ειμαι αρκετα σιγουρος ετσι νομιζω δεν εχω ιδεα το εβαλα στη τυχη send help

Μέρος Α'

μερικα μου τα βγαζει λαθος :/ (a=2 b=8 k=⅓)
Theorem Solver

1. Δεδομένου ενός μη κατευθυνόμενου γραφήματος G(V, E), είναι NP-πλήρες να αποφανθούμε αν το G έχει ανεξάρτητο σύνολο με τουλάχιστον 2020 κορυφές

○ Σωστό+++++++
Maximum Independent Set με k = 2020
είναι όντως σωστό (γτ αν λύσεις αυτο τότε στον συμπηρωματικό γράφο λύνεις το Clique → NP Complete)
○ Λάθος

2. Η αναδρομική σχέση T(n) = 4 T(n/2)+Θ(n), με T(1) = Θ(1), έχει λύση:

○ T(n) = Θ(n)
○ T(n) = Θ(n log(n))
○ T(n) = Θ(n^2)++++++++++++++++ +Master Theorem(Θ(n), a > b, Περ. 2), (είτε Wolfram χρχρ) τι βαζεις στο wolfram για να στο υπολογισει;
https://www.wolframalpha.com/input/?i=T%5Bn%5D%3D%3D2*T%5Bn%2F2%5D+%2B+nlogn ευχαριστω
το T(1) = Θ(1) πως το βαζουμε στο wolfram?επισης οταν βαζεις Θ(n) στο wolfram δεν σου βγαζει αποτελεσμα => βάζεις theta(n)
εμενα μου βγαίνει αλλο πραγμα όταν το βάζω στο wolfram μπορείτε να πείτε πως ακριβώς το γράφουμε γιατί του link μου βγάζει n(logn)^2 / 1 log(2) ???

3. Η αναδρομική σχέση T(n) = T(n/2)+T(n/3)+Θ(n log(n)), με T(1) = Θ(1), έχει λύση:

○ T(n) = Θ(n (log(n))^2)
○ T(n) = Θ(n log(n)) +++++++++++ ειδικη περιπτωση με τα γ1= ½ και γ2=⅓ και γ1+γ2<1 (αλλα η σχεση εχει ορο Θ(n log(n)) αντι για Θ(n))
○ T(n) = Θ(n)

4. Το DFS σε ένα γράφημα με n κορυφές χρειάζεται χρόνο Θ(n^2) αν το γράφημα αναπαρίσταται με πίνακα γειτνίασης.

○ Λάθος +
○ Σωστό ++++++++

5. Στην υλοποίηση της δομής δεδομένων Union-Find με βεβαρημένη ένωση, κάθε λειτουργία έχει χρόνο εκτέλεσης χειρότερης περίπτωσης O(log(n)).

○ Σωστό+++++++ (και οι δύο πράξεις δεν είναι O(log(n));;;) → το Union γίνεται σε O(1)

○ Λάθος++++ +++++ τη Union στις διαφάνειες την εχει O(1) (θεωρει ο Φωτακης οτι στη union βαζουμε τους αντιπροσωπους.. εκτος και αν ειναι ερωτηση παγιδα)
(και πάλι όμως, κάτι που είναι O(1) είναι και O(logn))
AMA KATI EINAI O(1) TOTE EINAI O(LOGN)!!!! ARA SWSTO?++++
ειναι O(1) και στη χειρότερη περίπτωση αρα οχι O(logn)!!
re paidia to union find me path reduction, to find to kanei se loglogn, den to kanei se 0(1)
lathos nomizw kai egw => δεν ειναι θεμα το find αλλα το union που γινεται σιγουρα σε O(1)

algorithm	order of growth for N sites (worst case)		
	constructor	union	find
quick-find	N	N	1
quick-union	N	*tree height*	*tree height*
weighted quick-union	N	$\lg N$	$\lg N$
weighted quick-union with path compresson	N	*very, very nearly, but not quite* 1 (*amortized*) (see EXERCISE 1.5.13)	
impossible	N	1	1

https://algs4.cs.princeton.edu/15uf/

Σε κάθε περίπτωση, αν ειναι O(1) ειναι και O(logn) και του O(γουατεβερ) (είναι άνω φραγμα)++-

