Ε.Α.Π./ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

4η ΓΡΑΠΤΗ ΕΡΓΑΣΙΑ

ΑΚΑΔΗΜΑΪΚΟΥ ΕΤΟΥΣ 2021-2022

16/03/2022

Ημερομηνία παράδοσης εργασίας: Κυριακή 08/05/2022
Καταληκτική ημερομηνία παραλαβής: Τετάρτη [1] 11/05/2022
Ημερομηνία ανάρτησης ενδεικτικών λύσεων: Σάββατο 14/05/2022
Καταληκτική ημερομηνία αποστολής σχολίων στον φοιτητή: Κυριακή 29/05/2022

ΥΠΟΕΡΓΑΣΙΑ 1. (βαθμοί 25)
Διαχείριση Αρχείων – Σειριοποίηση αντικειμένων με την pickle

ΥΠΟΕΡΓΑΣΙΑ 2. (βαθμοί 25)
Βιβλιοθήκες pandas και matplotlib.pyplot

ΥΠΟΕΡΓΑΣΙΑ 3. (βαθμοί 25)
Βιβλιοθήκη sqlite3

ΥΠΟΕΡΓΑΣΙΑ 4. (βαθμοί 25)
Βιβλιοθήκη NumPy

ΣΥΝΟΛΟ **(βαθμοί 100)**

---

[1] Σύμφωνα με τον Κανονισμό Σπουδών, η καταληκτική ημερομηνία για την παραλαβή της Γ.Ε. από το μέλος ΣΕΠ είναι η επόμενη Τετάρτη από το τέλος της εβδομάδας παράδοσης Γ.Ε.

ΥΠΟΕΡΓΑΣΙΑ 1. (βαθμοί 25)

Στην 2η ΟΣΣ έγινε παρουσίαση μιας μικρής εφαρμογής που έκανε απλή διαχείριση τηλεφωνικών επαφών (με χρήση λεξικού). Η εφαρμογή αυτή ζητούσε από τον χρήστη να προσθέσει επαφές στο λεξικό και του επέτρεπε να δει ταξινομημένες τις επαφές. Όμως, κάθε φορά που ξεκινούσε η εφαρμογή έπρεπε να εισάγουμε τις επαφές από την αρχή. Σε αυτή την υποεργασία ζητείται να επεκταθεί η εφαρμογή αυτή, ώστε να αποθηκεύει τις επαφές σε ένα αρχείο με χρήση του αρθρώματος pickle όταν τερματίζει η εφαρμογή, και κατά την έναρξή της να διαβάζει το λεξικό με τις επαφές από το αρχείο αυτό. Επίσης, εφαρμόζουμε το αντικειμενοστρεφές μοντέλο προγραμματισμού. Ειδικότερα:

Η νέα εφαρμογή περιλαμβάνει δύο κλάσεις που θα βρείτε στο αρχείο **1_code_template.py**: την κλάση **Store** και την κλάση **AddressBook**. Η πρώτη περιλαμβάνει δύο μεθόδους: τη μέθοδο ***load***, η οποία διαβάζει το αρχείο pickle το οποίο περιέχει δυαδικά (binary) δεδομένα και επιστρέφει το λεξικό προς παρουσίαση και επεξεργασία από τον χρήστη, και τη μέθοδο ***save***, η οποία αποθηκεύει το λεξικό με τις επαφές σε αρχείο pickle. Η δεύτερη περιλαμβάνει τις μεθόδους ***contactsManagement***, που είναι ελαφρά τροποποιημένος ο κώδικας που είδαμε στην ΟΣΣ2, και τη μέθοδο ***showContacts***, που τυπώνει τη λίστα των επαφών.

α) Να υλοποιήσετε την μέθοδο-κατασκευαστή της κλάσης AddressBook. Η μέθοδος αυτή δημιουργεί αρχικά ένα αντικείμενο τύπου Store και στη συνέχεια καλεί τη μέθοδο *load* του αντικειμένου αυτού, εκχωρώντας στη μεταβλητή self.epafes τα δεδομένα που επιστρέφει. Στη συνέχεια καλείται η μέθοδος *contactsManagement*, η οποία δίνεται και υλοποιεί την αλληλεπίδραση με τον χρήστη. Τέλος, όταν ολοκληρωθεί η αλληλεπίδραση, καλείται η μέθοδος *save* του αντικειμένου τύπου Store.

