ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ - ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
Ειδικά Θέματα Αλγορίθμων
Διδάσκων: Ε. Μαρκάκης, Φθινοπωρινό εξάμηνο 2025

1η σειρά ασκήσεων **Προθεσμία παράδοσης: 7 Δεκεμβρίου 2025**

Πρόβλημα 1. (15 μονάδες)

Στον παρακάτω πίνακα, δίνεται η είσοδος που λαμβάνουν κάποιοι αλγόριθμοι και η πολυπλοκότητά τους. Αποφασίστε για τον κάθε αλγόριθμο αν πρέπει να χαρακτηρισθεί ως πολυωνυμικός, ψευδοπολυωνυμικός ή εκθετικός. Δεν απαιτείται να αιτιολογήσετε την απάντησή σας (μπορείτε αν θέλετε να γράψετε 1 πρόταση αναφέροντας γιατί επιλέξατε όποια απάντηση επιλέξατε σε κάθε αλγόριθμο). Επίσης, όλοι οι παράμετροι που αναφέρονται στον πίνακα είναι θετικοί ακέραιοι, και ο συμβολισμός $[n]$ στον προτελευταίο αλγόριθμο υποδηλώνει το σύνολο $\{1, 2 \cdots, n\}$.

Input	Running time
Γράφος με n κορυφές	$\Theta(\sqrt{2^n})$
Σύνολο n αντικειμένων, με κόστη $c_1, c_2, \ldots, c_n$	$\Theta(n^2 \cdot (\log c_2)^3)$
Γράφος με n κορυφές και κόστη στις κορυφές $c_1, c_2, \ldots, c_n$	$\Theta(n^3 + min_i\ c_i)$
Σύνολο n αντικειμένων και μια παράμετρος κόστους για κάθε $S \subseteq [n]$	$\Theta(2^n)$
Γράφος με n κορυφές και κόστη στις κορυφές $c_1, c_2, \ldots, c_n$	$\Theta(n^2 \cdot \log(\sum_{i=1}^{n} c_i))$

Πίνακας 1: Είσοδος και χρόνος εκτέλεσης των αλγορίθμων.

Πρόβλημα 2. (16 μονάδες)

(i) (6 μονάδες) Θεωρήστε την ακολουθία των αριθμών Fibonacci, με $F_n = F_{n-1} + F_{n-2}$, $F_1 = 1$, $F_0 = 0$. Απόδειξτε ότι ισχύει η παρακάτω σχέση για κάθε $k \geq 1$.

$$F_{n+k} = F_{n+1} \cdot F_k + F_n \cdot F_{k-1}$$

(ii) (10 μονάδες) Έστω ακέραιος $x > 35$ με $gcd(x, 35) = 1$. Χρησιμοποιώντας το μικρό θεώρημα του Fermat και το Κινέζικο θεώρημα υπολοίπων, να δείξετε ότι $x^{12} \equiv 1\ (mod\ 35)$. Δεν επιτρέπεται να χρησιμοποιήσετε το θεώρημα του Euler.

Πρόβλημα 3. (23 μονάδες)

(i) (5 μονάδες) Υπολογίστε το $\phi(243)$.

(ii) (10 μονάδες) Χωρίς να χρησιμοποιήσετε κομπιουτεράκι ή οποιοδήποτε άλλο μέσο (ούτε τον αλγόριθμο επαναλαμβανόμενου τετραγωνισμού), υπολογίστε τις ποσότητες: $7 \cdot 3^{58}\ mod\ 19$, και $(3 \cdot 7^{17} + 11^{33} + 3 \cdot 13^{49})\ mod\ 60$.

(iii) (8 μονάδες) Θεωρήστε το Fermat test που παρουσιάστηκε στις διαφάνειες σαν ένας αλγόριθμος για το primality testing. Έστω ότι ένας αριθμός n είναι σύνθετος και δεν είναι

αριθμός Carmichael. Έστω επίσης ότι για να τρέξετε το Fermat test, επιλέγετε τυχαία έναν αριθμό α, έτσι ώστε $\alpha \in \{1, \ldots, n-1\}$ και $gcd(\alpha, n) = 1$. Να δείξετε ότι τουλάχιστον για τις μισές από τις πιθανές επιλογές για το α, ο αριθμός n δεν θα περάσει το Fermat test (δηλαδή θα αναγνωρισθεί ως σύνθετος).

