Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Πολυτεχνείο Κρήτης
Ακαδημαϊκό Έτος 2023-2024 (Χειμερινό Εξάμηνο)
Μάθημα: Εργαλεία Ανάπτυξης Λογισμικού & Προγραμματισμός Συστημάτων (ΠΛΗ 211)

Διδάσκων: Νίκος Γιατράκος

Εργασία 1 (20% της συνολικής βαθμολογίας του μαθήματος) Ομάδες 1 έως και 3 ατόμων

Θέμα: Ανάπτυξη Λογισμικού Συγκεντρωτικού Αναλυτή Αρχείου Ειδήσεων (Aggregative News Analyzer)

Σκοπός Εργασίας

Σκοπός της εργασίας είναι η εξοικείωσή σας με τα εργαλεία και τη διαδικασία ανάπτυξης λογισμικού σε Python και η συνεργασία σας σε ομάδες προγραμματιστών. Προκειμένου να επιτευχθεί αυτό, θα αναπτύξετε σε ομάδες, μικρής κλίμακας λογισμικό που θα αφορά συγκεντρωτική (aggregative) ανάλυση αρχείου ειδήσεων, με τη χρήση της γλώσσας Python και των εργαλείων Logging, Profiling, Refactoring και Unit Testing που εξετάζουμε στο μάθημα.

Στα πλαίσια της εργασίας δεν μπορείτε να υποθέσετε ότι τα news και άρα τα αρχεία που σας δίδονται δεν αλλάζουν ποτέ. Το αρχείο με τα news θεωρείται μεταβλητό καθώς σε ένα πραγματικό σενάριο τα news ανανεώνονται σε τακτά χρονικά διαστήματα. Άρα δεν μπορείτε να προ-υπολογίσετε το Jaccard Index και όσα άλλα ζητούνται μόνο μια φορά και να τα σώσετε μια μόνο φορά σε αρχείο.

Στα προγράμματα που θα αναπτύξετε επιτρέπεται ΜΟΝΟ η χρήση απλής Python. Δεν επιτρέπεται (και δεν βαθμολογούνται εργασίες με) χρήση βιβλιοθηκών και πακέτων όπως Pandas, Scikit Learn κλπ που δεν αποτελούν αντικείμενο αυτού του μαθήματος.

Σενάριο Εφαρμογής και Περιγραφή Συνόλου Δεδομένων (Dataset)

Για τους σκοπούς της εργασίας θα χρησιμοποιήσουμε το Reuters dataset το οποίο είναι διαθέσιμο στο φάκελο «Έγγραφα» του ecalss του μαθήματος.

Το dataset περιλαμβάνει ένα σύνολο αρχείων με πληροφορίες για **κείμενα ειδήσεων (documents)** που έχουν ανατεθεί σε συγκεκριμένες **κατηγορίες (categories)** το καθένα. Κάθε document επίσης περιλαμβάνει ένα σύνολο από **όρους (terms)** τους οποίους ενδιαφερόμαστε να αναλύσουμε σε συνδυασμό με τις προαναφερθείσες categories.

Πιο συγκεκριμένα ο στόχος της ανάλυσής μας είναι να συνθέσουμε την πληροφορία που υπάρχει στα διάφορα αρχεία, προκειμένου να υπολογίσουμε το βαθμό σχετικότητας κάθε term με κάθε category. Για να ποσοτικοποιήσουμε το βαθμό σχετικότητας, θα χρησιμοποιήσουμε το Jaccard Index score.

Για κάθε term T και category C, ο Jaccard Index δίνεται από την ακόλουθη φόρμουλα:

$$J(T, C) = \frac{|DOC(T) \cap DOC(C)|}{|DOC(T) \cup DOC(C)|}$$

όπου :

