους συντομίας τα εύκολα αυτά βήματα και βλέπουμε ότι στην επιφάνεια εργασίας μας έχει δημιουργηθεί ένας φάκελος με όνομα opencv με τα περιεχόμενα του όπως δείχνει η επόμενη εικόνα. Έτσι αν ο υπολογιστής μας έχει όνομα john και τοποθετήσουμε τον φάκελο στην επιφάνεια εργασίας τότε η πλήρης διαδρομή του θα είναι C:\Users\john\Desktop\opencv και περιέχει τα ακόλουθα αρχεία και υποφακέλους που φαίνονται (εικ.24).



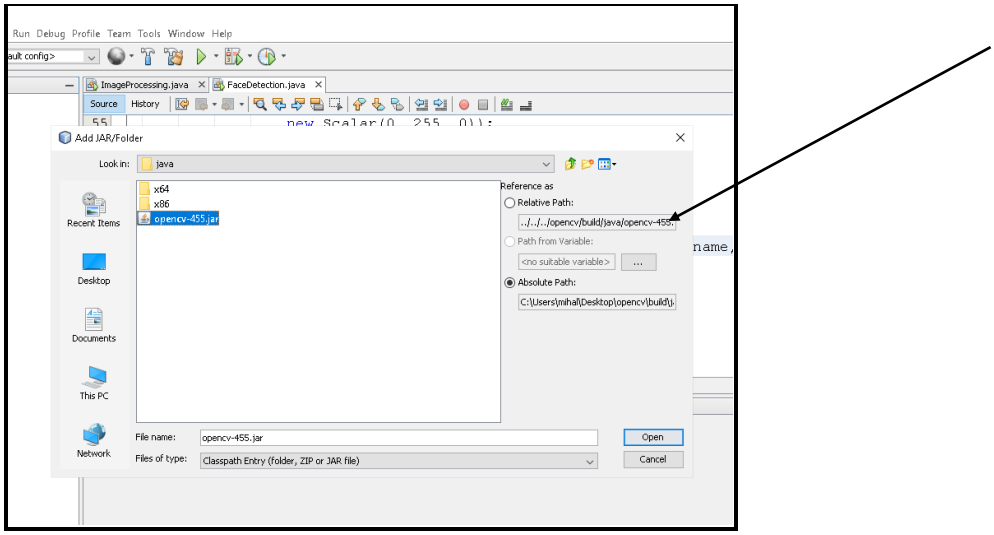
Εικόνα 24-Περιεχόμενα φακέλου OpenCV

Στη συνέχεια πάμε στο πρόγραμμα NetBeans όπου κάνουμε τις ακόλουθες ρυθμίσεις. Επιλέγουμε το project στο οποίο εργαζόμαστε και στον φάκελο Libraries κάνουμε δεξί κλικ και επιλέγουμε να προσθέσουμε ένα αρχείο με επέκταση *.jar



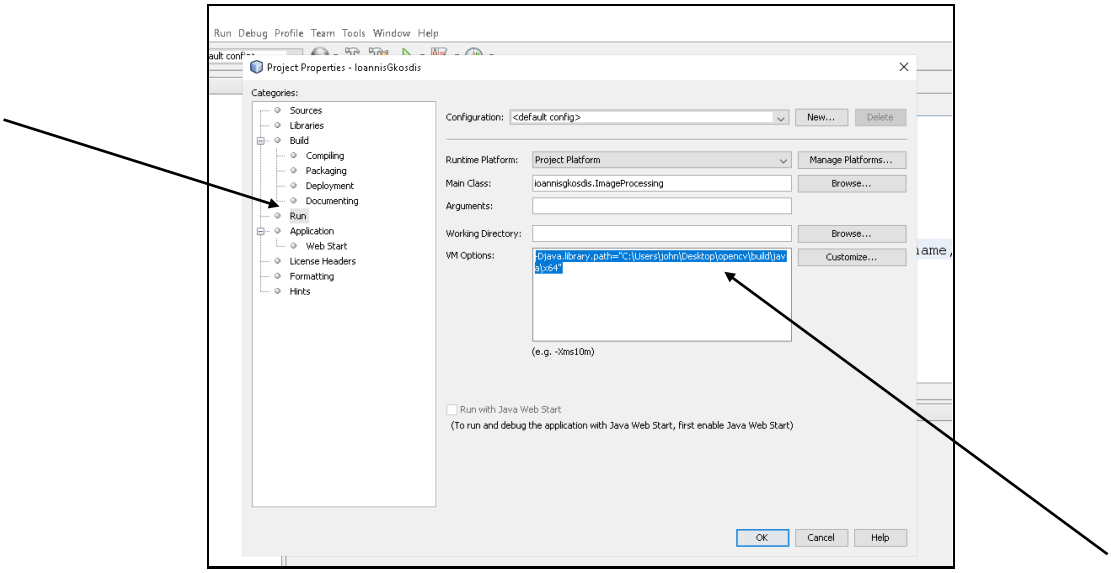
Εικόνα 25-Προσθήκη JAR αρχείου

Το αρχείο αυτό που θέλουμε να βρούμε είναι μέσα στον φάκελο opencv που δημιουργήθηκε πριν λίγο στην επιφάνεια εργασίας και πιο συγκεκριμένα στη διαδρομή \opencv\build\java όπως φαίνεται και στην επόμενη εικόνα.



Εικόνα 26-Επιλογή αρχείου opencv-455.jar

Επιλέγουμε το αρχείο και πατάμε Open. Στη συνέχεια θα κάνουμε δεξί κλικ στο όνομα του project και θα επιλέξουμε properties όπου θα πατήσουμε στο Run και θα προσθέσουμε και άλλη μία ιδιότητα όπως φαίνεται στην επόμενη εικόνα.



Εικόνα 27-Επιπλέον ρυθμίσεις παραμέτρου Run

Η ρύθμιση που προσθέσαμε ήταν η : -Djava.library.path="C:\Users\User\OneDrive\Desktop\opencv\build\java\x64". Επίσης για να φορτωθούν σωστά τα αρχεία που βοηθάνε την εφαρμογή μας στον εντοπισμό του προσώπου θα πρέπει μέσα στον κώδικά μας να τρέξει σωστά η εντολή που φορτώνει το αρχείο xml “haarcascade_frontalface_alt.xml”. Για να μην έχουμε προβλήματα κατά το χρόνο εκτέλεσης θα πρέπει να προσθέσουμε ρητά όλο το μονοπάτι όπου υπάρχει το αρχείο αυτό. Αυτό θα το κάνουμε με την εντολή faceDetector.load("C:\\Users\\john\\Desktop\\opencv\\sources\\data\\haarcascades\\haarcascade_frontalface_alt.xml"); που υπάρχει πλέον στον κώδικά μας.

Τώρα είμαστε έτοιμοι να προσθέσουμε τη λειτουργικότητα στην εφαρμογή μας. Θα αναφέρουμε σε αυτό το σημείο τις κυριότερες εντολές που χρειαζόμαστε:

System.loadLibrary(Core.NATIVE_LIBRARY_NAME);

CascadeClassifier faceDetector = new CascadeClassifier();

Είναι η βιβλιοθήκη OpenCV καθώς και ο κατηγοριοποιητής που θα εντοπίσει τα πρόσωπα

faceDetector.load("C:\\Users\\john\\Desktop\\opencv\\sources\\data\\haarcascades\\haarcascade_frontalface_alt.xml");

Στο αρχείο xml είναι πρόσωπα που βοηθάνε τον κατηγοριοποιητή faceDetector να εντοπίσει τα πρόσωπα.

Mat matimage = Imgcodecs.imread(selectedFile.getAbsolutePath());

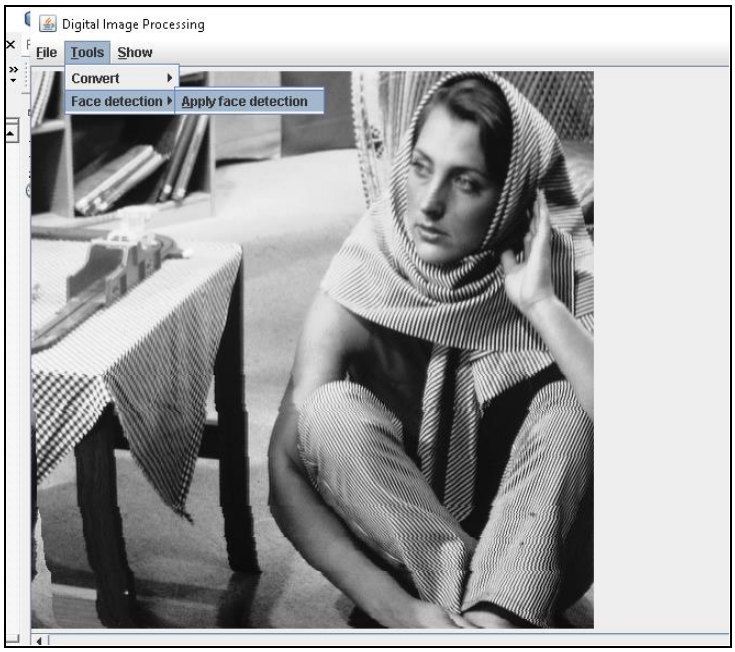
Mat είναι ένας τύπος για να φορτώσω την εικόνα μου. Είναι ένας πολυδιάστατος πίνακας.

