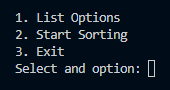
Το θέμα της εργασίας μας είναι η γραφική απεικόνιση του αλγορίθμου ταξινόμησης Insertion Sort, που μας παρέχει η βιβλιοθήκη Turtle της Python, μέσω της γραφικής διεπαφής του χρήστη.

Θέλαμε να χωρίσουμε την εργασία με τέτοιο τρόπο ώστε κάθε κομμάτι να λειτουργεί ξεχωριστά και να είναι αυτόνομο, αλλά όλα μαζί να απαρτίζουν ένα άρτιο πρόγραμμα. Έτσι κύριος στόχος μας ήταν η δημιουργία συναρτήσεων, ώστε να μπορού να γίνουν import στα άλλα αρχεία και να μπορούν να συνεργάζονται μεταξύ τους χωρίς όμως το ένα να επηρεάζει το άλλο. Η εργασία χωρίστηκε στα εξής 4 μέρη, ένα για κάθε μέλος της ομάδας:

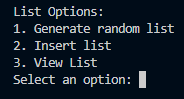
1. Το κύριο μενού στο οποίο ο χρήστης αλληλεπιδρά με το πρόγραμμα
2. Ο αλγόριθμος ταξινόμησης βήμα-βήμα
3. Το αρχείο turtle το οποίο θα είναι υπεύθυνο για τη σχεδίαση και μορφοποίηση
4. Σύνδεση του αλγορίθμου ταξινόμησης με το αρχείο σχεδίασης

Αρχικά για τη δημιουργία του κύριου μενού πήραμε ως ιδέα την άσκηση εργαστηρίου Ε4 με το calendar. Για την υλοποίηση του εισάγαμε τις βιβλιοθήκες os, απ την time τη συνάρτηση sleep και τέλος την random. Ο χρήστης θα μπορεί να επιλέξει με το (1) να δημιουργήσει μία λίστα, τυχαία ή δικιά του, με το (2) να ταξινομήσει τη λίστα και το (3) κλείνει το πρόγραμμα:

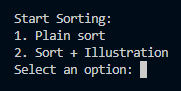


Το χτίσιμο του μενού έγινε με αμυντικό προγραμματισμό για τις περιπτώσεις που ο χρήστης δώσει λάθος είσοδο με στόχο να μην κλέινει λόγω error και να μπορεί να συνεχίσει να τρέχει κανονικά.

Με την επιλογή List Options ο χρήστης μπορεί να φτιάξει μια τυχαία λίστα όπου δίνει το εύρος των τιμών αλλά και το μήκος της λίστας ή να επιλέξει να εισάγει μία δικιά του, έχοντας την επιλογή να δώσει έναν-έναν τους αριθμούς. Τέλος υπάρχει η επιλογή να δεί ο χρήστης τη λίστα του:



Η επιλογή Start Sorting από το κεντρικό μενού δίνει στο χρήστη τη δυνατότητα να ταξινομήσει μια λίστα και να δεί βήμα-βήμα τη ταξινόμηση, γραφικά ή όχι. Άν δεν υπάρχει δοσμένη λίστα, αυτόματα το πρόγραμμα σε παραπέμπει στο List Options:

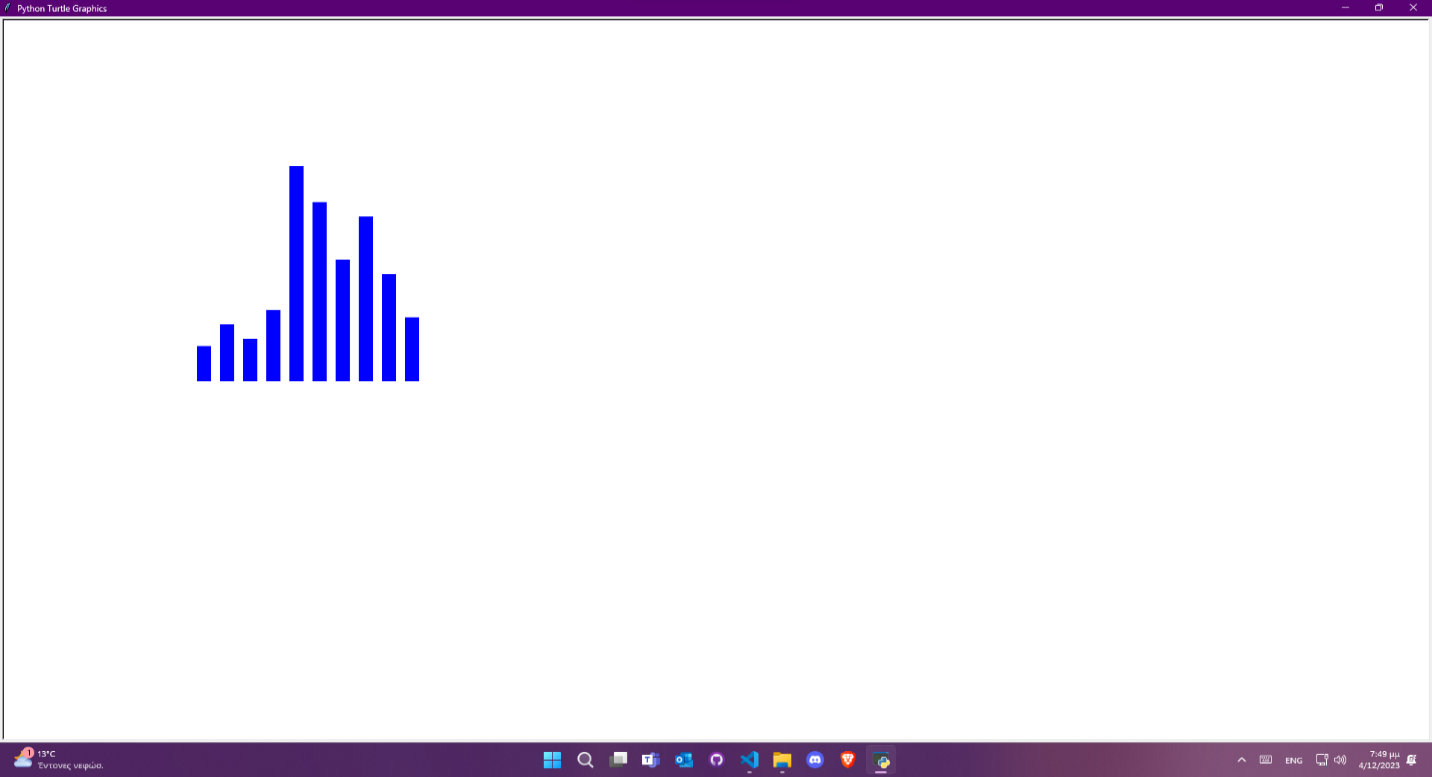


Τέλος βάλαμε μερικά κομμάτια που είναι για βελτιοποίηση και ωραιότερη παρουσίαση του μενού όπως η συνάρτηση clear\_terminal που καθαρίζει το terminal μετά από κάθε εντολή και το sleep που καθυστερεί τον κώδικα ώστε να προλαβαίνει ο χρήστης να διαβάσει τι γίνεται πρωτού κληθεί η επόμενη εντολή.