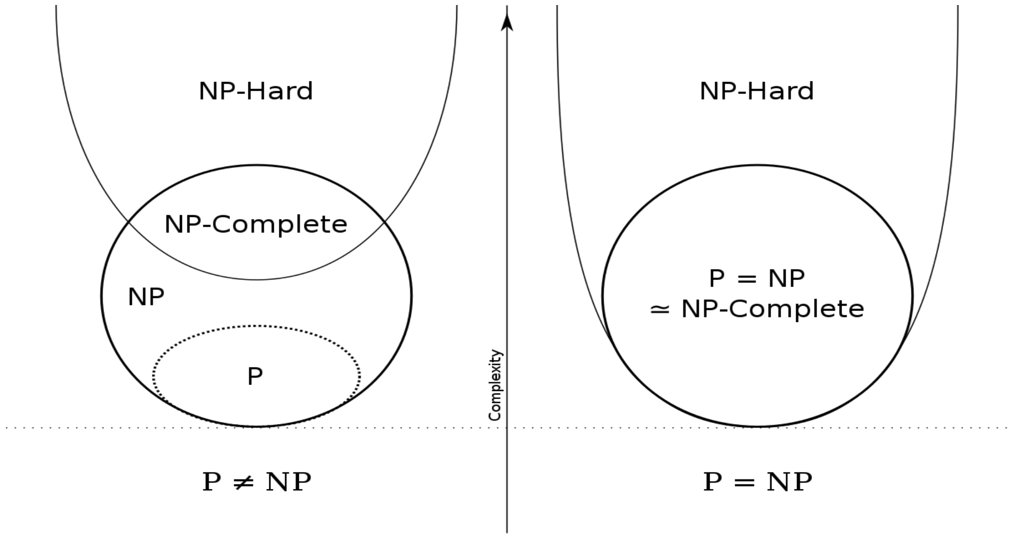
6. Ο αλγόριθμος δυναμικού προγραμματισμού για το πρόβλημα του Πλανόδιου Πωλητή έχει ψευδο πολυωνυμικό χρόνο εκτέλεσης.
○ Σωστό
○ Λάθος+++++++ O(n^2 * 2^n) → εκθετικό

7. Σε ένα s-t δίκτυο, αν αυξήσουμε τη χωρητικότητα δύο ακμών της μέγιστης τομής κατά k, τότε η μέγιστη ροή αυξάνεται κατά 2k.
○ Σωστό
○ Λάθος ++++++++(δεν εξαρτάται;-> για να αυξήσουμε τη ροή πρέπει να δούμε το μιν)
Μέγιστη ροή = Ελάχιστη τομή αρα αν αυξήσουμε τη μέγιστη τομή η Ελάχιστη τομή δεν επηρεαζεται

8. Η αναδρομική σχέση T(n) = T(n^{1/2})+Θ(1), με T(1) = Θ(1), έχει λύση:
○ T(n) = Θ(log(log(n)))++++++
Find theta of: T(n) = T(n^(1/2)) + 1
○ T(n) = Θ(n)
○ T(n) = Θ(log(n))

9. Αν οι κλάσεις P και NP είναι διαφορετικές, τότε κάθε πρόβλημα που δεν ανήκει στο P είναι NP-πλήρες.

○ Λάθος +++++++++
○ Σωστό
(Μπορεί να είναι και εκτός NP ή να είναι ενδιάμεσο)



10. Ένα κατευθυνόμενο ακυκλικό γράφημα έχει μοναδική τοπολογική διάταξη αν και μόνο αν υπάρχει μοναδική κατευθυνόμενη ακμή μεταξύ κάθε ζεύγους κορυφών (χωρίς να δημιουργείται κύκλος).