β) Να ορίσετε τη μέθοδο *load* της κλάσης Store, η οποία επιστρέφει το λεξικό που διαβάζει από το αρχείο pickle. Η συνάρτηση ανοίγει το αρχείο για ανάγνωση δυαδικών (binary) δεδομένων και διαβάζει το λεξικό με τη μέθοδο *load* της pickle. Η συνάρτηση εμφανίζει μήνυμα και επιστρέφει άδειο λεξικό σε περίπτωση σφάλματος / αποτυχίας ανάγνωσης.

γ) Να ορίσετε τη μέθοδο *save* της κλάσης Store, η οποία δέχεται ως παράμετρο το λεξικό με τις επαφές και αποθηκεύει το λεξικό στο αρχείο. Η συνάρτηση ανοίγει το αρχείο για εγγραφή δυαδικών (binary) δεδομένων και γράφει το λεξικό στο αρχείο με τη μέθοδο *dump* της pickle. Η συνάρτηση εμφανίζει μήνυμα σε περίπτωση σφάλματος / αποτυχίας εγγραφής.

Υπόδειξη: Να χρησιμοποιηθεί το αρχείο **1_code_template.py** (που περιέχει και την εφαρμογή απλής διαχείρισης τηλεφωνικών επαφών) ως οδηγός επίλυσης.

ΥΠΟΕΡΓΑΣΙΑ 2. (βαθμοί 25)

Στο αρχείο excel *weatherdata.xslx*, δίνονται βασικά μετεωρολογικά δεδομένα από τρεις πόλεις (Αθήνα, Θεσσαλονίκη, Πάτρα). Το αρχείο αποτελείται από τρία φύλλα εργασίας και κάθε φύλλο εργασίας περιέχει μετεωρολογικά δεδομένα σχετικά με τη θερμοκρασία, την ταχύτητα του ανέμου, το επίπεδο υγρασίας και την ορατότητα της αντίστοιχης πόλης, ανά μήνα. Ανοίξτε το αρχείο για να δείτε την οργάνωση των δεδομένων και τους τίτλους των στηλών του.

Ζητείται:

α) Να ορίσετε τη συνάρτηση *printMeteoData(file,city)*, με παραμέτρους το όνομα του αρχείου με τα δεδομένα και το όνομα μιας πόλης. Η συνάρτηση διαβάζει και αποθηκεύει σε ένα dataframe τα δεδομένα του αρχείου από το φύλλο εργασίας για την πόλη, με χρήση της συνάρτησης *read_excel* της βιβλιοθήκης *pandas,* και στη συνέχεια εκτυπώνει το dataframe αυτό.

β) Να ορίσετε τη συνάρτηση *printMeteoStats(file,metric,city)*, με παραμέτρους το όνομα του αρχείου με τα δεδομένα, το όνομα της μετρικής (στήλης του φύλλου) και το όνομα μιας πόλης. Η συνάρτηση διαβάζει, με αντίστοιχο τρόπο, τα δεδομένα για την πόλη, και υπολογίζει και εμφανίζει τη μέση τιμή, την ελάχιστη τιμή και τη μέγιστη τιμή των στοιχείων για τη συγκεκριμένη μετρική (στήλη).

γ) Να ορίσετε τη συνάρτηση *plotMonthlyData(file,x_axis,metric,*list_of_cities)*, με παραμέτρους το όνομα του αρχείου με τα δεδομένα, το όνομα της στήλης του φύλλου για τον οριζόντιο άξονα, το όνομα της μετρικής / στήλης του φύλλου) και τα ονόματα των πόλεων (μεταβλητό πλήθος παραμέτρων). Η συνάρτηση διαβάζει, για κάθε πόλη, τα δεδομένα της πόλης και, με χρήση της συνάρτησης *plot* της matplotlib.pyplot, σχεδιάζει μια (συγκριτική) γραφική παράσταση με τα στοιχεία υγρασίας των τριών πόλεων ανά μήνα. Προαιρετικά, η συνάρτηση ορίζει τίτλο του γραφήματος, ετικέτες των αξόνων και υπόμνημα.

δ) Να γράψετε κυρίως πρόγραμμα που ορίζει το όνομα του αρχείου των δεδομένων, καλεί την *printMeteoData* για την εμφάνιση όλων των δεδομένων της Αθήνας, καλεί την *printMeteoStats* για την εμφάνιση των στατιστικών για τη θερμοκρασία της Θεσσαλονίκης και, τέλος, καλεί την *plotMeteoData* που εμφανίζει συγκριτικό διάγραμμα για την υγρασία των τριών πόλεων.