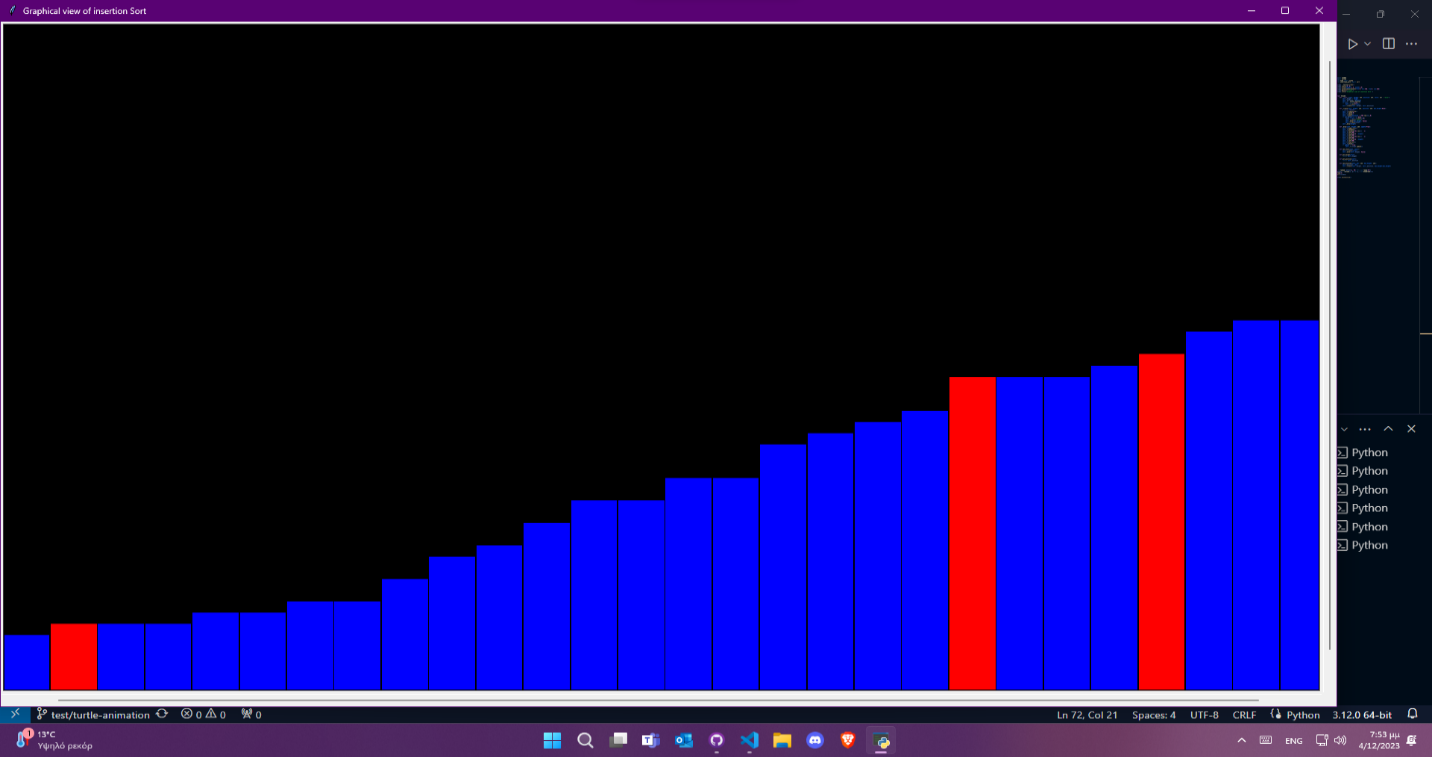
Για τη δημιουργία του αλγορίθμου ταξινόμησης ρωτήσαμε από το ChatGPT να μας εξηγήσει τον αλγόριθμο πως λειτουργεί, συμβουλευτήκαμε βίντεο στο YouTube καθώς και το visualgo.net το οποίο παρουσιάζει τον αλγόριθμο με απλά γραφικά βήματα. Η δομή του αλγορύθμου δεν είναι ιδιαίτερα περίπλοκη, ωστόσο το κύριο πρόβλημα που αντιμετωπίσαμε ήταν ότι δεν έπρεπε το στοιχείο που ελέγχαμε να μετέφερε κατευθείαν στην θέση που του αντιστοιχούσε. Κανονικά με βάση τον αλγόριθμο πρέπει να αλλάζει θέση με το αμέσως προηγούμενο μέχρι να βρει τη σωστή θέση, όπου,δηλαδή, το προηγούμενο στοιχείο της λίστας είναι μικρότερο από αυτό που ελέγχουμε.

Αυτό το καταφέραμε εύκολα με την εισαγωγή ενός loop μέσα στο οποίο τα στοιχεία j και j-1 αλλ΄ζουν θέσεις, αν το στοιχείο j-1 είναι μεγαλύτερο από το j. Αυτό γίνεται μέχρι το j > j-1 οπού το στοιχείο φτάνει στην τελική του σωστή θέση.

Το αρχείο turtle και ο αλγόριθμος που συνδέει ταξινόμηση και γραφικά ήταν οι μεγαλύτερες προκλήσεις που αντιμετωπίσαμε. Όσον αφορά το πρώτο είχαμε πολλές δυσκολίες στην πορεία μας. Χρησιμοποιήσαμε τη βοήθεια μιας κλάσης η οποία είναι υπεύθυνη για τη δημιουργία στηλών που αντιπροσωπεύουν τις τιμές της λίστας που θέλουμε να ταξινομήσουμε.