○ Λάθος ++++++++++
αντιπαράδειγμα: α->β, β->γ. εχει μοναδικη τοπολογική διάταξη χωρις να υπαρχει η ακμη α->γ.
******* Ισως να μην ειναι μοναδικη.. μπορεις να τα βαλεις με σειρα α, β, γ αλλα και β, α, γ!!
○ Σωστό + + [Μονοπάτι Hamilton](δεν λεει οτι δημιουργείται μοναδικο τοπολοτζικαλ στο wiki?)

an kathe zeugos korufwn exei monadikh akmh metaksu tou kai den exw kuklous tote kserw thn sxesh gia topological ordering metaksu olwn ton korufwn opote borw na exw monadiko topological ordering afou einai kathorismenes oles oi sxeseis metaksu korufwn ara swsto(??)
δεν ισχύει το “αν και μόνο αν”, γιατί μπορεί να έχεις μοναδική τοπολογική διάταξη χωρίς να έχεις ακμή μεταξύ κάθε ζεύγους κορυφών +++++
υπάρχει παράδειγμα για αυτό; (το ακριβώς από πάνω)
--νομιζω ειναι λαθος γτ η λεξη μοναδικη δεν ισχυει , αφου το DAG προκυπτει απο BFS και καποια BFS μπορει να διαφερουν μεταξυ τους αναλογα με το αν πηγαμε δεξια ή αριστερά πρώτα (με καθε επιφυλαξη)--
-https://en.wikipedia.org/wiki/Topological_sorting#Uniqueness (σωστο?) μάλλον...

11. (8 μονάδες) Έχουμε έναν κορμό δέντρου μήκους L > 0 (για ευκολία, μπορείτε να θεωρήσετε ότι ο κορμός ταυτίζεται με το διάστημα [0, L] στην πραγματική ευθεία). Έχουμε σημειώσει n θέσεις x1, …, xn, με 0 < x1 < … < xn < L, όπου θα κόψουμε τον κορμό, ώστε να προκύψουν n+1 κομμάτια ξύλου που αντιστοιχούν στα διαστήματα [0, x1), [x1, x2), …, [xn, L]. Το κόψιμο ενός κομματιού μήκους M απαιτεί M μονάδες ενέργειας.

Θέλουμε να υπολογίσουμε τη σειρά με την οποία πρέπει να γίνουν τα n κοψίματα, ώστε να ελαχιστοποιηθεί η συνολική ενέργεια που απαιτείται.
Να απαντήσετε στα παρακάτω ερωτήματα σχετικά με το παραπάνω πρόβλημα.

1 (4 μονάδες). Να διατυπώσετε μια αναδρομική σχέση από την οποία προκύπτει η ελάχιστη ενέργεια που απαιτείται για n κοψίματα, αν αυτά γίνουν με τη βέλτιστη σειρά;

C(i, j) = ελάχιστο κόστος για το κόψιμο του τμήματος από x_i μέχρι x_j,
0 <= i < j <= n+1,   (x_0 = 0 και x_{n+1} = L)

C(i, j) = cost(i, j) + min_{i < k < j} { C(i, k) + C(k, j) }
C(i, j) = (x_j - x_i) + min_{i < k < j} { C(i, k) + C(k, j) }, 0 <= i < j <= n+1
C(i, i+1) = 0

answer: C(0, n+1) = L + min_{0 < k < n+1} { C(0, k) + C(k, n+1) }
~~~~~~

```
Cut(i,j):                    //θεωρω x[i] = xi οπου x[0] = 0 και x[n+1] = L
        if i-j=1 return 0
        else return x[j]-x[i] + min_{i< k <j}(Cut(i,k) + Cut(k,j))
answer: Cut(0,n+1)
```

(Η ΑΠΑΝΤΗΣΗ ΕΙΝΑΙ ΟΛΗ Η ΑΠΟ ΠΑΝΩ Η΄ ΜΕΧΡΙ ΤΗ ΓΡΑΜΜΗ???) : ειναι δυο λυσεις απο δυο ατομα. τωρα το ποια ειναι πιο σωστη ¯_(ツ)_/¯

//ολος ο κορμος ειναι [0,n+1]. το καθε Cut(i,j) ειναι το ελαχιστο κοστος για να κοψουμε σε οποιοδήποτε σημείο το κομματι του κορμου [xi,xj)
//το οποιο θα ειναι το μηκος του κομματιου αυτου (x[j]-x[i]) + ελαχιστο κοστος για καθε κομματι που δημιουργηθηκε (αν πχ το ελαχιστο αυτο κοψιμο ηταν στο σημειο k: [xi,xk) και [xk,xj))
//η λυση του προβληματος ειναι Cut(0,n+1)

2 (2 μονάδες). Με ποια αλγοριθμική τεχνική θα υπολογίσετε την ελάχιστη ενέργεια που απαιτείται για n κοψίματα, με βάση την αναδρομική σχέση του (1);

Dynamic Programming αποθηκεύοντας τις προηγούμενες τιμές σε DP Array+

3. (2 μονάδες) Ποιος είναι ο χρόνος (ως συνάρτηση του n) που απαιτείται για τον υπολογισμό της ελάχιστης ενέργειας, με βάση την αναδρομική σχέση του (1) και την αλγοριθμική τεχνική του (2);

O(n^3) ++++
~~~~~~