Υπόδειξη: Να χρησιμοποιηθεί το αρχείο **2_code_template.py** ως οδηγός επίλυσης.

Σημείωση: *Βεβαιωθείτε ότι στο σύστημά σας υπάρχει το πρόγραμμα pip και στην συνέχεια κάνετε εγκατάσταση των βιβλιοθηκών: pandas, openpyxl, matplotlib* (pip3 σε MacOS, Linux)

ΥΠΟΕΡΓΑΣΙΑ 3. (βαθμοί 25)

Μια σχολική μονάδα αποφασίζει να δημιουργήσει μια εφαρμογή για τη δανειστική βιβλιοθήκη της. Με τη χρήση της sqlite3 έχει δημιουργηθεί μία αρχική βάση δεδομένων (*library.db*), η οποία σας δίνεται και περιέχει τους εξής πίνακες:

α) Μαθητές (**students**), με τα εξής πεδία: κωδικός (id), όνομα (name) και επώνυμο (surname). Ο κωδικός του κάθε μαθητή (id) δημιουργείται με αυτόματη αρίθμηση
β) Βιβλία (**books**), με τα εξής πεδία: κωδικός ISBN (isbn), τίτλος (title), εκδότης (editor) και έτος έκδοσης (year) και
γ) Δανειστικές πράξεις (**lending**), με τα εξής πεδία: αριθμός (no), κωδικός μαθητή (student_id), κωδικός ISBN (isbn), ημερομηνία δανεισμού (lend_date) και ημερομηνία επιστροφής (return_date). Ο αριθμός της δανειστικής πράξης (no) δημιουργείται με αυτόματη αρίθμηση.

Στους παραπάνω πίνακες της βάσης δεδομένων library.db έχουν γίνει κάποιες αρχικές καταχωρήσεις δεδομένων. Χρησιμοποιείστε την εφαρμογή DB browser for SQLite για να επιθεωρήσετε τα δεδομένα (δείτε τη σχετική σημείωση το τέλος της εκφώνησης).

Ζητείται να υλοποιηθεί εφαρμογή, η οποία θα εμφανίζει μενού με τις ακόλουθες επιλογές: 1) Προβολή μαθητών, 2) Προβολή βιβλίων, 3) Προβολή δανεισμών, 4) Καταχώρηση μαθητή, 5) Διαγραφή μαθητή, 6) Έξοδος.

Ειδικότερα:

α) Θα πρέπει να ορίσετε τη συνάρτηση *show_records(table)*, η οποία θα παρουσιάζει τις εγγραφές ενός από τους πίνακες students, books ή lending. Η συνάρτηση θα δέχεται ως όρισμα το όνομα του πίνακα. Για την προβολή των εγγραφών του πίνακα θα πρέπει να χρησιμοποιήσετε το ερώτημα (sql script): "SELECT * from {}". Η συνάρτηση show_records (table) θα πρέπει να εμφανίζει τις ονομασίες των πεδίων του πίνακα καθώς και τις σχετικές εγγραφές.
β) Θα πρέπει να ορίσετε τη συνάρτηση insert_student(name, surname), η οποία θα καταχωρεί ένα νέο μαθητή στον πίνακα students. Η συνάρτηση θα δέχεται ως ορίσματα το όνομα και το επώνυμο του μαθητή. Για την εισαγωγή ενός νέου μαθητή θα πρέπει να χρησιμοποιήσετε το sql script: "INSERT INTO students (name, surname) VALUES (? , ?)"
γ) Θα πρέπει να ορίσετε τη συνάρτηση *delete_student(code)*, η οποία θα διαγράφει ένα μαθητή από τον πίνακα students. Η συνάρτηση θα δέχεται ως όρισμα τον κωδικό του μαθητή. Για τη διαγραφή ενός μαθητή θα πρέπει να χρησιμοποιήσετε το sql script: "DELETE FROM students WHERE id==(?)"

Σε καθεμία από τις παραπάνω συναρτήσεις, να χρησιμοποιήσετε χειρισμό λαθών (π.χ. με τη δομή try/except ή/και τη δομή with) κατά τη σύνδεση με τη βάση δεδομένων library.db και την εκτέλεση των εντολών SQL.