Πρόβλημα 4. (17 μονάδες) Έστω ότι σε ένα σύστημα RSA, ο Oscar υποκλέπτει το ciphertext, το οποίο είναι ίσο με $C = 10$. Αν ξέρει ότι το ciphertext αυτό είχε σταλεί σε ένα χρήστη με public key $e = 13$, και $n = 35$, ποιο ήταν το plaintext? Δείξτε αναλυτικά όλα τα βήματα που πρέπει να ακολουθήσετε για να βρείτε το μήνυμα. Αν χρειαστεί να υψώσετε σε δύναμη για τους υπολογισμούς που θα κάνετε, θα πρέπει να χρησιμοποιήσετε τον αλγόριθμο επαναλαμβανόμενου τετραγωνισμού.

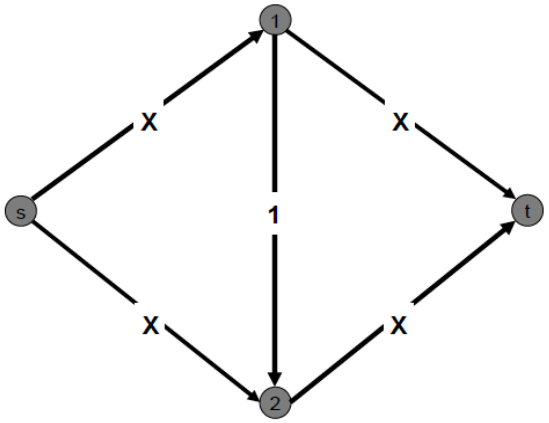
Πρόβλημα 5. (14 μονάδες) Έστω ένας πίνακας Α n θέσεων, από το 1 ως το n. Θεωρήστε τον παρακάτω ψευδοκώδικα, που παίρνει ως είσοδο έναν ακέραιο m με $1 \leq m \leq n$.

```
int foo(int m)
if (m > √n) return A[n]
else {
    int x=0
    for i = 1 to n    {x = x + A[i]}
    return x
    }
```

(i) (7 μονάδες) Αναλύστε την πολυπλοκότητα καλύτερης, χειρότερης και μέσης περίπτωσης. Για την ανάλυση της μέσης περίπτωσης, θεωρήστε ότι ο ακέραιος m επιλέγεται τυχαία με ομοιόμορφη κατανομή από τους ακεραίους $1, 2, \ldots, n$.

(ii) (7 μονάδες) Απαντήστε στα ίδια ερωτήματα αν τώρα αντικαταστήσουμε την συνθήκη του if με if $(m \leq n/2)$.

Πρόβλημα 6. (15 μονάδες) Δίνεται ο παρακάτω κατευθυνόμενος γράφος G, όπου σε κάθε ακμή φαίνεται και η χωρητικότητά της.



Να εξηγήσετε πόσες επαναλήψεις θα χρειαστεί στην χειρότερη περίπτωση ο αλγόριθμος των Ford-Fulkerson, ως συνάρτηση του X, για τη μεγιστοποίηση της ροής. Να δείξετε τουλάχιστον τις πρώτες 3 επαναλήψεις του αλγορίθμου και στη συνέχεια μπορείτε να εξηγήσετε αν διαφαίνεται κάποιο μοτίβο για τις επόμενες επαναλήψεις, από το οποίο να μπορείτε να συμπεράνετε το συνολικό πλήθος επαναλήψεων.

Πρόβλημα 7. (10 μονάδες) Extra credit: Χρησιμοποιήστε την σχέση που αποδείξατε στο Πρόβλημα 2- (i) για να σχεδιάσετε έναν $O(logn)$ αλγόριθμο για τον υπολογισμό του n-οστού αριθμού Fibonacci. Επιχειρηματολογήστε αν ο αλγόριθμος αυτός είναι πιο γρήγορος στην πράξη από τον άλλο αλγόριθμο που έχουμε δει με πολυπλοκότητα $O(logn)$.