- DOC(T): το σύνολο των documents όπου εμφανίζεται ο term T και |DOC(T)| η πληθικότητα (αριθμός στοιχείων -- cardinality) αυτού του συνόλου.
- DOC(C): το σύνολο των documents που ανήκουν στην category C και |DOC(C)| η πληθικότητα αυτού του συνόλου.
- DOC(T) ∩ DOC(C): το σύνολο των documents όπου εμφανίζεται ο term T ΚΑΙ ανήκουν στο category C. Άρα, με |DOC(T) ∩ DOC(C)| εκφράζουμε τον αριθμό των documents που περιέχουν τον term T και ανήκουν στο category C.
- DOC(T) ∪ DOC(C): το σύνολο των documents όπου είτε περιέχουν τον term T, είτε το document ανήκει στο category C, ή και τα δύο. Αντίστοιχα, με |DOC(T) ∪ DOC(C)| εκφράζουμε τον αριθμό τέτοιων κειμένων. Σημειώνεται ότι:

$$|DOC(T) \cup DOC(C)| = |DOC(T)| + |DOC(C)| - |DOC(T) \cap DOC(C)|$$

Ισχύει ότι $0 \leq J(T, C) \leq 1$, δλδ τιμές του Jaccard Index κοντά στο 1 εκφράζουν μεγαλύτερο βαθμό συσχέτισης του term T με το category C.

Από τα αρχεία που σας δίνονται, έχουμε το **ASCII file rcv1-v2.topics.qrels.txt** 35,382,548 bytes unzipped. Το αρχείο αυτό περιγράφει ποια categories έχουν ανατεθεί σε κάθε document. Κάθε ζεύγος <category, document> αναπαρίσταται από μια εγγραφή, και υπάρχουν 2,606,875 εγγραφές στο αρχείο κάθε μια από τις οποίες έχει την εξής μορφή:

<category name> <documentid> 1 (το «1» δεν ενδιαφέρει την ανάλυσή μας)
Για παράδειγμα, έχουμε τις εξής εγγραφές:

E11 2286 1

ECAT 2286 1

MCAT 2286 1

C24 2287 1

CCAT 2287 1

Οι εγγραφές αυτές λένε ότι το document με documentid = 2286 ανήκει στις κατηγορίες E11, ECAT, MCAT και το document με documentid = 2287 ανήκει στις κατηγορίες C24 και CCAT.

Σας δίνονται επίσης 5 ASCII αρχεία με ονόματα της μορφής lyrl2004_vectors_test_pt*.dat.gz. Τα αρχεία αυτά περιέχουν συνολικά 804.414 εγγραφές, τα οποία σε gzipped format έχουν τα παρακάτω μεγέθη (bytes):

lyrl2004_vectors_test_pt0.dat.gz : 159879168

lyrl2004_vectors_test_pt1.dat.gz : 161878016

lyrl2004_vectors_test_pt2.dat.gz : 158580736

lyrl2004_vectors_test_pt3.dat.gz : 149512192

lyrl2004_vectors_train.dat.gz : 18620416

Αν θέλετε, μπορείτε να χρησιμοποιήσετε ένα υποσύνολο από αυτά τα αρχεία αρκεί τα data που θα χρησιμοποιήσετε να είναι αρκετά τόσο για να έχει νόημα η συγκεντρωτική ανάλυση που θέλουμε να κάνουμε, όσο και για να μπορείτε να κάνετε καλό profiling, refactoring, unit testing.

Κάθε εγγραφή σε αυτά τα αρχεία αφορά ένα συγκεκριμένο document και περιλαμβάνει την λίστα από terms που περιέχει το document στην ακόλουθη μορφή:
<did> [<tid>:<weight>]+ weight

Όπου:
<did> : Το document id όπως έχει ανατεθεί από το πρακτορείο Reuters
<tid> : Το term id που προσδιορίζει μοναδικά κάθε term. Τα term ids είναι μεταξύ 1 και 47.236
<weight>: ΔΕ θα χρησιμοποιήσουμε αυτή την τιμή για την ανάλυσή μας.