MatOfRect faceDetections = new MatOfRect();

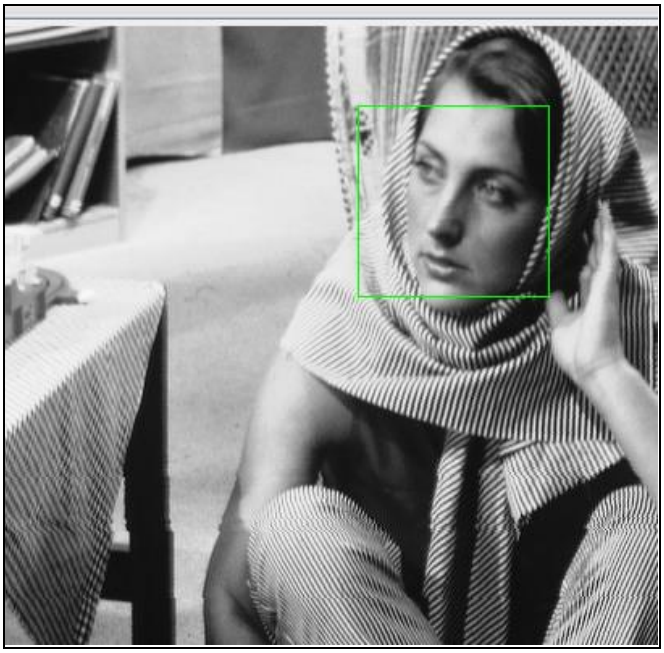
Στο αντικείμενο τύπου MatOfRect αποθηκεύω τα πρόσωπα που εντοπίζω. Αν αυτός ο πίνακας έχει μήκος μετά την εκτέλεση ίσο με μηδέν σημαίνει ότι δεν εντοπίστηκε κανένα πρόσωπο.

faceDetector.detectMultiScale(matimage, faceDetections);

Η συνάρτηση detectMultiScale εντοπίζει τα πρόσωπα

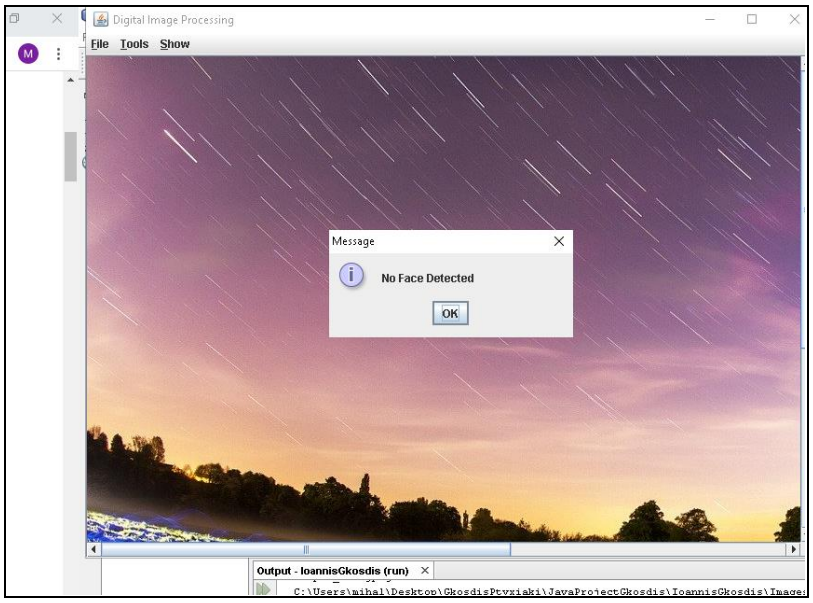


Εικόνα 28-Ανοιγμα εικόνας για face detection



Εικόνα 29-Εντοπισμός προσώπου

Αν επαναλάβουμε το ίδιο για μια εικόνα που δεν περιέχει πρόσωπο θα πάρουμε σχετικό μήνυμα.



Εικόνα 30-Περίπτωση χωρίς εντοπισμό προσώπου

3.8 Εισαγωγή υδατογραφήματος σε εικόνα (watermark)

Στη συνέχεια θα προσθέσουμε άλλη μία εντολή μενού που θα εισάγει ένα κείμενο της επιλογής μας ως υδατογράφημα (watermark) στο μέσο της εικόνας που θα έχουμε ανοίξει. Για να γίνει αυτό θα εργαστούμε με ένα αντικείμενο της κλάσης Graphics το οποίο ουσιαστικά θα το συνδέσουμε με ένα αντικείμενο BufferedImage δηλαδή με κάποια εικόνα.

Παραθέτουμε τις πιο σημαντικές εντολές που υλοποιούν τη λειτουργία αυτή. Στον κώδικά μας εμείς δίνουμε την δυνατότητα στον χρήστη να εισάγει από το πληκτρολόγιο το κείμενο που θα εμφανίζεται.

Δημιουργούμε ένα αντικείμενο graphics για να γράψουμε πάνω σε αυτό

```
Graphics graphics = image.getGraphics();
```

Τοποθετώ την εικόνα μου στο σημείο 0,0 του αντικειμένου graphics

```
graphics.drawImage(image, 0, 0, null);
```

Ορίζω γραμματοσειρά και χρώμα κειμένου

```
graphics.setFont(new Font("Arial", Font.PLAIN, 80));
```

Ορίζω τα χρώματα για το κείμενό μου (κόκκινο , πράσινο, μπλε, διαφάνεια) (Class Color,2022).

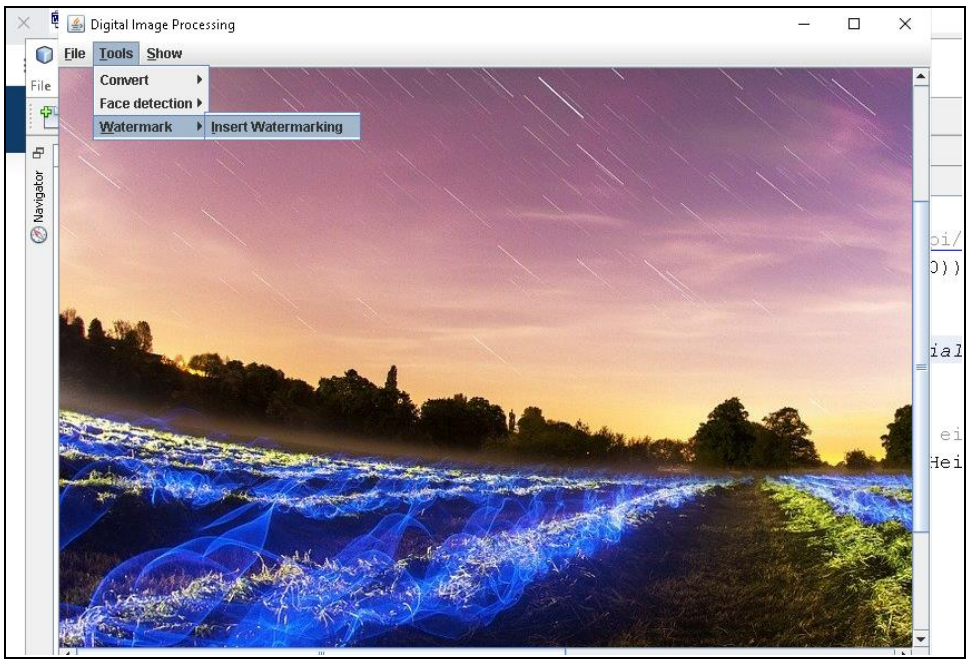
```
graphics.setColor(new Color(255, 0, 0, 140));
```

Ορίζω το κείμενο που θα εμφανιστεί. Στην εφαρμογή μας θα το εισάγουμε από το πληκτρολόγιο.

