ΦΥΣ 140 – Εισαγωγή στην Επιστημονική Χρήση Υπολογιστών

1η Εργασία **Επιστροφή: 29/9/2023**

Υπενθύμιση: Οι εργασίες πρέπει να επιστρέφονται με e-mail στο ptochos.fotios@ucy.ac.cy που θα στέλνετε από το πανεπιστημιακό σας λογαριασμό το αργότερο μέχρι την ημερομηνία που αναγράφεται.
Ως subject του e-mail θα πρέπει να αναγράφεται την εργασία (username_phy140_hmX όπου X ο αριθμός της εργασίας, 01, 02, 03…)
Κάθε αρχείο που επισυνάπτετε (attach) στο e-mail σας θα πρέπει να έχει το όνομα στη μορφή username_hmX.tgz όπου username είναι το username του e-mail σας και X ο αριθμός της εργασίας. Επίσης σαν πρώτο σχόλιο μέσα σε κάθε file που περιέχει το πρόγραμμά σας θα πρέπει να αναφέρεται το ονοματεπώνυμό σας. Οι εργασίες είναι ατομικές και πανομοιότυπες εργασίες δε θα βαθμολογούνται. Για να κάνετε ένα tgz file (ουσιαστικά tar zipped file) θα πρέπει να δώσετε στο terminal την εντολή tar -czvf username_hmX.tgz *.py όπου py είναι όλα τα py files των προγραμμάτων σας.

1. Στο εργαστήριο 2 γράψατε ένα πρόγραμμα το οποίο μελετούσε την κίνηση ενός σώματος. Στην άσκηση αυτή θα κατασκευάσετε την γραφική παράσταση της θέσης του σώματος συναρτήσει του χρόνου. Για να το κάνετε αυτό θα πρέπει να κάνετε import στον κώδικα της άσκησης 1 του 2ου εργαστηρίου την βιβλιοθήκη matplotlib.pyplot. Όπως έχουμε δει στην διάλεξη 2, θα πρέπει να γράψουμε στην αρχή του προγράμματος:

import matplotlib.pyplot as plt

Μπορούμε να δώσουμε πολλές τιμές στον χρόνο *t* και να υπολογίσουμε τις αντίστοιχες τιμές της θέσης. Για παράδειγμα μπορούμε να μελετήσουμε την κίνηση του σώματος για 2sec και να έχουμε 500 υποδιαστήματα από 0 έως 2s. Αυτό μπορούμε να το κάνουμε δίνοντας τη συνάρτηση *linspace*(*0,2,501*) όπου 0 είναι η αρχική χρονική στιγμή, 2 η τελική χρονική στιγμή και 501 είναι το πλήθος των χρονικών στιγμών από 0 έως και 2s.

Μπορούμε πλέον να βρούμε τη θέση για κάθε μια χρονική στιγμή, δίνοντας την εντολή:

*y = v0*t – 0.5*g*t*t*

Θα υπολογίσει 501 τιμές θέσης για τις αντίστοιχες χρονικές στιγμές.

Μπορούμε να κάνουμε το γράφημα τώρα δίνοντας τις ακόλουθες εντολές:

plt.plot(*t,y*)

plt.xlabel(*‘t* (s)’)

plt.ylabel(‘*y* (*m*)’)

plt.show()

Τροποποιήστε την 1η άσκηση του εργαστηρίου 2 που γράψατε ώστε να κάνετε την γραφική παράσταση θέσης – χρόνου για τα πρώτα 5s της κίνησης του σώματος υπολογίζοντας τη θέση του σώματος κάθε 0.01s.

2. Γράψτε το ακόλουθο πρόγραμμα *ball.py*

#Program for computing the height of a ball in vertical motion

```
v0 = 5      # initial velocity
g = 9.81    # acceleration of gravity
t = 0.6     # time
y = v0*t – 0.5*g*t**2
print(y)
```

Γράψτε σε ένα αρχείο *hm02_askisi02.txt* τις απαντήσεις και εξηγήσεις στα σφάλματα που θα εμφανιστούν στην οθόνη σας όταν κάνετε τα ακόλουθα και τρέξετε το πρόγραμμα κάθε φορά:

(α) Εισάγετε την λέξη *hello* στη κενή γραμμή πάνω από την εντολή ανάθεσης *v0 = 5*.