Για παράδειγμα μια εγγραφή

2286 864:0.0497399253756197 1523:0.044664135988103 1681:0.0673871572152868 ….
Λέει ότι το document με did = 2286 περιλαμβάνει τους όρους με term ids 864,1523,1681…

Τέλος σας δίνεται ένα ASCII αρχείο stem.termid.idf.map.txt, με μέγεθος 1.411.031 bytes. Κάθε εγγραφή αυτού του αρχείου αντιστοιχεί το term id με τον πραγματικό term/λέξη που περιγράφει. Υπάρχουν 47.236 εγγραφές στο αρχείο, μια για κάθε όρο. Οι εγγραφές είναι στην ακόλουθη μορφή:

<stem> <tid> <idf>

Όπου:

<stem> : η ρίζα (ετυμολογικά) του όρου
<tid> : term id που προσδιορίζει μοναδικά κάθε term
<idf>: ΔΕ θα το χρησιμοποιήσουμε στην ανάλυσή μας

Για παράδειγμα, οι εγγραφές:

map 25376 6.41892286389214

reduce 34542 2.83102755083624

αναφέρουν ότι το tid = 25376 αντιστοιχεί στον όρο «map» και το tid = 34542 αντιστοιχεί στον όρο «reduce».

Θέλουμε να φτιάξουμε ένα driver πρόγραμμα που θα δίνει στο χρήστη τη δυνατότητα να κάνει ερωτήσεις συγκεντρωτικής ανάλυσης του αρχείου των ειδήσεων. Για να γίνει αυτό, ο χρήστης θα πρέπει να μπορεί να δίνει από command line menu τις ακόλουθες εντολές:

Εντολή	Περιγραφή
@ <category> <k>	Δείξε μου τα <k> πιο σχετικά (βάσει Jaccard Index) stems (όχι tid) για την κατηγορία <category>
# <stem> <k>	Δείξε μου τα <k> πιο σχετικά (βάσει Jaccard Index) categories για το <stem>
$ <stem> <category>	Δείξε μου το Jaccard Index του συγκεκριμένου (stem, category) ζεύγους
*** <filename>.<filetype>**	Σώσε σε ένα αρχείο με όνομα <filename>.<filetype> ΟΛΑ τα (category, stem) ζεύγη μαζί με τον Jaccard Index τους στη μορφή <stem> <category> <Jaccard Index>. Το filename θα είναι αλφαριθμητικό που δίδεται από το χρήστη, ενώ το filetype που θα δώσει ο χρήστης θα μπορεί να είναι ένα εκ των {json, xlsx} οπότε και το αρχείο, όπου θα πρέπει να σώζονται τα αποτελέσματα, θα πρέπει να δημιουργηθεί κατάλληλα ως json ή excel αρχείο σύμφωνα με βάση όσα δείξαμε στην αντίστοιχη διάλεξη. Για παράδειγμα στο excel αρχείο θα έχουμε 3 στήλες με αντίστοιχα ονόματα, στο json θα έχουμε 3 πεδία με αντίστοιχα ονόματα και τιμές κοκ.
P <did> -c|-t	Αν στην εντολή χρησιμοποιείται το switch "-t" τότε δείξε μου όλα τα stem που περιέχει το document με κωδικό did. Αν στην εντολή χρησιμοποιείται το switch "-c" τότε δείξε μου όλες τις κατηγορίες που έχουν ανατεθεί στο document με κωδικό did.
C <did> -c|-t	Αν στην εντολή χρησιμοποιείται το switch "-t" μέτρησε και δείξε μου στην οθόνη πόσα (count) μοναδικά terms περιέχει το document με κωδικό did. Αν στην εντολή χρησιμοποιείται το switch "-c" τότε μέτρησε και δείξε μου στην οθόνη πόσες (count) κατηγορίες έχουν ανατεθεί στο document με κωδικό did.