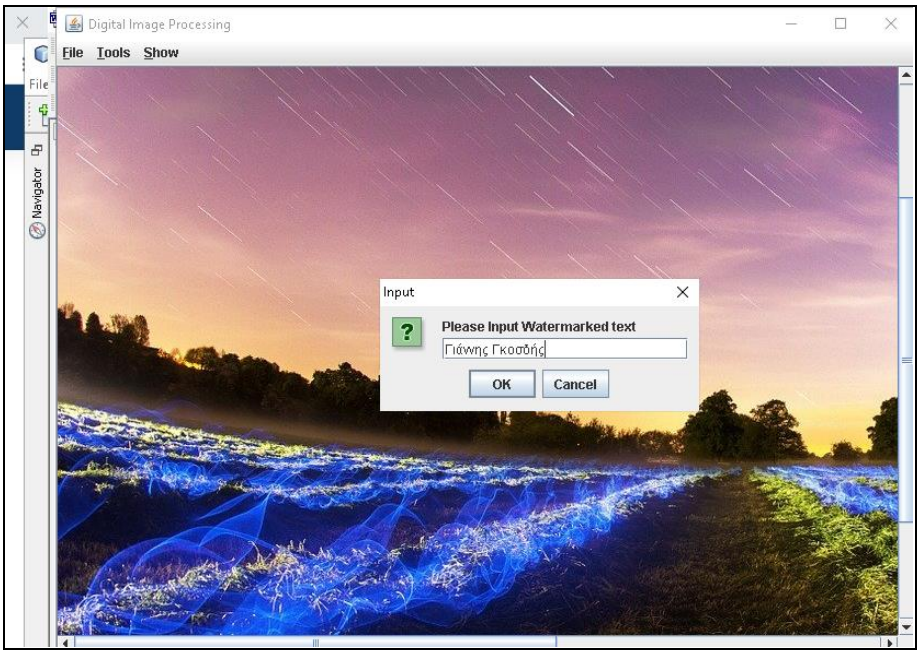
```
String watermark = "John Gkosdis";
```

Τοποθετώ το υδατογράφημα τέρμα αριστερά και στο μέσο της εικόνας

```
graphics.drawString(watermark,0,image.getHeight()/2);
```



Εικόνα 31-Εισαγωγή υδατογραφήματος(1)



Εικόνα 32-Εισαγωγή υδατογραφήματος(2)

Εικόνα 33-Εισαγωγή υδατογραφήματος(3)

3.9 Περιστροφή εικόνας

Λέγοντας εδώ περιστροφή εικόνας ονομάζουμε την περιστροφή κατά τον άξονα y'Oy και κατά τον x'Ox καθώς και τον συνδυασμό των δύο αυτών. Αυτό ίσως να το έχουμε συναντήσει σε εφαρμογές κινητών ή στις γνωστές εφαρμογές photoshop καθώς και paint.Net ως flip vertical και flip horizontal. Πάλι σε αυτόν τον μετασχηματισμό θα χρησιμοποιήσουμε τη γνωστή βιβλιοθήκη OpenCV όπου θα φορτώσουμε μια εικόνα αληθινού χρώματος σε ένα αντικείμενο τύπου Mat , θα κάνουμε την περιστροφή κάθετα δίνοντας την κατάλληλη παράμετρο και στη συνέχεια θα εμφανίσουμε την περιστραμμένη εικόνα. Προφανώς και αυτός ο μετασχηματισμός θα προσαρμοστεί με τη σχετική εντολή στη γραμμή μενού και θα συνδυαστεί με την αντίστοιχη συνάρτηση ακροατή γεγονότων (action listener).

Στη συνέχεια θα επικεντρωθούμε στις κυριότερες εντολές που χρειαστήκαμε για αυτόν τον μετασχηματισμό και θα δείξουμε και σχετικά αποτελέσματα από την εκτέλεση του προγράμματος (Class Core,2022).

Δημιουργώ ένα αντικείμενο Mat από την εικόνα που επέλεξα

```
Mat  src = Imgcodecs.imread(selectedFile.toPath().toString());
```

Δημιουργώ ένα αντικείμενο Mat για την εικόνα που θα περιστραφεί.

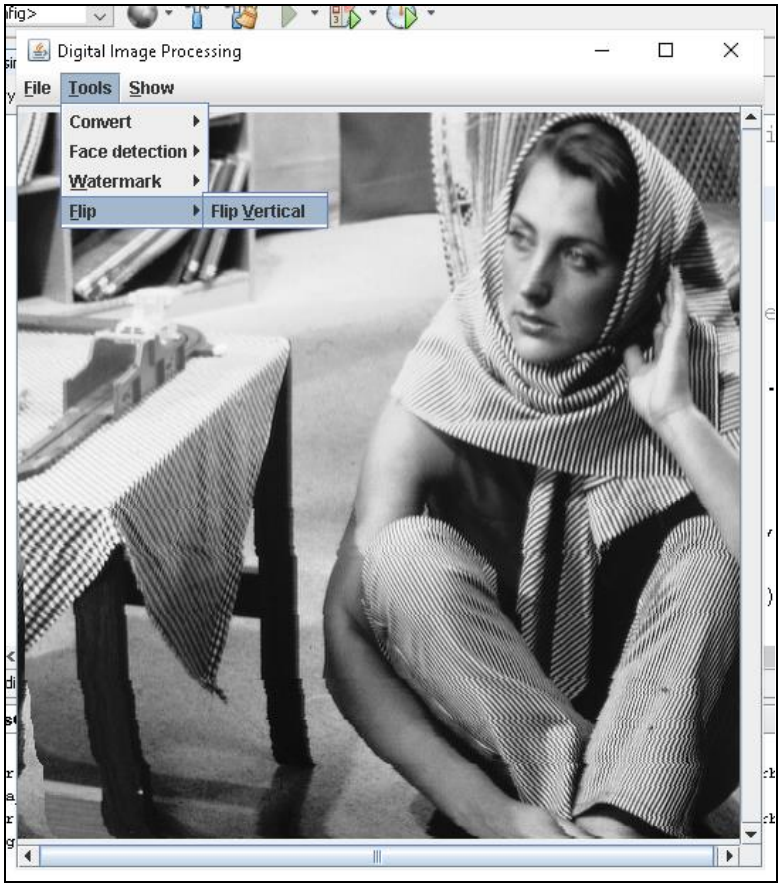
```
Mat dst = new Mat();
```

Εδώ βάζω την παράμετρο 0 και κάνω την περιστροφή κάθετα. Με παράμετρο 1 είναι οριζόντια και με παράμετρο -1 είναι οριζόντια και κάθετα μαζί

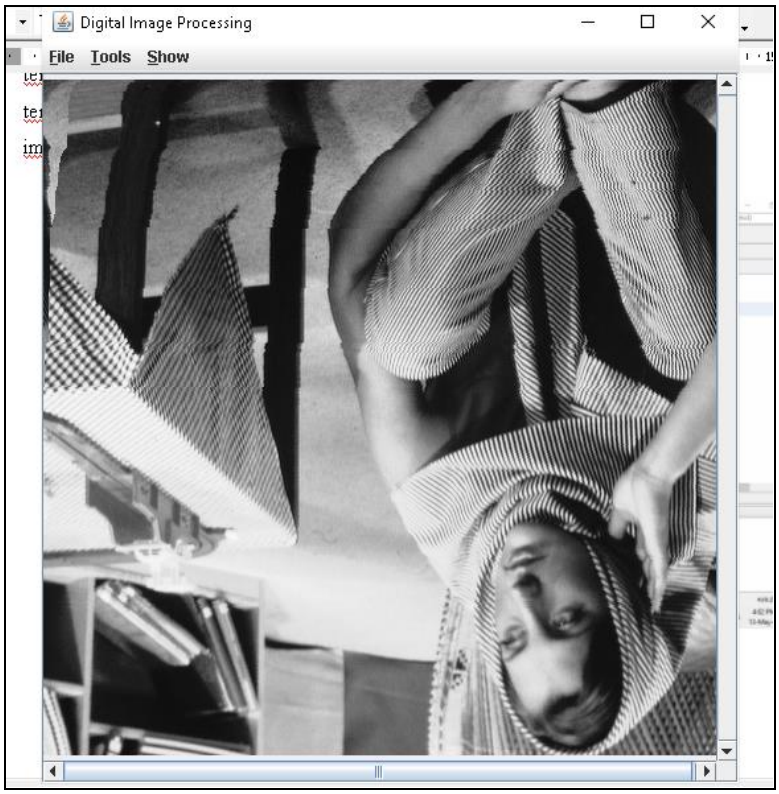
```
Core.flip(src, dst,0);
```

Το αντικείμενο dst που είναι τύπου Mat το μετατρέπω σε BufferedImage και το φορτώνω στο κυρίως παράθυρο της εφαρμογής.

```
byte[] newData= new byte[dst.rows() * dst.cols() * (int)(dst.elemSize())];
dst.get(0, 0, newData);
tempimage = new BufferedImage(dst.cols(), dst.rows(), 5);
tempimage.getRaster().setDataElements(0,0,dst.cols(),dst.rows(),newData);
imgLabel.setIcon(new ImageIcon(tempimage));
```