Υπόδειξη: Να χρησιμοποιηθεί το αρχείο **3_code_template.py** ως οδηγός επίλυσης.

Σημείωση: *Μπορείτε να διαχειριστείτε τη βάση δεδομένων library.db με τη χρήση της εφαρμογής DB Browser (SQLite). Μπορείτε να εγκαταστήσετε τη συγκεκριμένη εφαρμογή στον υπολογιστή σας από την ιστοσελίδα:* https://sqlitebrowser.org/dl/

ΥΠΟΕΡΓΑΣΙΑ 4. (βαθμοί 25)

Η βιβλιοθήκη NumPy προσφέρει ένα σύνολο από συναρτήσεις κατάλληλες για την αποδοτική επεξεργασία και εκτέλεση μαθηματικών συναρτήσεων ή αλγεβρικών πράξεων. Από τις σημαντικότερες συναρτήσεις που διαθέτει η βιβλιοθήκη είναι αυτές που διαχειρίζονται τις διανυσματικές πράξεις (πράξεις μεταξύ πινάκων). Η βιβλιοθήκη προσφέρει συναρτήσεις, οι οποίες είναι έως και 3 τάξεις μεγέθους (1000 φορές) πιο γρήγορες από τις τυπικές υλοποιήσεις με βρόχους (for loops).

Για να γίνει αντιληπτή η απόδοση των συναρτήσεων της NumPy ζητείται να υλοποιήσετε πρόγραμμα που συγκρίνει τον χρόνο εκτέλεσης μιας τυπικής υλοποίησης υπολογισμού του εσωτερικού γινομένου δύο διανυσμάτων (πινάκων μίας διάστασης) με μια υλοποίηση της ίδιας πράξης με συναρτήσεις της NumPy.

Για τις ανάγκες της σύγκρισης, το πρόγραμμα παράγει δυο μεγάλα διανύσματα με τυχαίους αριθμούς, εκτελεί πολλές φορές επαναληπτικά τον υπολογισμό του γινομένου διανυσμάτων (dot product) τόσο με τον τυπικό τρόπο, όσο και με κλήση της συνάρτησης *dot* της numpy, υπολογίζει και εκτυπώνει τον χρόνο εκτέλεσης των υπολογισμών.

Ειδικότερα, ζητείται να συμπληρώσετε το υπόδειγμα με:

α) Την τυπική υλοποίηση του γινομένου διανυσμάτων, χωρίς τη χρήση της numpy, στη συνάρτηση *internalProduct(v1, v2)*, η οποία δέχεται ως ορίσματα τα δύο διανύσματα (δείτε παρακάτω σημείωση–υπόδειξη για υλοποίηση του αλγορίθμου).

β) Την υλοποίηση της συνάρτησης *internalProduct_np(v1, v2)*, η οποία υπολογίζει το γινόμενο με χρήση της numpy.

γ) Τη δημιουργία της συνάρτησης *timeit(mode, rep,v1,v2)*, η οποία δέχεται ως όρισμα τον τρόπο υπολογισμού mode, το πλήθος επαναλήψεων υπολογισμού rep, και τα δύο διανύσματα v1, v2. Η συνάρτηση επαναλαμβάνει τον υπολογισμό όσες φορές ορίζει η rep και μετράει τον συνολικό χρόνο που απαιτήθηκε. Επίσης, τυπώνει το αποτέλεσμα του υπολογισμού. Να σημειωθεί ότι η συνάρτηση αυτή καλεί είτε τη συνάρτηση *internalProduct* αν mode="τυπική υλοποίηση", είτε τη συνάρτηση *internalProduct_np* αν mode="numpy".

δ) Την υλοποίηση του κυρίως προγράμματος, το οποίο, καλώντας τις παραπάνω συναρτήσεις, κάνει τον υπολογισμό με διαφορετικές μεθόδους και εμφανίζει τον χρόνο επεξεργασίας, και τον λόγο χρόνου υπολογισμού με τον τυπικό τρόπο προς τον χρόνο υπολογισμού με την dot της numpy

Υπόδειξη: Να χρησιμοποιηθεί το αρχείο 4_code_template.py ως οδηγός επίλυσης.