(β) Σβήστε το # μπροστά από το initial velocity και τρέξτε το πρόγραμμα.

(γ) Σβήστε το = στην εντολή ανάθεσης v0.

(δ) Αλλάξτε την λέξη *print* σε *pint*

(ε) Αλλάξτε τον υπολογισμό του *y* σε *y=v0*t*

(στ) Αλλάξτε την εντολή *print*(*y*) σε *print*(*x*)

(ζ) Αλλάξτε την εντολή ανάθεσης *y = v0*t – 0.5*g*t**2* σε *y = v0*t – (1/2)*g*t**2*

3. Θέλουμε να υπολογίσουμε τον όγκο τριών κύβων διαφορετικού μήκους πλευράς.

(α) Χρησιμοποιήστε μία λίστα για να αποθηκεύσετε τις τιμές του μήκους της πλευράς κάθε κύβου. Οι τιμές θα πρέπει να δίνονται από το πληκτρολόγιο.

(β) Γράψτε το πρόγραμμα το οποίο υπολογίζει τον όγκο των αντίστοιχων κύβων.

(γ) Μετατρέψτε το πρόγραμμά σας ώστε να υπολογίζει τον όγκο των τριών κύβων σε μια επαναληπτική διαδικασία χρησιμοποιώντας τα στοιχεία της λίστας.

(δ) Κάντε το γράφημα του όγκου, *V*, ως προς το μήκος της πλευράς *L* χρησιμοποιώντας τις πληροφορίες που δίνονται στην άσκηση 1.

4. Γράψτε ένα πρόγραμμα που να υπολογίζει τον αριθμό των δευτερολέπτων *s,* όταν δίνεται ένα χρονικό διάστημα στην μορφή *Δt = hh.mm.* Για παράδειγμα για το διάστημα *Δt = 1.32h* αντιστοιχεί σε *5520s*. Ο χρόνος θα πρέπει να δίνεται από το πληκτρολόγιο και η απάντηση να τυπώνεται στην οθόνη.

5. Υποθέστε ότι έχετε ένα πολύγωνο όπως στο σχήμα. Οι κορυφές του πολυγώνου έχουν συντεταγμένες $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ αριθμημένες είτε κινούμενοι αριστερόστροφα είτε δεξιόστροφα. Το εμβαδό, *A,* του πολυγώνου μπορεί να υπολογιστεί γνωρίζοντας τις συντεταγμένες των κορυφών:

$$A = \frac{1}{2}\left|(x_1 y_2 + x_2 y_3 + \cdots + x_{n-1} y_n + x_n y_1) - (y_1 x_2 + y_2 x_3 + \cdots + y_{n-1} x_n + y_n x_1)\right|$$

Γράψτε ένα πρόγραμμα το οποίο γεμίζει δύο λίστες X και Y με τις συντεταγμένες *x* και *y* των κορυφών του πολυγώνου και υπολογίζει το εμβαδόν. Δοκιμάστε το πρόγραμμά σας για τις περιπτώσεις του τριγώνου, παραλληλογράμμου, πενταγώνου και συγκρίνετε τα αποτελέσματα του προγράμματός σας με αυτά που παίρνετε με διαφορετικούς τρόπους υπολογισμού του εμβαδού των συγκεκριμένων περιπτώσεων. ***Υπόδειξη:*** (α) Μη ξεχάσετε ότι στις λίστες η αρίθμηση των στοιχείων ξεκινά από το 0 οπότε ο παραπάνω τύπος θα πρέπει να ξαναγραφεί για να σας βοηθήσει στην αρίθμηση. (β) Για να δείτε τα αποτελέσματα θα πρέπει να τυπώσετε την αντίστοιχη μεταβλητή που ορίζετε για το εμβαδό. Για παράδειγμα print(“To emvado isoutai me: “, A)

6. Θεωρήστε έναν κύκλο και ένα ορθογώνιο. Ο κύκλος έχει ακτίνα r = 10.6 (τυχαίες μονάδες μέτρησης). Το ορθογώνιο έχει πλευρές μήκους a και b αλλά μόνο το μήκος της πλευράς a είναι γνωστό από την αρχή και ίσο με a = 1.3. Γράψτε ένα πρόγραμμα το οποίο χρησιμοποιεί την δομή ενός βρόχου (το while loop) το οποίο βρίσκει τον μεγαλύτερο δυνατό ακέραιο b, ώστε το εμβαδό του ορθογωνίου είναι μικρότερο αλλά όσο το δυνατόν πιο κοντά στο εμβαδό του κύκλου.