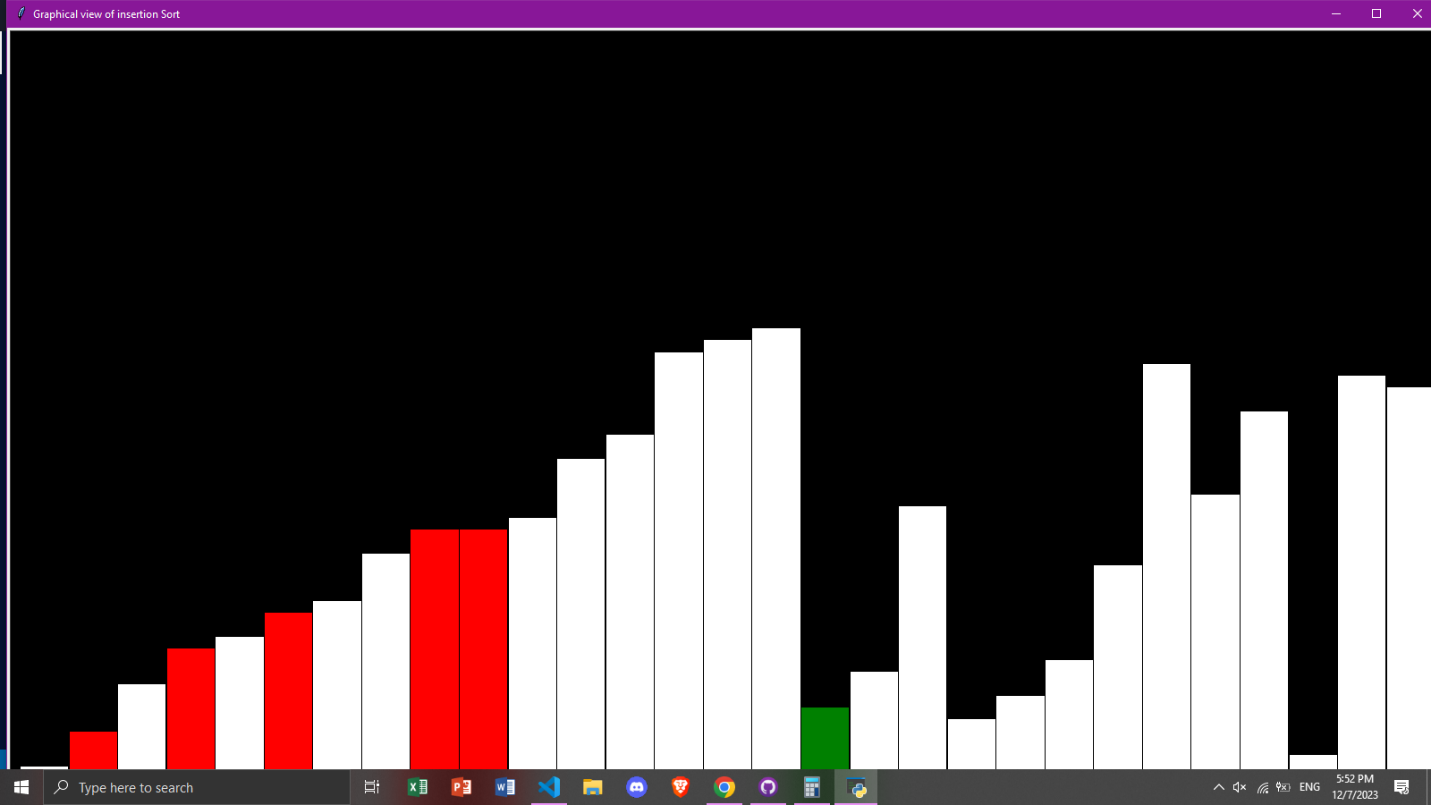


Στη συνέχεια μορφοποιήσαμε τις στήλες ώστε να ξεκινάνε από κάτω αριστερά και αυτόματα να ρυθμίζουν το πάχος τους ανάλογα το μήκος της λίστας ώστε όλες να χωράνε ακριβώς στην οθόνη του γραφικού περιβάλλοντος, με τη χρήση μιας μαθηματικής εξίσωσης που χρησιμοποιεί ως παράμετρο το μήκος της λίστας στη συνάρτηση δημιουργίας στηλών.