Βοηθητικά/προτεινόμενα βήματα για τον υπολογισμό Jaccard Index όπου απαιτείται:

- Υπολογίζουμε |DOC(C)| για όλες ή για μια κατηγορία που μας ενδιαφέρει
- Υπολογίζουμε |DOC(T)| για όλους ή για έναν term που μας ενδιαφέρει
- Υπολογίζουμε $|DOC(T) \cap DOC(C)|$ για όλα ή για όποια ζεύγη μας ενδιαφέρει
- Αντιστοιχούμε κάθε tid με το stem στο οποίο ανήκει
- Υπολογίζουμε και εξάγουμε (τυπώνουμε ή σώζουμε ανάλογα με την εντολή του χρήστη) το <category name> <stem> <jaccard_index_score>

Παραδοτέα και Υποβολή

- **Ένας από την ομάδας σας**, θα κατεβάσει και θα αποθηκεύσει το αρχείο PLH211_Project1_2023_2024.ipynb στον υπολογιστή του από την ενότητα Εργασίες του eclass του μαθήματος: https://www.eclass.tuc.gr/courses/HMMY314/
- **Πρώτη του δουλειά** μετά από αυτό θα είναι να ανεβάσει το PLH211_Project1_2023_2024.ipynb στο δικό του Colab και ***ΝΑ ΤΟ ΜΕΤΟΝΟΜΑΣΕΙ ΟΠΩΣΔΗΠΟΤΕ*** με τους Αριθμούς Μητρώου των μελών της ομάδας σας χωρισμένους με underscore. Αν για παράδειγμα η ομάδα σας έχει το Φοιτητή 1 με ΑΜ:11111, το Φοιτητή 2 με ΑΜ:22222 και το Φοιτητή 3 με ΑΜ:33333, το αρχείο σας θα πρέπει να μετονομαστεί σε 11111_22222_33333.ipynb
- Στη συνέχεια, το μέλος της ομάδας θα κάνει share από το Google Drive του (εκεί αποθηκεύονται έτσι κι αλλιώς by default τα Colab Notebooks) με τα υπόλοιπα μέλη της ομάδας σας. Κάθε μέλος μπορεί να δημιουργήσει copy του .ipynb αρχείου αυτού αν θέλει.
- Αφού συνεννοηθείτε μεταξύ σας, χωρίς διαμεσολάβηση του διδάσκοντα, θα σχηματίσετε ομάδες. Έπειτα, θα πρέπει να εγγραφείτε στις κενές ομάδες που έχω δημιουργήσει στην αντίστοιχη περιοχή του eclass (https://www.eclass.tuc.gr/modules/group/index.php?course=HMMY314&urlview=1).
  Έως την 15/11/2023 θα μπορείτε να εγγραφείτε και να απεγγραφείτε σε ομάδες. Έπειτα από την 15/11/2023 οι ομάδες θα κλείσουν και θα ΠΡΕΠΕΙ να παραμείνουν έτσι και για τα 2 project του μαθήματος. ΜΗΝ εγγραφείτε σε τυχαίες ομάδες και ΜΗΝ εγγραφείτε σε ομάδα αν δεν έχετε πρώτα επικοινωνήσει και συμφωνήσει με τα υπάρχοντα μέλη της. Αν δε βρίσκετε ομάδα, μπορείτε να εκπονήσετε την εργασία μόνοι σας.
- Όταν τελειώσετε την εργασία σας, κάθε ομάδα θα ανεβάσει ΤΟ ΕΝΑ ΚΑΙ ΤΕΛΙΚΟ .ipynb αρχείο στο eclass. Αυτό είναι το παραδοτέο σας. Όλος ο κώδικας και η τεκμηρίωση της εργασίας σας θα πρέπει να είναι σε code cells και text cells αυτού του .ipynb αρχείου. Το .ipynb αρχείο είναι δομημένο ώστε να λειτουργεί ως template για τη δομή των απαντήσεών σας. Ακολουθήστε αυτό το template υποχρεωτικά. Η τεκμηρίωση του κώδικά σας στα text cells όπου ζητείται, βαθμολογείται.
- Στο Colab θα είστε έτοιμοι να δουλέψετε όλοι μαζί επί της εργασίας σας, αλλά προσοχή, όχι ταυτόχρονα. Το Colab δεν είναι Git ή Github (θα τα γνωρίσουμε στη συνέχεια στο μάθημα). Δεν μπορείτε να δουλεύετε όλοι μαζί ταυτόχρονα στο ίδιο .ipynb αρχείο (θα σώζει τις αλλαγές ενός, στους άλλους θα δίνει warnings ότι κάποιος άλλος άλλαξε το αρχείο).