Εικόνα 34-Flip Vertical (1)



Εικόνα 35-Flip Vertical (2)

3.10 Ενίσχυση αντίθεσης

Η ενίσχυση αντίθεσης είναι μία διαδικασία που παίρνει το ιστόγραμμα συχνοτήτων από τιμές φωτεινότητας μίας εικόνας και ουσιαστικά φροντίζει να το διορθώνει κατά κάποιο τρόπο. Το ιστόγραμμα συχνοτήτων απλά παρουσιάζει τη συχνότητα που έχουν οι τιμές φωτεινότητας από [0..255] για όλα τα εικονοστοιχεία της εικόνας μας. Αν η εικόνα μας είναι μία gray scale image τότε όλα είναι πολύ απλά. Αν έχουμε μία εικόνα αληθινού χρώματος τότε δεν υπάρχει τιμή φωτεινότητας για να αλλοιώσουμε. Για το λόγο αυτό μετατρέπουμε την εικόνα σε gray scale , κάνουμε την εξισορρόπηση ιστογράμματος και στη συνέχεια μετατρέπουμε την εικόνα σε RGB. Μετά την εξισορρόπηση ιστογράμματος το ιστόγραμμα ουσιαστικά επεκτείνεται σε όλο το φάσμα τιμών φωτεινότητας πράγμα που σημαίνει ότι έχουμε ενισχύσει την αντίθεση στην εικόνα μας αφού πλέον υπάρχουν εικονοστοιχεία και πολύ φωτεινά και πολύ σκοτεινά.

Ο αλγόριθμος (Histogram Equalization, χ.η) για την εξισορρόπηση ιστογράμματος είναι ο εξής:

- Υπολογίζουμε τις αθροιστικές συχνότητες των τιμών φωτεινότητας για όλα τα εικονοστοιχεία.
- Κανονικοποιούμε διαιρώντας τις αθροιστικές συχνότητες με το πλήθος όλων των εικονοστοιχείων
- Πολλαπλασιάζουμε κάθε τιμή του προηγούμενου βήματος με τη μεγαλύτερη τιμή που είναι το 255 και στρογγυλοποιούμε στον κοντινότερο ακέραιο.

Για να κάνουμε λίγο πιο κατανοητή τη λειτουργία αυτή θα υποθέσουμε ότι έχουμε μία εικόνα που χρησιμοποιεί βάθος χρώματος 3 , δηλαδή η μικρότερη τιμή είναι 0 και η μεγαλύτερη 7 και με διαστάσεις 6*9=54 εικονοστοιχεία όπως δείχνουμε στον επόμενο πίνακα.

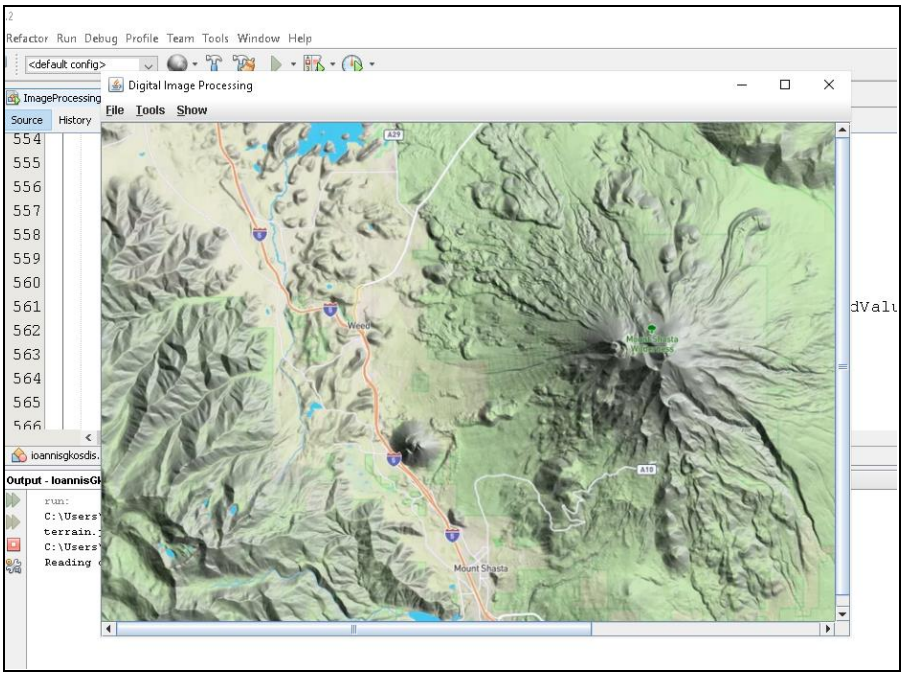
2	1	1	3	0	2	5	7	1
0	4	0	4	4	1	6	7	0
2	4	3	2	4	4	1	7	6
4	0	4	1	0	4	7	6	6
4	4	2	2	4	1	0	7	2
0	4	0	1	2	2	4	7	0

Εφαρμόζουμε τον αλγόριθμο και δείχνουμε τις νέες τιμές μετά από εξισορρόπηση ιστογράμματος.

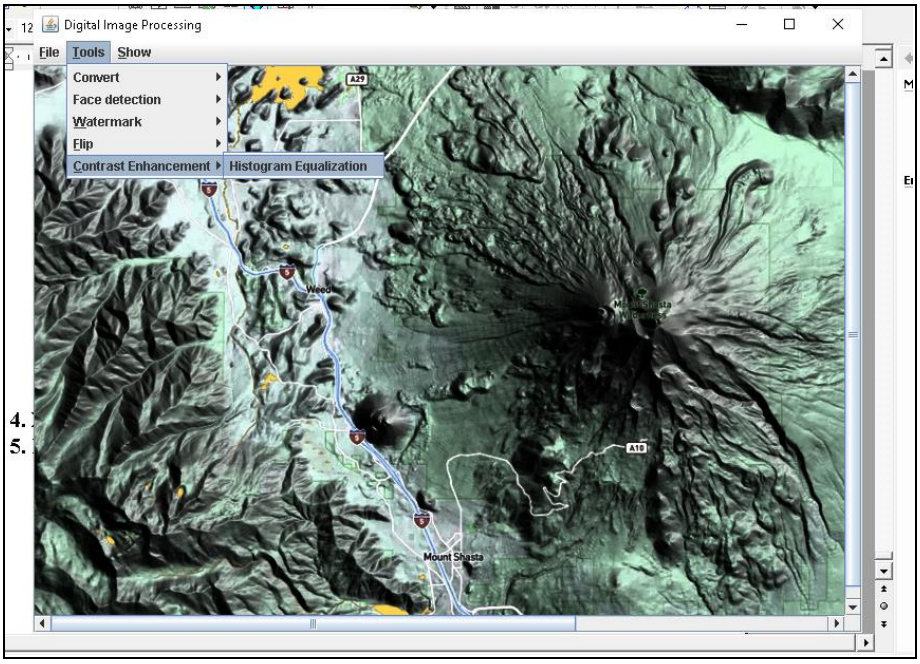
Τιμή gray	0	1	2	3	4	5	6	7
Συχνότητα(f)	10	8	9	2	14	1	4	6
Αθροιστική Συχνότητα (F)	10	18	27	29	43	44	48	54
F/54	10/54	18/54	27/54	29/54	43/54	44/54	48/54	54/54
(F/54)*max_value	1	2	4	4	6	6	6	7

Old pixel	0	1	2	3	4	5	6	7
New pixel	1	2	4	4	6	6	6	7

Στην εφαρμογή μας αφού προσθέσουμε την σχετική επιλογή μενού θα φορτώσουμε μία εικόνα RGB που δείχνει το ανάγλυφο ενός χάρτη. Στη συνέχεια θα φορτώσουμε την εικόνα αυτή σε ένα αρχείο Mat της OpenCV και θα το μετατρέψουμε σε ένα άλλο αντικείμενο της ίδιας κλάσης , αλλά στο χρωματικό μοντέλο HSV. Η μετατροπή αυτή είναι απαραίτητη για να κάνουμε εξισορρόπηση στο ιστόγραμμα της εικόνας μας. Στο σημείο αυτό να συμπληρώσουμε ότι η OpenCV δεν έχει δικιά της συνάρτηση που να κάνει εξισορρόπηση σε RGB εικόνα. Η μοναδική συνάρτηση που μας παρέχει είναι η equalizeHist που δέχεται μόνο gray scale εικόνες. Για το λόγο αυτό μετά τη μετατροπή σε HSV μοντέλο κρατήσαμε την 3^{η} συνιστώσα σε έναν πίνακα όπου εφαρμόσαμε τον άνω αλγόριθμο εξισορρόπησης. Όταν τελειώσαμε με τις νέες εξισορροπημένες τιμές την εικόνα την οποία είχαμε σε μορφή αντικειμένου Mat τότε μετατρέψαμε το αντικείμενο αυτό σε RGB φορμάτ, το εγγράψαμε σε αρχείο και το εμφανίσαμε στην οθόνη. Στη συνέχεια ενδιαφέρον είχε το οπτικό αποτέλεσμα που καταφέραμε με τον κώδικά μας που κάνει πιο ευανάγνωστο το εδαφικό ανάγλυφο της αρχικής εικόνας μετά από την εξισορρόπηση που εφαρμόσαμε με τον κώδικά μας.