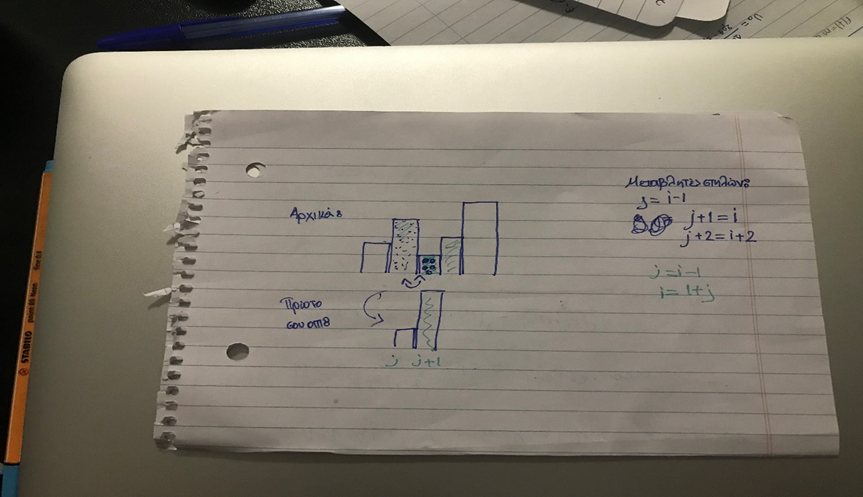


Το δεύτερο σκέλος των δυσκολιών μας επιλύθηκε πρώτων με την χρήση της συνάρτησης get height και μιας νέας μεταβλητής που μετατρέπει κάθε στοιχείο σε ύψος στηλης.Έτσι μπορέσαμε να έχουμε έναν κατάλληλο τρόπο σύγκρισης μιας κι μεταβήκαμε απ την αριθμιτική μορφή σε ένα πρόβλημα σύγκρισης υψών. Αυτό λύθηκε με την αξιοποίση συναρτήσεων απ το αρχείο turtle γεγονός που μας οδήγησε κι στην ολοκλήρωση της αλληλεπίδρασης τους