Υλοποίηση

Το πρόγραμμά σας θα ξεκινάει και θα έχει υποχρεωτικά μια main μέθοδο. Στη main, μέσα σε ένα while True loop θα υπάρχει ένα μήνυμα που θα τυπώνει τον πίνακα - menu με τις πιθανές εντολές που μπορεί να δώσει ο χρήστης.

Όταν ο χρήστης πληκτρολογήσει μια εντολή θα εκτελείται αντίστοιχη μέθοδος στον κώδικα που θα υλοποιεί την αντίστοιχη εντολή. Για παράδειγμα για την εντολη «s <filename>.<filetype>» που φαίνεται στον παραπάνω πίνακα, θα υπάρχει μέθοδος που θα

φροντίζει να εκτελείται η λειτουργικότητα που απαιτείται.

Η διαδικασία ανάπτυξης λογισμικού περιλαμβάνει τις εξής φάσεις:

- **Development Phase** (30% του βαθμού της παρούσας εργασίας): Σε 1 code cell όλος ο κώδικας του προγράμματος σας, όταν τελειώσετε την πρώτη φάση της ανάπτυξής του, αλλά πριν το logging, profiling, refactoring και unit testing.
- **Logging Phase** (10% του βαθμού της παρούσας εργασίας**).** Κατά τη χρήση του Αναλυτή Ειδήσεων σας, θα πρέπει να καταγράφονται σε ένα αρχείο logme.txt οι ενέργειες/εντολές που χρησιμοποίησε μόνο ο τελευταίος χρήστης που το έτρεξε. Θέλουμε να logάρουμε INFO, κατά βάση, μέσω κατάλληλου conf αρχείου στη μορφή (formatter) `format=%(asctime)s - %(name)s - %(levelname)s - %(message)s`
  Σε ένα code cell θα βάλετε τα περιεχόμενα του .conf αρχείου (μπορείτε να το κάνετε και με .yaml αν προτιμάτε). Σε 1 άλλο code cell θα βάλετε τον κώδικα του προγράμματος σας, όταν τελειώσετε το Logging Phase, αλλά πριν το profiling, refactoring και unit testing.
- **Profiling Phase** (20% του βαθμού της παρούσας εργασίας): Σε 1 ή περισσότερα code cell θα βάλετε το version του κώδικα στο οποίο βάλατε εντολές για το profiling του προγράμματος σας, πριν το refactoring και το unit testing. Στην τεκμηρίωσή σας θα χρησιμοποιήσετε τα αποτελέσματα του profiling για να επιχειρηματολογήσετε για το ποια είναι τα major memory ή/και computational bottlenecks στον κώδικά σας από τα οποία και θα προσπαθήσετε να απαλλαγείτε στο Refactoring Phase που ακολουθεί. Αφού βρείτε τις μαύρες τρύπες στον κώδικά σας (αναφέροντας ακριβή νούμερα για computational time και memory utilization), θα πρέπει να πειραματιστείτε με πιθανές αλλαγές στον κώδικα και profiling αυτών των αλλαγών.
- **Refactoring Phase** (20% του βαθμού της παρούσας εργασίας): Σε 1 code cell όλο τον τελικό κώδικα του προγράμματος σας μετά το refactoring αλλά πριν το unit testing.
  [*Υπόδειξη:* *Πέραν της Idiomatic Python που είδαμε στο μάθημα στην ενότητα του refactoring, στο Refactoring Phase εξετάστε και βελτιστοποιήστε τον τρόπο με τον οποίο χειρίζεστε τα δεδομένα σας. Κάποιες από τις αποφάσεις που θα πρέπει να πάρετε:*
  - *Θα υπολογίζονται όλοι οι Jaccard Index με την εκκίνηση του προγράμματος και έπειτα θα δίδεται η δυνατότητα ερώτησης από το χρήστη; Αυτό θα καθυστερεί την εκκίνηση του προγράμματος. Θα υπολογίζεται ο Jaccard Index μόνο για ό,τι ζητάει ο χρήστης χωρίς κάποιο προϋπολογισμό; Αυτό θα καθυστερεί την εκτέλεση των εντολών του χρήστη. Μήπως συμφέρει να κρατάμε σε ένα αρχείο μερικά ζεύγη από categories και stems για τα οποία ο χρήστης ρωτάει συχνά; Αν ναι, πως διατηρούμε εκείνο το αρχείο ώστε να ενημερώνεται με βάση τη συχνότητα των ερωτήσεων για συγκεκριμένα stem, categories κλπ;*
  - *Θα φορτώνονται τα δεδομένα σε dictionary; Συμφέρει να μετατραπούν σε lists ή μήπως tuples; Lists of tuples; Tuples of lists; Άλλα Nested collections;*
  - *Συμφέρει να χρησιμοποιούμε comprehensions; Να χρησιμοποιούμε generators;*
  - *Όλα τα παραπάνω αποτελούν θέματα που πρέπει να εξετάσετε στο refactoring (και σε loops με το profiling phase) ώστε να βρείτε καλά memory utilization vs computational time trade-offs για τον κώδικά σας.]*
- **Unit Testing Phase** (20% του βαθμού της παρούσας εργασίας): Σε 1 code cell θα βάλετε όλον τον test κώδικα του unit testing και σε 1 code cell όλο τον κώδικα του προγράμματος σας μετά το unit testing (π.χ. επειδή το testing έδειξε πράγματα που έπρεπε να διορθωθούν).