Εικόνα 36-πριν την εξισορρόπηση ιστογράμματος

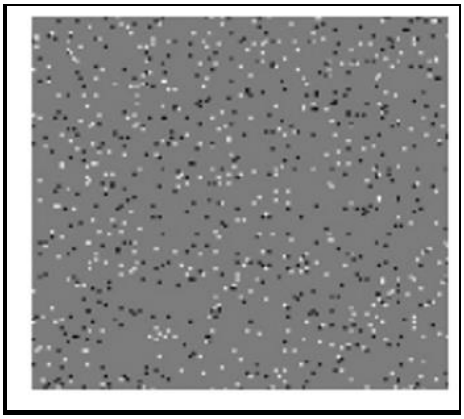


Εικόνα 37-μετά την εξισορρόπηση ιστογράμματος

3.11 Αφαίρεση θορύβου με εφαρμογή φίλτρου

Στη συνέχεια θα προσθέσουμε στην εφαρμογή μας άλλη μια λειτουργικότητα. Θα δημιουργήσουμε με δικό μας κώδικα ένα φίλτρο μέσου όρου μεταβλητής γειτονίας και θα το εφαρμόσουμε σε μια εικόνα στην οποία θα έχουμε προσθέσει νωρίτερα θόρυβο με δικό μας κώδικα επίσης. Πριν συνεχίσουμε με λεπτομέρειες από την υλοποίηση θα αναφερθούμε στις έννοιες θόρυβος σε ψηφιακή εικόνα και στην έννοια του φίλτρου.

Ως θόρυβο σε μια εικόνα εννοούμε την πληροφορία αυτή που δεν χρειαζόμαστε και είναι αυτή που προκαλεί αλλοιώσεις στο οπτικό αποτέλεσμα. Σε αντιστοιχία είναι σαν να έχουμε δυνατά την ένταση σε ένα ηχητικό κομμάτι και να συνειδητοποιούμε ότι ταυτόχρονα ακούγεται και άλλος ένας ενοχλητικός ήχος (Digital Camera Image Noise-Part 1, 2020). Με όρους φωτογράφων ο θόρυβος είναι κάτι σαν παραμόρφωση στην εικόνα μας . Μοιάζει σαν να υπάρχουν κόκκοι πάνω στην εικόνα μας ή ακόμα χειρότερα θυμίζει κηλίδα. Ο θόρυβος χειροτερεύει όταν η λήψη γίνεται σε συνθήκη χαμηλού φωτισμού (Dam, 2022). Στην ακόλουθη εικόνα δείχνουμε τον θόρυβο που ονομάζεται salt and pepper .Ουσιαστικά είναι σαν να έχω στην εικόνα μου τυχαίες κουκίδες άσπρες ή μαύρες (Lendave, 2021).

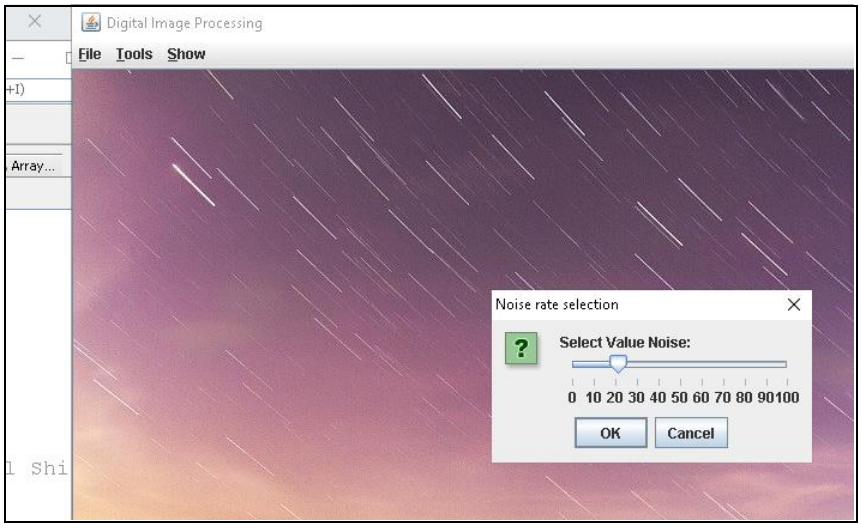


Εικόνα 38-Θόρυβος Salt & Pepper

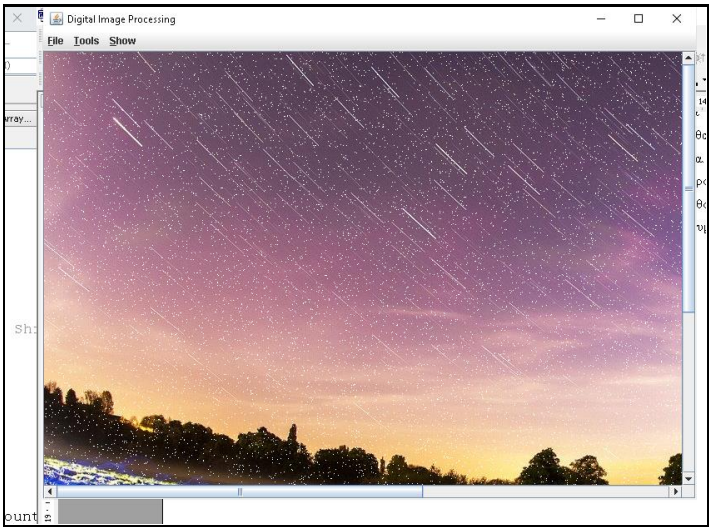
Ο θόρυβος μπορεί να μειωθεί ικανοποιητικά με χρήση φίλτρων. Το φίλτρο, γενικότερα , είναι ένας μηχανισμός που θα πάρει ως είσοδο μία εικόνα και θα προσπαθήσει να αλλοιώσει κάποια χαρακτηριστικά της εικόνας όπως είναι το μέγεθός της, το χρώμα της , τη σκιά της ή άλλα χαρακτηριστικά της. Τυπικά ένα φίλτρο εικόνας εργάζεται σε επίπεδο εικονοστοιχείου όπου κάνει αλλαγές στην εικόνα στο επίπεδο

τιμών του pixel (Image Filter, 2022). Στην εφαρμογή μας υλοποιήσαμε το φίλτρο μέσου όρου με επιλογή γειτονιάς 3,5 ή 7 εικονοστοιχείων. Το φίλτρο αυτό διασχίζει κάθε ένα εικονοστοιχείο στην εικόνα μου και αντικαθιστά την τιμή του με το μέσο όρο των κελιών που είναι περιμετρικά του σε μορφή γειτονιάς με μέγεθος 3,5 ή 7. Το μέγεθος το επιλέγει ο χρήστης και όπως θα δούμε στη συνέχεια με διαφορετικά στιγμιότυπα εκτέλεσης παίζει ρόλο στο αποτέλεσμα.

Στην εφαρμογή μας αφού προσθέσαμε τις δύο σχετικές εντολές για προσθήκη θορύβου και την προσθήκη φίλτρου. Με την επιλογή του ποσοστού του θορύβου διατρέχουμε ένα ποσοστό από τα εικονοστοιχεία με τυχαία σειρά και τους δίνουμε τιμή (255,255,255) δηλαδή λευκό σε αυτά.