παρομοια αναδρομη με το matrix-chain multiplication. O(state space επι το min) ++

Παράδειγμα (για την κατανόηση του ορισμού του προβλήματος):
Έστω L = 10, n = 3, x1 = 3, x2 = 5, και x3 = 7. Αν κόψουμε πρώτα στο 5, μετά στο 3, και τέλος στο 7, δαπανούμε 10+5+5=20 μονάδες ενέργειας. Αν κόψουμε πρώτα στο 3, μετά στο 5, και τέλος στο 7, δαπανούμε 10+7+5=22 μονάδες ενέργειας. Αν κόψουμε πρώτα στο 3, μετά στο 7, και τέλος στο 5, δαπανούμε 10+7+4=21 μονάδες ενέργειας.

12. (4 μονάδες) Ποια είναι η υπολογιστική πολυπλοκότητα (ως συνάρτηση του n) του καλύτερου αλγόριθμου που γνωρίζετε (ή μπορείτε να σκεφτείτε) για τον υπολογισμό των συντομότερων μονοπατιών μεταξύ όλων των ζευγών κορυφών ενός κατευθυνόμενου γραφήματος με n κορυφές και n(log(n))^6 ακμές, όταν κάποιες ακμές μπορούν να έχουν αρνητικό μήκος.

Johnson's algorithm με E = V(log(V))^6: O(V^2*(log(V))^6 + V^2*log(V)) =O(V^2*(logV)^6) ++++

https://en.wikipedia.org/wiki/Johnson%27s_algorithm
και
https://courses.corelab.ntua.gr/pluginfile.php/6548/course/section/791/15_ShortestPaths.pdf
σελ 34

13. Δίνεται ένα μη κατευθυνόμενο συνεκτικό γράφημα G με θετικά βάρη στις ακμές. Υπάρχει αλγόριθμος γραμμικού χρόνου για τον υπολογισμό ενός Συνδετικού Δέντρου του G στο οποίο η βαρύτερη ακμή έχει το ελάχιστο δυνατό βάρος.
○ Λάθος
○ Σωστό ++++ το έχει στην 3η σειρά προτεινόμενων ασκήσεων, ασκηση 7!!!
https://stackoverflow.com/questions/33531475/linear-time-algorithm-for-mst
me vash auto swsto?

https://centralntua.webex.com/recordingservice/sites/centralntua/recording/31f60103aef1400882ffa56ebab84683/playback
στο 1:05:50 το λεει και ο Φωτακης (οτι λυνεται γραμμικα)

14. Δεδομένων ενός μη κατευθυνόμενου γραφήματος G(V, E) και ενός φυσικού k >= 3, είναι NP-πλήρες να αποφανθούμε αν το G έχει (απλό) κύκλο μήκους τουλάχιστον k.
○ Σωστό+ +++++
για k=n το πρόβλημα ανάγεται στο Hamiltonian → NP Complete
an auto anagetai se hamiltonian den shmainei oti einai np-complete
prepei to hamiltonian na anagetai se auto arA?
○ Λάθος++ υπάρχει αιτιολόγηση? με DFS αν υπάρχει back edge

αν κανουμε
 και καθε φορα που βρισκουμε ενα κυκλο να αφαιρουμε τα υψη των δυο κομβων μεχρι να βρουμε >=k ? ετσι θα εβγαινε γραμμικος χρονος

