ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

Τεχνητή Αναφορά 1ης Εργασίας Εξαμήνου

Αλγόριθμοι & Πολυπλοκότητα

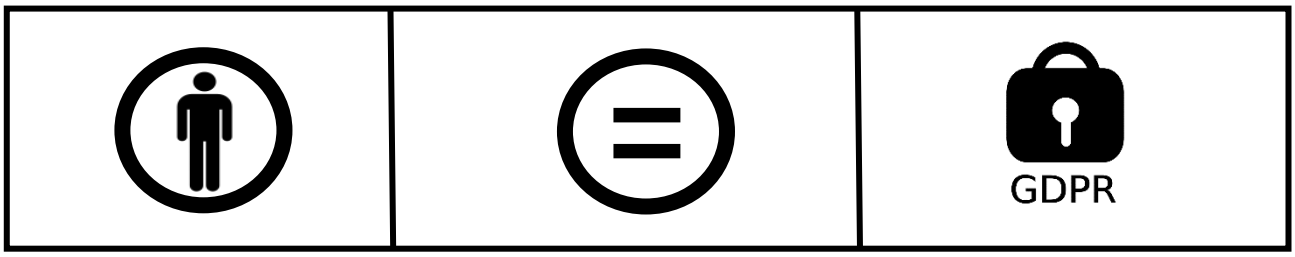
Βίννη Παναγιώτα

Α.Μ. :1873

penyvinni@gmail.com

Άρτα, 2021

Περιεχόμενα



1. Υλικό που χρησιμοποιήθηκε για τα πειράματα της εργασίας

Υλικό	Χαρακτηριστικά
Επεξεργαστής	Intel(R) Core(TM) i7-1065G7 CPU @ 1.30GHz 1.50 GHz
RAM	8 GB (7,70 GB χρησιμοποιήσιμη)
Τύπος Συστήματος	Λειτουργικό σύστημα 64 bit, επεξεργαστής τεχνολογίας x64
Λειτουργικό Σύστημα	Windows 10 Home

2. Το λογισμικό που χρησιμοποιήθηκε στην εργασία

- ✓ Vs Code
- ✓ Python 3.9.2 64 – bit
- ✓ Matplotlib
- ✓ Excel

3. Η θεωρητική πολυπλοκότητα των αλγορίθμων

➔ **First Algorithm (Kadane's) :** Υπολογίζει τη max υποακολουθία χρησιμοποιώντας μόνο 1 for loop, δηλαδή με O (n) πολυπλοκότητα. Στην ουσία από μία λίστα παίρνουμε μόνο τα θετικά στοιχεία και όπου υπάρχει αρνητικός αριθμός το μετατρέπει σε 0 κι έτσι βγαίνει το μεγαλύτερο άθροισμα στην max υποακολουθία.

➔ **Second Algorithm (Prefix) :** Υπολογίζει τη max υποακολουθία χρησιμοποιώντας 2 for loops, δηλαδή με O (n^2) πολυπλοκότητα. Στην ουσία από μία λίστα δημιουργείται μία άλλη ξεχωριστή με όνομα prefix με τα αθροίσματα των στοιχείων της (το προηγούμενο του prefix με το επόμενο της λίστας) και από αυτά βρίσκει την max υποακολουθία.