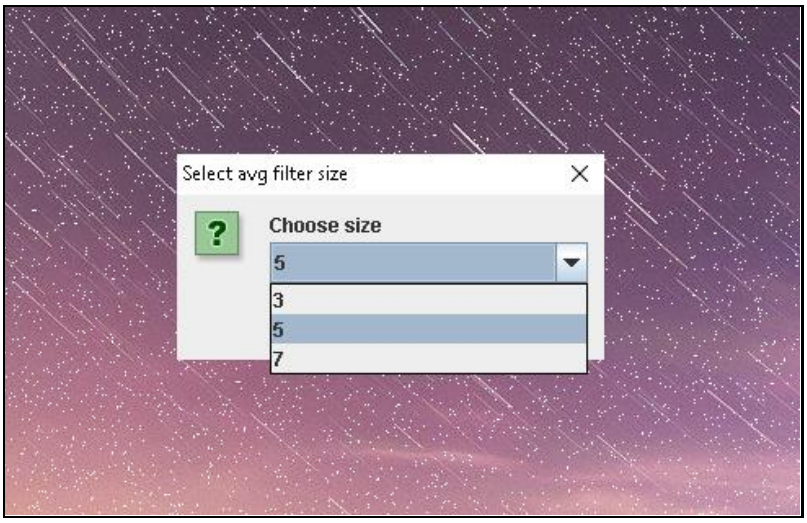


Εικόνα 39-Επιλογή ποσοστού θορύβου

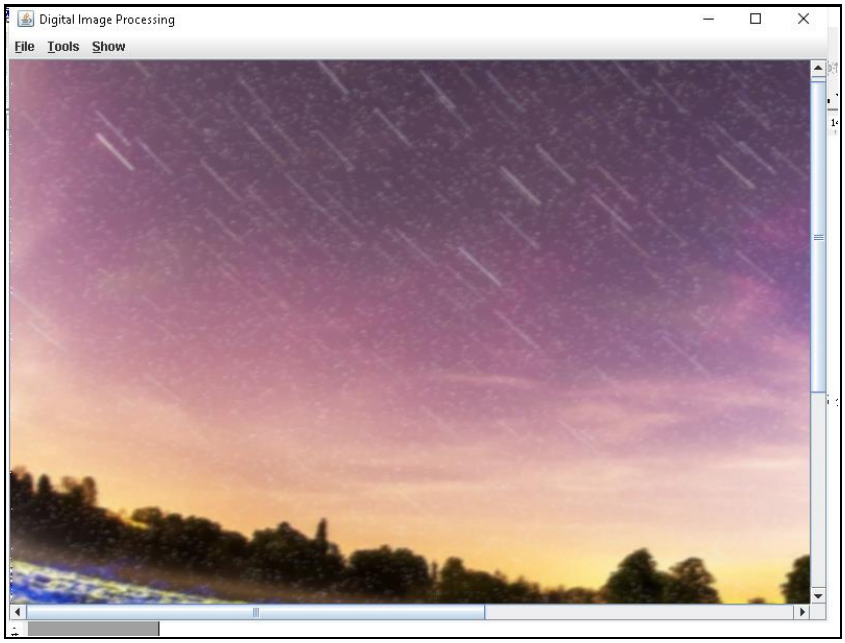


Εικόνα 40-Εικόνα μετά την προσθήκη θορύβου

Στη συνέχεια θα ζητήσουμε από το χρήστη να επιλέξει το μέγεθος της γειτονιάς που θα εφαρμοστεί το φίλτρο. Υπάρχουν τρεις επιλογές: για γειτονιά 3,5 ή 7 εικονοστοιχείων. Εδώ με την επιλογή του χρήστη γνωρίζαμε το μέγεθος της γειτονιάς των εικονοστοιχείων όπου θα εφαρμόζαμε το φίλτρο μέσου όρου. Αυτό που κάναμε ήταν να παίρνουμε αυτόν τον υποπίνακα που είχε διαστάσεις γειτονιά X γειτονιά X 3 και να τον δίνουμε ως παράμετρο σε μία συνάρτηση με όνομα getAvgOfNeighboorArray(NeighboorArray,NewColorValues), που θα μας επέστρεφε σε έναν πίνακα με τρία κελιά τις τιμές Red, Green και Blue από την νέα τιμή. Αγνοήσαμε την τιμή Alpha μιας και θεωρήσαμε ότι η τιμή transparency είναι ίδια σε όλη την εικόνα και δεν θα έπρεπε να αλλαχθεί.

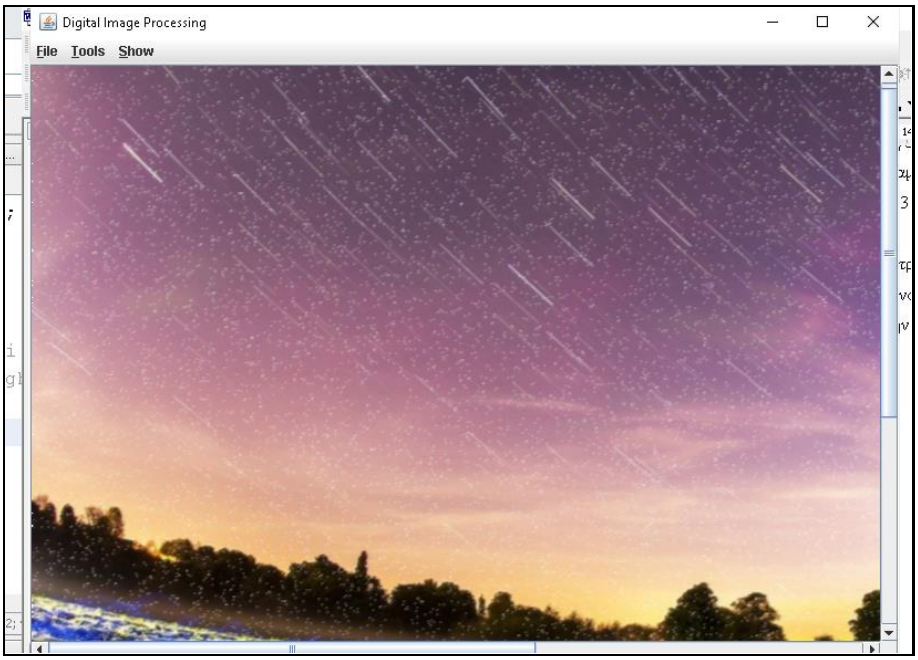


Μετά την εφαρμογή του φίλτρου στη θορυβώδη εικόνα με γειτονιά 5X5 πήραμε την επόμενη εικόνα

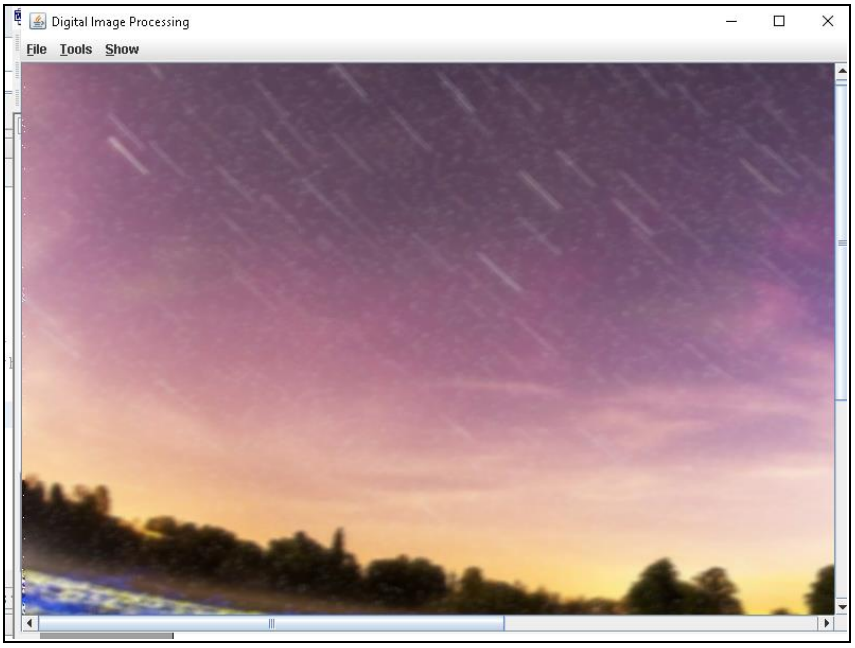


Εικόνα 41-Εφαρμογή φίλτρου μέσου όρου γειτονιάς 5X5

Επίσης επαναλάβαμε τη διαδικασία και με γειτονιά 3X3



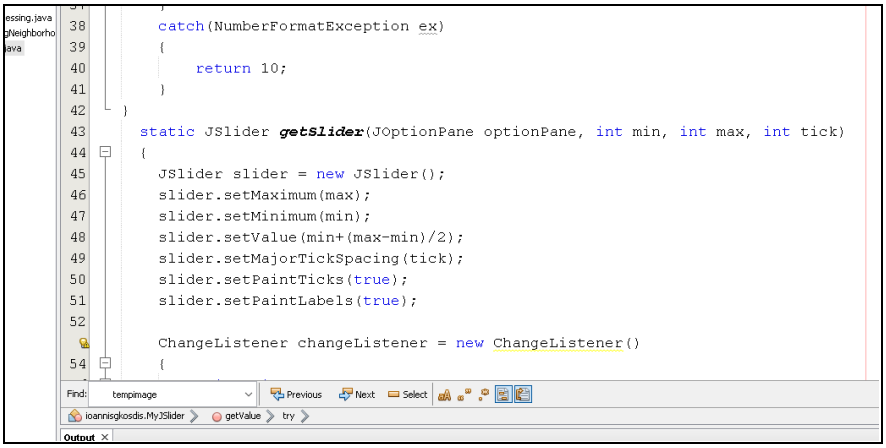
Εικόνα 42-Εφαρμογή φίλτρου μέσου όρου γειτονιάς 3X3