Ο κώδικάς σας αρκεί να είναι καλά δομημένος, δε χρειάζεται να είναι απαραίτητα αντικειμενοστρεφής.

Τρόπος Βαθμολόγησης και Κώδικας Δεοντολογίας

Η εργασία και το μάθημα αφορούν την ανάπτυξη λογισμικού. Η έμφαση στη βαθμολόγηση, λοιπόν, είναι σε σωστό (Development, Logging και Unit Testing Phase) και αποδοτικό (Logging, Profiling και Refactoring) κώδικα. Ως εκ τούτου:

Ο κώδικας των ομάδων ελέγχεται με λογισμικού εντοπισμού αντιγραφών. Ομάδες που φαίνεται να έχουν αντιγράψει οποιοδήποτε μέρος των φάσεων και του κώδικα της εργασίας μηδενίζονται (όλα τα μέλη και των δύο ομάδων - ανεξαρτήτως ποιος αντέγραψε από ποιόν).

Κώδικας που δεν κάνει compile, δίνει runtime errors ή δεν είναι σωστός ως προς τη λειτουργικότητα που ζητείται να αποδοθεί, βαθμολογείται με έως μηδενικό βαθμό ακόμη και αν η τεκμηρίωση και ο σχολιασμός του είναι ακέραια.

Όλες οι άλλες περιπτώσεις (ελλειπής αλλά σωστή λειτουργικότητα, ελλειπής αλλά καταληπτή τεκμηρίωση κλπ) που δεν εμπίπτουν παραπάνω, αξιολογούνται ξεχωριστά ανάλογα με την περίπτωση.

Στο .ipynb που θα παραδώσετε θα αναφέρεται **ακριβώς** ποια μέλη της ομάδας δούλεψαν σε κάθε φάση (ζητείται μαζί με τα ΑΜ και τα ονόματα των φοιτητών στο .ipynb). Ο στόχος της εργασίας και ο λόγος που η ομάδα έχει περισσότερα μέλη είναι να δουλέψετε συνεργατικά, αλλά ανεξάρτητα στις φάσεις ανάπτυξης. **ΔΕΝ θα πρέπει να δουλέψετε όλοι σε όλα, αλλά συγκεκριμένα μέλη της ομάδας θα αναλάβουν κάθε φάση.** Για παράδειγμα, σε μια ομάδα 3 ατόμων, η διαδικασία ανάπτυξης του λογισμικού σας θα μπορεί να γίνει, ενδεικτικά, κάπως έτσι:

- 1 μέλος της ομάδας θα αναλάβει το development και το logging phase, θα συμπληρώσει όσα ζητούνται σε αυτές τις φάσεις στο .ipynb αρχείο και θα το παραδώσει στα άλλα μέλη της ομάδας. Ο κώδικας θα πρέπει να είναι (φυσικά) σωστός, δομημένος και ευανάγνωστος.
- 1 άλλο μέλος της ομάδας θα πάρει τον κώδικα των προηγουμένων φάσεων και θα κάνει το profiling εναλλακτικές υλοποιήσεις για να βρει τα bottlenecks και τα memory utilization vs execution time trade offs. Θα συμπληρώσει όσα ζητούνται σε αυτή τη φάση στο .ipynb αρχείο και θα το παραδώσει στα άλλα μέλη της ομάδας καταγράφοντας τα συμπεράσματα του. Τι αργεί; Τι είναι πιο γρήγορο; Τι τρώει πολλή μνήμη; Τι μπορεί να αλλάξω για να κάνω τον κώδικα πιο γρήγορο και να καταναλώνει λιγότερη μνήμη; Ποιες οι προτάσεις βελτίωσης;
- 1 άλλο μέλος της ομάδας θα πάρει τον κώδικα των προηγούμενων φάσεων και θα κάνει το refactoring και το τελικό unit testing των μεθόδων. Θα συμπληρώσει όσα ζητούνται σε αυτή τη φάση στο .ipynb αρχείο και θα το παραδώσει σε όλα τα μέλη της ομάδας.
- Όλη η ομάδα ελέγχει και σχολιάζει, βελτιώνει την τελική εργασία. Ακολουθεί η τελική υποβολή.

Κατά την προφορική εξέταση κάθε φοιτητής θα ερωτηθεί επί όσων ανέφερε ότι υλοποίησε ο ίδιος στην εργασία. Μειωμένη συμμετοχή στην ανάπτυξη της εργασίας (π.χ. κάποιος που έκανε μόνο το logging) ή αδυναμία απάντησης στις προφορικές ερωτήσεις επηρεάζει αρνητικά τη βαθμολόγηση.

Διαδικαστικά

- **Αριθμός Μελών Ομάδας Εργασίας**: 1 **έως** και 3 άτομα. Οι ομάδες ΔΕ θα πρέπει να αλλάξουν ανάμεσα στην πρώτη και τη δεύτερη εργασία.
- **Ημερομηνία Παράδοσης Εργασίας**: έως τα μεσάνυχτα της **01/12/2023**. Η εργασία ΔΕΝ πρόκειται να πάρει παράταση. Την 01/12/2023 θα ανακοινωθεί η δεύτερη εργασία.
- **Τρόπος Παράδοσης**: Ομαδική υποβολή μέσω eclass σε ένα .ipynb αρχείο με κατάλληλο όνομα όπως περιγράφεται παραπάνω.
- **Ημερομηνία Προφορικής Εξέτασης**: Κατόπιν ανακοίνωσης, θα εξεταστείτε ΚΑΙ επί των 2 εργασιών που υλοποιήσατε την τελευταία εβδομάδα μαθημάτων. Για το λόγο αυτό οι ομάδες ΔΕ μπορούν να αλλάξουν από την 1η στη 2η εργασία.
- Η εργασία ισχύει για την εξεταστική Ιανουαρίου και Σεπτεμβρίου (δηλαδή δεν υπάρχει επαν-υποβολή – βελτίωση για το Σεπτέμβρη). Δεν κρατείται βαθμός εργασιών για το επόμενο ακαδημαϊκό έτος.

Ερωτήσεις/Απορίες Σχετικά με την Εργασία

Για τις ερωτήσεις - απορίες σας μπορείτε

- να χρησιμοποιείτε την αντίστοιχη κατηγορία στην περιοχή συζητήσεων του μαθήματος στο eclass: https://www.eclass.tuc.gr/modules/forum/viewforum.php?course=HMMY314&forum=47335
- να απευθύνεστε στο διδάσκοντα μέσω email:
    - Θέμα Email : ProjectPLH211 – ΟΝΟΜΑΤΕΠΩΝΥΜΟ & ΑΜ φοιτητή
    - Παραλήπτης: **ngiatrakos@ tuc.gr**

Καλή Δουλειά!