7. Θεωρήστε δύο συναρτήσεις $f(x) = x$ και $g(x) = x^2$ στο διάστημα τιμών του x, [-4, 4]. Γράψτε ένα πρόγραμμα το οποίο δοκιμάζοντας διάφορες τιμές του x, βρίσκει προσεγγιστικά την τιμή του x για την οποία οι δύο συναρτήσεις τέμνονται, δηλαδή $g(x) = f(x)$. Για να το επιτύχετε αυτό μπορείτε να θεωρήσετε N ισαπέχουσα σημεία στο διάστημα [-4, 4] και σε κάθε σημείο ελέγξετε αν $|f(x) - g(x)| < \varepsilon$, όπου ε είναι κάποιος μικρός αριθμός. Τα N και ε πρέπει να εισάγονται από το πληκτρολόγιο και τα αποτελέσματα να τυπώνονται στην οθόνη. Τρέξτε το πρόγραμμά σας για N = 400 και ε = 0.01. Προσπαθήστε να εξηγήσετε το αποτέλεσμα που βλέπετε. Κρατώντας το ε σταθερό τρέξτε το πρόγραμμά σας για διαφορετικές τιμές του N και προσπαθήστε να εξηγήσετε τα αποτελέσματά σας.

8. Στο παρελθόν πολλοί εξαίρετοι επιστήμονες πρότειναν υπολογιστικούς αλγορίθμους για την εύρεση της τιμής του π. Στην άσκηση αυτή θα ασχοληθείτε με δύο τέτοιους αλγορίθμους.
Ο πρώτος αλγόριθμος προτάθηκε από τον Leibniz (1646 – 1716) και στηρίζεται στη σχέση:

$$\pi = 8\sum_{k=1}^{\infty}\frac{1}{(4k+1)(4k+3)}$$

Ο δεύτερος αλγόριθμος προτάθηκε από τον Euler (1707 – 1783) και στηρίζεται στη σχέση:

$$\pi = \sqrt{6\sum_{k=1}^{\infty}\frac{1}{k^2}}$$

Αν μόνο οι N πρώτοι όροι του αθροίσματα της κάθε σειράς αλγορίθμου χρησιμοποιηθούν στον υπολογισμό της τιμής του π, η τιμή αυτή δε θα είναι ακριβής αλλά θα περιέχει κάποιο σφάλμα. Το σφάλμα ορίζεται ως η απόλυτη τιμή της διαφοράς της τιμής του αθροίσματος από την τιμή του π που επιστρέφει η συνάρτηση ***pi*** της βιβλιοθήκης numpy.
Γράψτε ένα πρόγραμμα το οποίο λαμβάνει το πλήθος N των όρων της σειράς που θα χρησιμοποιηθούν στα παραπάνω αθροίσματα και κάνει το γράφημα του σφάλματος του υπολογισμού της τιμής του π καθώς ο αριθμός των όρων που χρησιμοποιούνται στο άθροισμα πλησιάζει το N. Το πρόγραμμά σας θα πρέπει να τυπώνει το τελικό σφάλμα που βρέθηκε και για τους 2 αλγορίθμους, δηλαδή το σφάλμα όταν έχουν χρησιμοποιηθεί ακριβώς οι N όροι. Δοκιμάστε το πρόγραμμά σας για N=100 και εξηγήστε σαν σχόλιο στο πρόγραμμά σας τι είναι αυτό που δείχνει το γράφημα που παίρνετε. *Υπόδειξη:* Για το γράφημα θα χρησιμοποιήσετε ότι αναφέρθηκαν και στην 1η άσκηση. Για κάθε όρο του αθροίσματος θα πρέπει να φυλάξετε το σφάλμα σε μια λίστα. Μπορείτε να διαφοροποιήσετε τα δύο γραφήματα δίνοντας διαφορετικά χρώματα στις καμπύλες. Για παράδειγμα χρησιμοποιώντας την εντολή από την άσκηση 1, θα μπορούσαμε να γράψουμε *plt.plot*(*t,y,’b-‘*) και η καμπύλη του γραφήματος θα ήταν χρώματος μπλέ και συνεχής. Θα μπορούσε να είχαμε ‘*r-*‘ για κόκκινο χρώμα.