15. Ο χρόνος εκτέλεσης του αλγορίθμου ταξινόμησης Counting Sort είναι πολυωνυμικός στο μέγεθος της εισόδου.
○ Λάθος++++++++++++ (εφόσον είναι O(n+r), ψευδοπολυωνυμικός δηλαδή, δεν είναι λάθος;)++
○ Σωστό++++
Το μεγεθος της εισοδου ειναι nlogk οπου k το μεγεθος bit των ψηφιων => πολυωνυμικος στο πληθος των στοιχειων ΑΛΛΑ εκθετικος στο πληθος bit => σιγουρα οχι πολυωνυμικος, νο;
++++
https://www.geeksforgeeks.org/counting-sort/
Time Complexity: O(n+k) where n is the number of elements in input array and k is the range of input. =>swsto , δεν ειναι σωστο, εαν προσθεσεις ενα παραπανω bit διπλασιαζεται ο χρονος σου, επηρεαζεται εκθετικα
https://stackoverflow.com/questions/19647658/what-is-pseudopolynomial-time-how-does-it-differ-from-polynomial-time

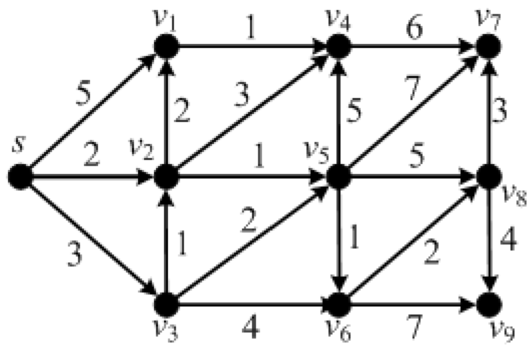
That said, in many cases pseudopolynomial time algorithms are perfectly fine because the size of the numbers won't be too large. For example, counting sort has runtime O(n + U), where U is the largest number in the array. This is pseudopolynomial time (because the numeric value of U requires O(log U) bits to write out, so the runtime is exponential in the input size). If we artificially constrain U so that U isn't too large (say, if we let U be 2), then the runtime is O(n), which actually is polynomial time. This is how radix sort works: by processing the numbers one bit at a time, the runtime of each round is O(n), so the overall runtime is O(n log U). This actually *is* polynomial time, because writing out n numbers to sort uses Ω(n) bits and the value of log U is directly proportional to the number of bits required to write out the maximum value in the array.

Δυναμικός Προγραμματισμός σελ 12
++
https://centralntua.webex.com/recordingservice/sites/centralntua/recording/c44b90f8cb674d0ca18b131ea2c00d9a/playback (44:48 δεν είναι πολυωνυμικός σύμφωνα με φωτάκη)+

Μέρος Β’

1. Έστω κατευθυνόμενο γράφημα G(V, E, w) με θετικά μήκη w στις ακμές, και έστω p ένα συντομότερο u – v μονοπάτι στο G. Το p παραμένει συντομότερο u – v μονοπάτι αν υψώσουμε τα μήκη όλων των ακμών στο τετράγωνο.
○ Λάθος++++++++++++ εχει πλεονεκτημα το μακρυτερο μονοπατι και οχι το βαρυτερο
πχ α->γ με 3, α->β με 2, β->γ με 2, θα γινει α->γ με 9, α->β με 4, β->γ με 4
○ Σωστό

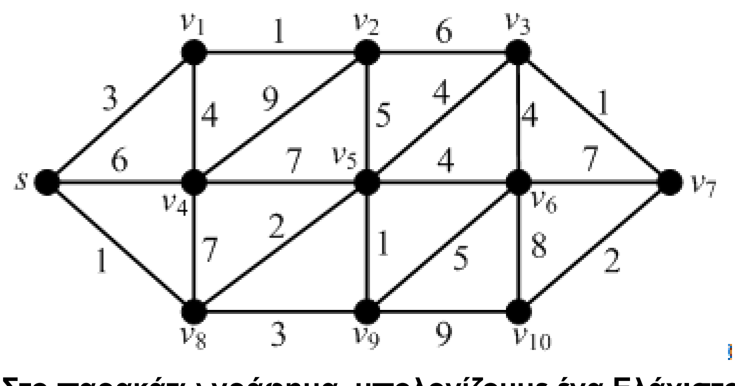
2.