Εικόνα 43-Εφαρμογή φίλτρου μέσου όρου γειτονιάς 7X7

3.12 Αλλαγή φωτεινότητας εικόνας

Στη συνέχεια της εργασίας μας θα προσθέσουμε μία ακόμα εντολή στο μενού επιλογών για την αυξομείωση της φωτεινότητας. Η προσθήκη θα γίνει με τον ίδιο τρόπο που έγινε ως τώρα και για τις υπόλοιπες εντολές. Επίσης για την επιλογή της φωτεινότητας θα χρησιμοποιήσουμε ένα αντικείμενο της κλάσης MyJSlider που χρησιμοποιήσαμε και για την εισαγωγή τιμής θορύβου. Επειδή στο τμήμα αυτό για την αύξηση ή τη μείωση της φωτεινότητας θέλαμε να εμφανίζονται άλλες τιμές στον JSlider που έχουμε αποφασίσαμε να παραμετροποιήσουμε λίγο τη συνάρτηση δημιουργίας της MyJSlider έτσι ώστε να δέχεται ελάχιστη τιμή, μέγιστη τιμή καθώς και βήμα με το οποίο θα γίνεται η υποδιαίρεση της κλίμακας. Ο αλλαγμένος κώδικας είναι στην επόμενη εικόνα.



Εικόνα 44-Συνάρτηση getSlider()

Η getSlider() είναι μια συνάρτηση που χρησιμοποιεί η υπερφορτωμένη συνάρτηση δημιουργίας για να σχηματίσει το αντικείμενο έτσι όπως το θέλουμε.

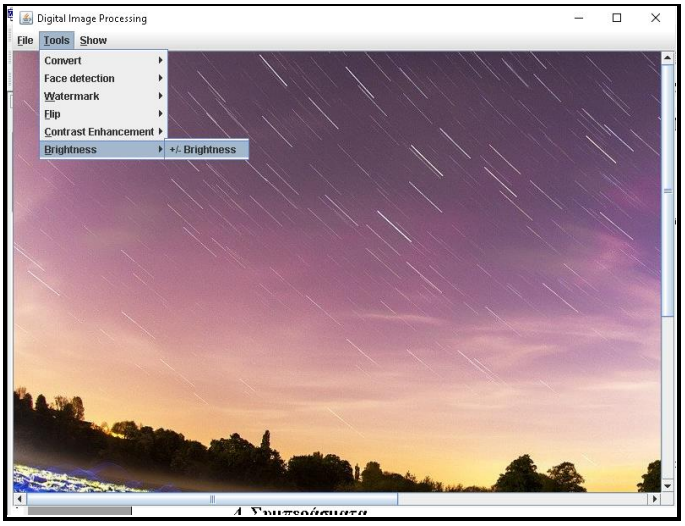
Μετά λοιπόν από τις κατάλληλες αυτές προσθήκες ανοίξαμε μία εικόνα και την φορτώσαμε σε ένα αντικείμενο BufferedImage. Στη συνέχεια με γνωστές σε εμάς τις διαστάσεις της εικόνας διατρέξαμε αυτό το αντικείμενο σαν να ήταν ένας διδιάστατος πίνακας. Σε κάθε εικονοστοιχείο που στοχεύαμε προσθέταμε την τιμή που είχε επιλέξει ο χρήστης για αύξηση ή μείωση της φωτεινότητας. Έτσι λοιπόν αν ο χρήστης επιλέξει προσθήκη φωτεινότητας +10 η τιμή σε όλα τα εικονοστοιχεία αυξάνεται κατά 10 , ενώ αν επιλέξει τιμή -10 τότε αναλόγως αυτή μειώνεται. Προσέξαμε επίσης να μην ξεφύγουν

οι τιμές πάνω από 255 όπως και κάτω από 0 . Τη δουλειά αυτή ανέλαβε μία επιπλέον συνάρτηση που φτιάξαμε με όνομα MakeValueInRange0_to_255(). Ο κώδικας της είναι στην ακόλουθη εικόνα :

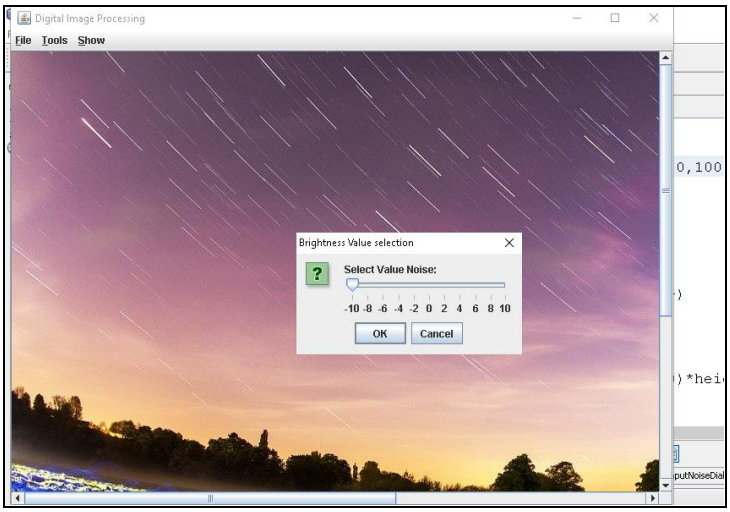
```
1060        frame.setJMenuBar(menuBar);
1061    }
1062    public static int MakeValueInRange0_to_255(int value)
1063        {
1064
1065            if (value < 0) {
1066                value = 0;
1067            }
1068            else if (value > 255) {
1069                value = 255;
1070            }
1071            return value;
1072        }
1073
```

Εικόνα 45-Συνάρτηση MakeValueInRange0_to_255()

Εκτελώντας λοιπόν των κώδικά μας όπως δείχνουμε στη συνέχεια μειώσαμε την φωτεινότητα από την εικόνα που επιλέξαμε .



Εικόνα 46-Αλλαγή φωτεινότητας (1)



Εικόνα 47-Αλλαγή φωτεινότητας (2)

Όπου η εικόνα μας έχασε λίγη από τη φωτεινότητά της.

4. Συμπεράσματα

Στην παρούσα εργασία ξεκινώντας περιμέναμε να συναντήσουμε πολλές δυσκολίες στην υλοποίηση όλων των μετασχηματισμών που επιχειρήσαμε. Ωστόσο είδαμε ότι η γλώσσα προγραμματισμού Java ξεπέρασε εύκολα αυτό το εμπόδιο. Η χρήση του πακέτου java.awt.image έχει ένα πλήθος από αντικείμενα, κλάσεις και μεθόδους που μας βοήθησαν να φορτώσουμε και να αναπαραστήσουμε μία εικόνα εύκολα στην εφαρμογή μας. Δεν χρειάστηκε δηλαδή να υλοποιήσουμε από την αρχή όλες τις δομές «ανακαλύπτοντας τον τροχό από την αρχή». Η φόρτωση των εικόνων γινόταν εύκολα συνδυάζοντας τις εντολές που μας παρέχει πακέτο java.awt.image. Στη συνέχεια κατασκευάσαμε για την εφαρμογή μας το κατάλληλο interface με χρήση των κλάσεων JFrame και JMenu όπως γνωρίζαμε από το μάθημα της Java. Η εφαρμογή αυτή θα μπορεί να εξαχθεί σε μία εκτελέσιμη μορφή *.jar και να διαμοιραστεί σε ένα μεγάλο πλήθος υπολογιστικών συσκευών, αρκεί να είναι εγκατεστημένη η Java.

