ΗΥ380 Θέματα Ιούνιος

θέμα 1ο
http://cseweb.ucsd.edu/classes/su00/cse101/l2.pdf
Στην Α’ δεν ειναι n/4 ειναι n/2 στο δικο μας.

θεμα 2ο
a) DFS παραλλαγη για ευρεση κυκλου σε undirected graph
b) DFS παραλλαγή για εύρεση κύκλου
c) Bellman-Ford σελίδα 590

θέμα 3ο
http://www.csd.uoc.gr/~hy380/old/shmeiwseis/DivideAndConquer.pdf
είναι στην τελευταία διαφάνεια
goodrich selida 270
Θέμα 4ο
a) search: problem 34-3 http://cc.ee.ntu.edu.tw/~ywchang/Courses/Alg/ss5.pdf
b) search: (theorem 2.2) http://www.cs.cmu.edu/~./anupamg/251-notes/graphs2.pdf
c) http://www2.ee.ntu.edu.tw/~yen/courses/al-03/HW3Sol.pdf
d) http://www2.ee.ntu.edu.tw/~yen/courses/al-03/HW3Sol.pdf

θέμα 5ο
1.Περιγράψτε ενα μη τεριμένο κάτω φράγμα (πχ φράγμα για ταξινόμιση με σύγκριση)
Απάντηση
Είναι στο βιβλίο σελίδα
162
Ή απο εδώ http://www.csd.uoc.gr/~hy380/old/shmeiwseis/SortingLowerBound.pdf
ok

2.Ποια η σημασία των ενοιών P,NP,NP-Complete,NP-Hard
Απάντηση
http://cgi.di.uoa.gr/~vassilis/ac/VZalgorithms08.pdf

P είναι μια κλάση πολυπλοκότητας που αντιπροσωπεύει το σύνολο όλων των προβλημάτων απόφασης που μπορεί να λυθεί σε πολυωνυμικό χρόνο. (Δηλαδή, δίνεται ένα στιγμιότυπο του προβλήματος, η απάντηση ναι ή όχι μπορεί να αποφασιστεί σε πολυωνυμικό χρόνο).
NP είναι μια κλάση πολυπλοκότητας που αντιπροσωπεύει το σύνολο όλων των προβλημάτων απόφασης για τα οποία η απάντηση είναι ναι, έχουμε αποδείξεις ότι μπορεί να επαληθευτεί σε πολυωνυμικό χρόνο.

NP- complete είναι μια κλάση πολυπλοκότητα η οποία αντιπροσωπεύει το σύνολο όλων των προβλημάτων X για τις οποίες είναι δυνατόν να μειωθεί οποιοδήποτε άλλο πρόβλημα NP Y στο X σε πολυωνυμικό χρόνο.
NP-hard O ακριβής ορισμός εδώ είναι ότι ένα πρόβλημα X είναι NP-hard, αν υπάρχει NP- complete πρόβλημα Y, έτσι ώστε το Y είναι αναγώγιμο σε X σε πολυωνυμικό χρόνο.

3.Πρόβλημα NP-hard με καλο/γρήγορο 2-προσεγιστικό
Απάντηση
TSP ή Vertex Cover
http://ww3.algorithmdesign.net/handouts/Approximation.pdf
ok

Σεπτεμβρίου

Θεμα 1ο
a) εχουν ιδιους χρονους
b) συμφερει ο αυτος με βάση 3

Θεμα 2ο
A) Θα χρησιμοποιήσω τον αλγόριθμο του Kruskal με αλλαγή στην σειρά δηλαδή αντι για αύξουσα ταξινόμιση κάνω φθίνουσα.
Ο αλγόριθμος

```
KRUSKAL(G):
A = ∅
foreach v ∈ G.V:
    MAKE-SET(v)
foreach (u, v) in G.E ordered by weight(u, v), increasing:
    if FIND-SET(u) ≠ FIND-SET(v):
    A = A ∪ {(u, v)}
    UNION(u, v)
return A
```

B) Η απάντηση ειναι “ΟΧΙ” .Αν κάνω κάτι παρόμοιο δηλαδή χρησιμοποιήσω του sortest path αναποδα θα βρίσκω κάθε φορά το μεγαλύτερο μονοπάτι και δεν θα τελειώσει ποτέ ο αλγόριθμος.

Θεμα 3ο

Ειναι ο αλγόριθμος Strassen
Γράφω τον αλγόριθμο απο εδώ:
http://www.programering.com/a/MDM0UjNwATU.html
Και λέω και τα λογια απο διαφάνεια 14-16
https://alg12.wikischolars.columbia.edu/file/view/MASTER.pdf/295439724/MASTER.pdf

Θεμα 4ο
a) θεωρεία NP,P κλπ
b) λύση (σελ 112) : http://www.cs.cmu.edu/~avrim/451f11/lectures/lect1108.pdf
c) http://www.geeksforgeeks.org/bipartite-graph/
d)
λύση 1η: http://www3.cs.stonybrook.edu/~skiena/373/hw/keys/hw3.pdf
λύση2η:http://tristan-interview.blogspot.gr/2012/03/find-minimun-vertex-cover-for-tree.html
e)
Η λυση ειναι στο aproximation.pdf slide 6

Θεμα 5ο
Αλγόριθμος Merge-sort και Quick-sort

Οι διαφορές τους ειναι πως ο quicksort εχει χειροτερο χρόνο σε περίπτωση κακής επιλογής pivot οπου μπορει να εχει και εκεθετικο χρόνο n^2 σε αντιθεση με τον mergesort που εχει χρόνο o(nlogn).