Σημείωση: *Το εσωτερικό γινόμενο δύο διανυσμάτων είναι ίσο με το άθροισμα των γινομένων των ομώνυμων στοιχείων τους. Έτσι η τυπική υλοποίηση υπολογισμού του εσωτερικού γινομένου δύο διανυσμάτων γίνεται πολλαπλασιάζοντας ανά δύο τα ομώνυμα στοιχεία τους (δηλαδή, αυτά που βρίσκονται στην ίδια θέση) και αθροίζοντας τα γινόμενα. Ο εν λόγω αλγόριθμος συνοψίζεται στα παρακάτω βήματα:*

1. *Αρχικοποίησε το αποτέλεσμα σε 0 (μηδέν)*
2. *Επανάλαβε για κάθε θέση*
   - *2.1. Πολλαπλασίασε τα στοιχεία των διανυσμάτων που βρίσκονται στην αντίστοιχη θέση*
   - *2.2. Πρόσθεσε το γινόμενο των στοιχείων στο αποτέλεσμα*

Γενικές Υποδείξεις:

I) Για τις απαντήσεις της εργασίας μπορείτε να ανατρέξετε στη συμπληρωματική βιβλιογραφία που δίνεται και στα βοηθητικά κείμενα που υπάρχουν στον δικτυακό τόπο / portal της θεματικής ενότητας. Συνιστάται να προσθέσετε στο τέλος της εργασίας σας κατάλογο βιβλιογραφίας.

II) Οδηγίες σχετικές με τον κώδικα

- Το όνομα κάθε .py αρχείου να περιλαμβάνει το επώνυμό σας με λατινικούς χαρακτήρες, το χαρακτήρα της υπογράμμισης και τον αριθμό του συγκεκριμένου υποερωτήματος (π.χ. αν το επώνυμό σας είναι Γεωργίου, τότε ο κώδικας για την υποεργασία 1β θα έχει το όνομα Georgiou_1b.py). Κάθε αρχείο κώδικα που θα παραδοθεί θα πρέπει τουλάχιστον να περνάει τη φάση της διερμηνείας χωρίς συντακτικά σφάλματα.
- Τα αρχεία .py θα πρέπει να τα ανοίξετε και να τα επεξεργαστείτε με το ολοκληρωμένο περιβάλλον ανάπτυξης κώδικα IDLE. Ο κώδικας να είναι επαρκώς σχολιασμένος, σωστά στοιχισμένος και ενσωματωμένος μέσα στο έγγραφο Word, με τις απαντήσεις σας σε γραμματοσειρά courier.
- Στο έγγραφο της απάντησής σας και στο αρχείο του κώδικα θα πρέπει να δίνεται ολόκληρο το πρόγραμμα, επισημαίνοντας με σχόλια πού απαντάτε κάθε ερώτημα ώστε να θεωρούνται πλήρεις οι απαντήσεις.
- Όλα τα .py αρχεία με τον πηγαίο κώδικα και το .doc αρχείο κειμένου να υποβληθούν μέσω της πλατφόρμας **https://study.eap.gr**.

III) Τρόπος παράδοσης εργασίας:

α) Οι απαντήσεις πρέπει να είναι γραμμένες με χρήση **επεξεργαστή κειμένου** (π.χ. **Word**) σε σελίδες **διαστάσεων Α4 χωρίς χρώματα.** Το αρχείο να περιέχει ως **πρώτη σελίδα** το κείμενο του **Εντύπου Υποβολής – Αξιολόγησης** και ως δεύτερη σελίδα τον τίτλο «Σχόλια προς τον φοιτητή» (θα συμπληρωθεί από τον καθηγητή σας). Οι απαντήσεις στις υποεργασίες θα αρχίζουν από την τρίτη σελίδα, **χωρίς να επαναλαμβάνονται οι εκφωνήσεις**. Κάθε υποεργασία θα αρχίζει από νέα σελίδα. Για την απάντησή σας θα πρέπει να χρησιμοποιείτε υποχρεωτικά το **Πρότυπο Υποβολής Γραπτής Εργασίας**.

β) Το .doc αρχείο κειμένου να υποβληθεί στη διεύθυνση **https://study.eap.gr** με **όνομα αρχείου το επώνυμό σας με λατινικούς χαρακτήρες και τον Αριθμό Μητρώου σας**, π.χ. Ioannou_82345.

IV) Η καλή παρουσίαση της εργασίας λαμβάνεται υπόψη στην αξιολόγηση της εργασίας.

****