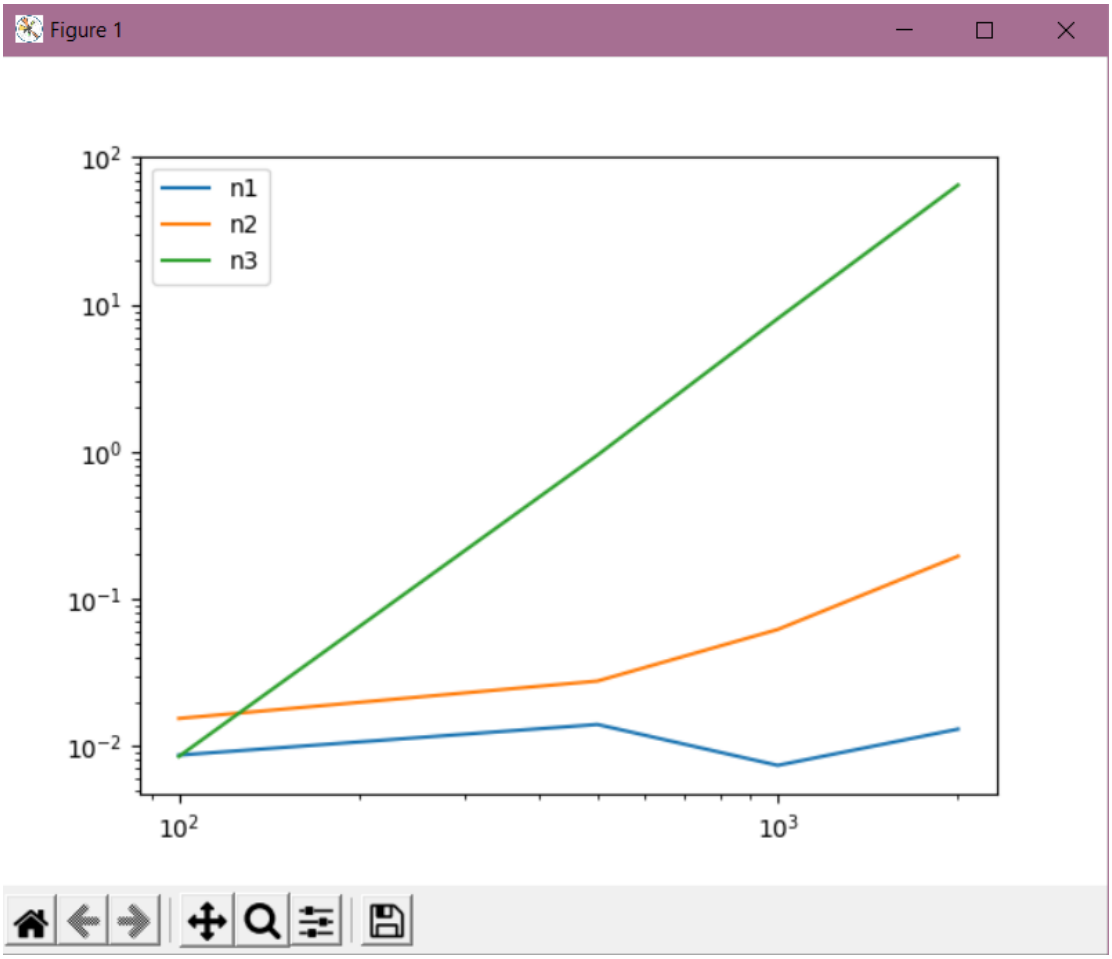
➔ **First Algorithm (Kadane's) :** Υπολογίζει τη max υποακολουθία χρησιμοποιώντας 3 for loops, δηλαδή με O (n^3) πολυπλοκότητα. Στην ουσία από μία λίστα υπολογίζει όλα τα αθροίσματα όλων των στοιχείων και το μεγαλύτερο που θα βρει θα είναι το max sum στην max υποακολουθία.

4. Οι μετρήσεις χρόνων εκτέλεσης των αλγορίθμων με εισόδους : 100, 500, 1000, 2000 (για λόγους ευκολίας χρησιμοποιήθηκαν μικρότεροι αριθμοί)

n	100	500	1000	2000
O (n)	0,00155	0,00348	0,00491	0,00713
O (n^2)	0,00377	0,02526	0,09720	0,37322
O (n^3)	0,01008	1,05861	8,33476	135,31060

5. Γραφική σύγκριση μέσω διαγραμμάτων των αποτελεσμάτων που παράγουν οι αλγόριθμοι

Για τη δημιουργία του παρακάτω διαγράμματος με τους χρόνους εκτέλεσης κάθε αλγορίθμου χρησιμοποιήθηκε η βιβλιοθήκη matplotlib της python. Παρατηρούμε ότι ο αλγόριθμος με πολυπλοκότητα O (n^3) καθυστερεί αρκετά σε σχέση με τους άλλους δύο, για αυτό και δεν είναι καθόλου αποδοτικός εν αντιθέσει με τον αλγόριθμο του O (n) ο οποίος τρέχει σε ελάχιστα δευτερόλεπτα.



Τέλος, για δεύτερη λύση χρησιμοποιήθηκε και το πρόγραμμα Excel, στο οποίο τοποθετήθηκαν χειρόγραφα τα αποτελέσματα και με βάση αυτά δημιουργήθηκε το παρακάτω διάγραμμα:

n	100	500	1000	2000
n1	0,00155	0,00348	0,00491	0,00713
n2	0,00377	0,02526	0,0972	0,37322
n3	0,01008	1,05861	8,33476	135,3106