Η γνώση που αποκτήσαμε ωστόσο κατά τη διάρκεια των σπουδών μας, μας έκανε να κατανοήσουμε την φύση της αναπαράστασης μιας εικόνας σε χαμηλότερο επίπεδο. Η αναπαράσταση μιας εικόνας ανάγεται σε μια αναπαράσταση σε έναν δισδιάστατο ή τρισδιάστατο πίνακα. Συνειδητοποιώντας το γεγονός αυτό μπορέσαμε να προχωρήσουμε στην υλοποίηση όλων των στοιχειωδών μετασχηματισμών επί των εικόνων. Ακόμα και αν υπήρχαν έτοιμες λύσεις σε κάποιους μετασχηματισμούς εμείς προσπαθήσαμε να δώσουμε λύσεις με χρήση δικών μας αλγορίθμων που θα επεξεργαζόταν αυτούς του δισδιάστατους πίνακες έτσι όπως μάθαμε κατά τη διάρκεια των σπουδών μας στη Java καθώς και σε άλλες γλώσσες προγραμματισμού. Οι γνώσεις μας από γραμμική άλγεβρα, οι πράξεις με δυαδικούς τελεστές (bitwise operators) και πράξεις επί των πινάκων που διδαχθήκαμε σε μαθήματα προγραμματισμού μας βοήθησαν επίσης στους μετασχηματισμούς που απαιτούσε η εργασία.

Η επιλογή της γλώσσας προγραμματισμού Java καθώς και του περιβάλλοντος NetBeans δεν έγινε τυχαία. Η Java μας παρείχε όλα τα εργαλεία για να αναπαραστήσουμε μια εικόνα καθώς και ένα πλήθος εργαλείων για να δημιουργήσουμε μια παραθυρική εφαρμογή. Το περιβάλλον ανάπτυξης NetBeans μας δημιούργησε ένα αίσθημα σιγουριάς και φιλικότητας. Το μεγάλο πλήθος εργαλείων του αρχικά ξαφνιάζει

τον αρχάριο χρήστη του , αλλά τα πάντα είναι τοποθετημένα σωστά σε μια στιβαρή εφαρμογή. Επίσης εντοπίσαμε πλούσιο αρχείο με πληροφορίες για εκμάθηση του όπως συνηθίζεται σε κάθε σοβαρή εφαρμογή.

Εν κατακλείδι, ολοκληρώνοντας την παρούσα εργασία μας, κατανοήσαμε ότι ο χώρος της επεξεργασίας της εικόνας με Java είναι πολύ μεγαλύτερος από όσο νομίζαμε στην αρχή όπως συμβαίνει με όλο τον χώρο της πληροφορικής και οι δυνατότητες είναι σχεδόν απεριόριστες. Η γνώση δεν έχει όρια και ανήκει σε όποιον αποφασίσει να ανοίξει την πόρτα και να ξεκινήσει αυτή την όμορφη διαδρομή.

5. Βιβλιογραφία

Ξενόγλωσση

Image Types in the Toolbox, (2022). Ανακτήθηκε από https://www.mathworks.com/help/images/image-types-in-the-toolbox.html#f14-33397

Convert a color photo to Grayscale mode, (2022). Ανακτήθηκε από https://helpx.adobe.com/photoshop/key-concepts/grayscale.html

Image Types, (2022). Ανακτήθηκε από https://ch.mathworks.com/help/matlab/creating_plots/image-types.html?searchHighlight=grayscale%20images&s_tid=srchtitle_grayscale%2520images_5

Batnagar, A. (2022). *The complete History of Java Programming Language*. Ανακτήθηκε από https://www.geeksforgeeks.org/the-complete-history-of-java-programming-language/

Dheerendra, K. (2022). *Difference between Abstract Class and Interface in Jav*. Ανακτήθηκε από https://www.geeksforgeeks.org/difference-between-abstract-class-and-interface-in-java/

How to Make Frames. (2022). Ανακτήθηκε από https://docs.oracle.com/javase/tutorial/uiswing/components/frame.html

Class BufferedImage, (2020). Ανακτήθηκε από https://docs.oracle.com/javase/7/docs/api/java/awt/image/BufferedImage.html

Java BufferedImage Class, (2022). Ανακτήθηκε από https://www.tutorialspoint.com/java_dip/java_buffered_image.htm

Agarwal, P. (2021). *Image Processing In Java – Get and Set Pixels*. Ανακτήθηκε από https://www.geeksforgeeks.org/image-processing-in-java-get-and-set-pixels/?ref=lbp

Understanding color models, (2013). Ανακτήθηκε από http://product.corel.com/help/Designer/540224261/Main/EN/Doc/wwhelp/wwhimpl/common/html/wwhelp.htm?context=Corel_DESIGNER_Help&file=Corel-DESIGNER-Understanding-color-models.html

Doughty, M. (2009). *Graphics Color Models*. Ανακτήθηκε από http://www.sketchpad.net/basics4.htm

Kennedy, E. (2020). *The HSB Color System: A Practitioner's Primer*. Ανακτήθηκε από https://learnui.design/blog/the-hsb-color-system-practicioners-primer.html

Types of Images, (2021). Ανακτήθηκε από https://www.javatpoint.com/dip-types-of-images

History of Java, (2021). Ανακτήθηκε από https://www.javatpoint.com/history-of-java

Bhatnagar, A. (2022*). The complete History of Java Programming Language*. Ανακτήθηκε από https://www.geeksforgeeks.org/the-complete-history-of-java-programming-language/

Class Definition in Java, (2021). Ανακτήθηκε από https://www.javatpoint.com/class-definition-in-java

What is a Class, (2022). Ανακτήθηκε από https://docs.oracle.com/javase/tutorial/java/concepts/class.html

Java Inheritance, (2022). Ανακτήθηκε από https://www.w3schools.com/java/java_inheritance.asp

Inheritance, (2022). Ανακτήθηκε από https://docs.oracle.com/javase/tutorial/java/IandI/subclasses.html

Method Overriding in Java, (2021). Ανακτήθηκε από https://www.javatpoint.com/method-overriding-in-java

Σφέτσος, Π. (χ.η). *Διασυνδέσεις / Διεπαφές*. Αντικειμενοστραφής Προγραμματισμός , Διεθνές Πανεπιστήμιο της Ελλάδος, Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων. Ανακτήθηκε από https://people.iee.ihu.gr/~sfetsos/OOP7.pdf

Java BufferedImage Class, (2022). Ανακτήθηκε από https://www.tutorialspoint.com/java_dip/java_buffered_image.htm

Sample JPEG Files Download, (2022). Ανακτήθηκε από https://filesamples.com/formats/jpeg

Class JScrollPane, (2020). Ανακτήθηκε από https://docs.oracle.com/javase/7/docs/api/javax/swing/JScrollPane.html

Bitwise and Bit Shift Operators, (2022). Ανακτήθηκε από https://docs.oracle.com/javase/tutorial/java/nutsandbolts/op3.html

Fisch, E. & Ballos, G. (2022). *Make an impression with sepia photography*. Ανακτήθηκε από https://www.adobe.com/creativecloud/photography/discover/sepia-photography.html

Parada, R. (2015). *How to create sepia image in gui?*. Ανακτήθηκε από https://ch.mathworks.com/matlabcentral/answers/256664-how-to-create-sepia-image-in-gui

Open Source Computer Vision, (2022). Ανακτήθηκε από https://docs.opencv.org/4.5.4/index.html

Class Color, (2022). Ανακτήθηκε από https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html

Class Core, (2022). Ανακτήθηκε από https://docs.opencv.org/3.4/javadoc/org/opencv/core/Core.html#flip(org.opencv.core.Mat,org.opencv.core.Mat,int)

Histogram Equalization, (χ.η) Ανακτήθηκε από https://www.uobabylon.edu.iq/eprints/paper_5_10128_257.pdf

Digital Camera Image Noise- Part 1, (2020). Ανακτήθηκε από https://www.cambridgeincolour.com/tutorials/image-noise.htm

Dam, P. (2022). *What is Photo Noise?*. Ανακτήθηκε από https://www.adorama.com/alc/what-is-noise-in-photography/#:~:text=What%20is%20Photo%20Noise%3F,and%20can%20ruin%20a%20photograph.

Lendave, V. (2021). *A Guide to Different Types of Noises and Image Denoising Methods*. Ανακτήθηκε από https://analyticsindiamag.com/a-guide-to-different-types-of-noises-and-image-denoising-methods/

Image Filter, (2022). Ανακτήθηκε από https://www.techopedia.com/definition/7687/image-filter