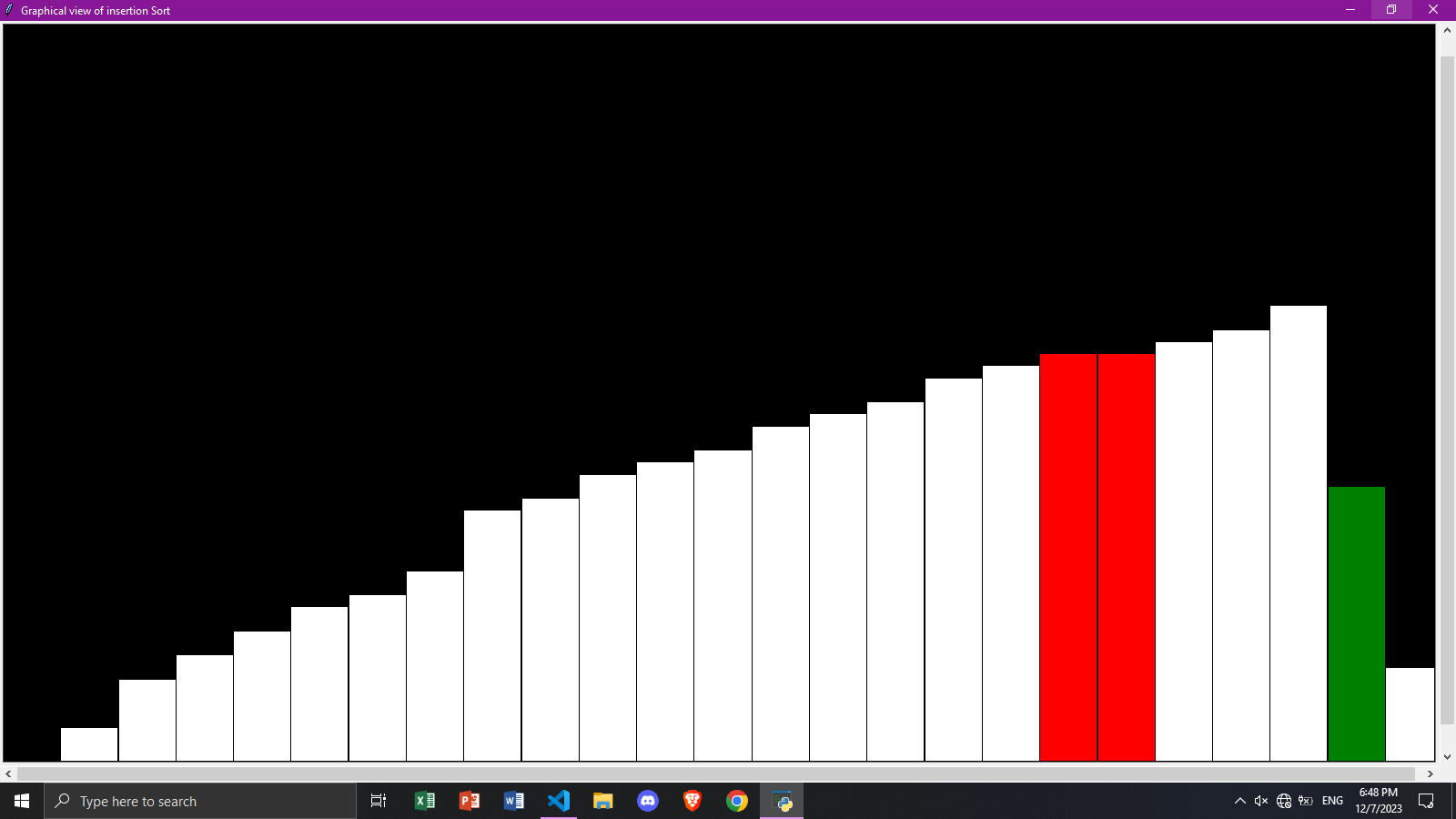
Έχοντας πλέον καταφέρει τη σύνδεση μεταξύ του κύριου αλγόριθμου ταξινόμησης και του γραφικού περιβάλλοντος θέλαμε να εντάξουμε χρώματα τα οποία είναι ως εξής: Πράσινη θα είναι η στήλη που το πρόγραμμα θα ταξινομήσει αμέσως μετά και με κόκκινο θα σημειώνει την «κίνηση» της στήλης που ταξινομείται εκείνη τη στιγμή, το οποίο ήταν και το κύριο πρόβλημά μας εκείνη τη στιγμή εξαιτίας κάποιας δυσλειτουργίας με την παράμετρο j που αναφέραμε πρωτύτερα.



Για την επίλυση αυτού του προβλήματος προσεγγίσαμε το ζήτημα με ένα διάγραμμα σε χαρτί όπου προσπαθήσαμε να δούμε πώς θα μετατοπίζονται οι δύο γειτονικές στήλες και πώς θα αλλάζουν τα χρώματά τους αφού γίνει η μετατόπιση. Έτσι αλλάζοντας τις παραμέτρους που είχαμε θέσει στον αλγόριθμο ταξινόμησης καταφέραμε πλέον το πρόγραμμα να αλλάζει ανά δύο τις στήλες και να τις ζωγραφίζει κόκκινες, μαρκάροντας έτσι την στήλη που κουνιέται μέχρι αυτή να καταλήξει στην θέση που πρέπει.



Έτσι μετά από την επίλυση αυτών των ζητημάτων το πρόγραμμά μας πήρε την εξής μορφή:



Έχωντας υλοποιήσει την κύρια πτυχή της εργασίας μας εντός του timeline αναζητήσαμε τρόπους εξέληξεις κι μορφοποίησης. Μία ιδέα ήταν η σταδιακή γέμιση των στηλών με χρώμα μετα το πέρας του αλγορύθμου, η οποία ενώ ξεκίνησε ως μία απλή μετατροπή αποδείχτηκε κάτι πιο ιδιαίτερο,καθώς το ‘sleep’ οδηγούσε στο πάγωμα όλου του αλγορύθμου. Λύση μας έδωσε, εν αγνοία του, ο κύριος Παλιουράς όταν μας μίλησε για το multiprocessing σε μια απ τις τελευταίες διαλέξεις του εξαμήνου.Έτσι αξιοποιώντας το threading μπορέσαμε να έχουμε μια ομαλή ροή και να κλείσουμε γενικώς με τον κώδικα που αφορά το project μας