Στο παραπάνω γράφημα, υπολογίζουμε τις αποστάσεις και τα συντομότερα μονοπάτια όλων των κορυφών από την s χρησιμοποιώντας τον αλγόριθμο του Dijkstra. Ποια (μία μόνο) από τις παρακάτω προτάσεις είναι αληθής:

○ Υπάρχουν δύο συντομότερα μονοπάτια από την κορυφή s στην κορυφή v5.
○ Η κορυφή v4 αποκτά μόνιμη ετικέτα μετά την κορυφή v6.++++++++
○ Η κορυφή v7 είναι η τελευταία που αποκτά μόνιμη ετικέτα.
○ Η κορυφή v1 μπορεί να είναι η 4η κορυφή που αποκτά μόνιμη ετικέτα ++ --
(η 1η κορυφή που αποκτά μόνιμη ετικέτα είναι η κορυφή s και η 2η είναι η κορυφή v2, 3η η v3, 4η η v5 , 5η η v1).

3.



Στο παρακάτω γράφημα, υπολογίζουμε ένα Ελάχιστο Συνδετικό Δέντρο χρησιμοποιώντας τον αλγόριθμο του Kruskal με αρχική κορυφή την s. Ποια (μία μόνο) από τις παρακάτω προτάσεις είναι αληθής:

○ Η ακμή (v7, v10) μπορεί να είναι η τελευταία ακμή που προστίθεται στο Ελάχιστο Συνδετικό Δέντρο από τον αλγόριθμο.
○ Όλες οι ακμές βάρους 4 ανήκουν στο Ελάχιστο Συνδετικό Δέντρο.
○ Η ακμή (s, v1) μπορεί να είναι η 7η ακμή που προστίθεται στο Ελάχιστο Συνδετικό Δέντρο από τον αλγόριθμο.++++++++++
○ Όλες οι ακμές βάρους 3 ανήκουν στο Ελάχιστο Συνδετικό Δέντρο.

4. (4 μονάδες) Θεωρούμε απλό συνεκτικό μη κατευθυνόμενο γράφημα G(V, E, w), με n κορυφές, m ακμές και θετικό βάρος w(e) σε κάθε ακμή e. Αξιολογούμε τα μονοπάτια μεταξύ δύο κορυφών u και v με βάση το μήκος της βαρύτερης ακμής τους. Το λεγόμενο bottleneck κόστος c(p) ενός u - v μονοπατιού p είναι c(p) = max_{e ανήκει στο p} { w(e) }. Το ζητούμενο είναι να υπολογίσουμε ένα (όσο το δυνατόν πιο αραιό) συνδετικό (spanning) υπογράφημα H του G το οποίο για κάθε ζευγάρι διαφορετικών κορυφών u και v, περιέχει κάποιο u - v μονοπάτι ελάχιστου bottleneck κόστους (ως προς όλα τα u - v μονοπάτια στο G).

Να απαντήσετε στα παρακάτω ερωτήματα σχετικά με το παραπάνω πρόβλημα.

1 (2 μονάδες). Πόσες είναι οι ακμές που απαιτείται να έχει ένα τέτοιο γράφημα H (ως συνάρτηση των n και m); Αν το πλήθος των ακμών του H μπορεί να ποικίλει, να δώσετε ένα άνω και ένα κάτω φράγμα στο πλήθος των ακμών του H (ως συνάρτηση των n και m).

~~~~~~

Για να παραμένει συνδετικό, σίγουρα το πλήθος των ακμών του H>=n-1.
Ωστόσο μπορεί να χρησιμοποιηθούν σχεδόν όλες οι ακμές. Οπότε πιστεύω H<=m

για το 1ο ερώτημα θα έλεγα πλήθος ακμών ακριβώς n-1 (-+-)
prepei na exw mst gia na exw minimum bottleneck ,ara thelw akrivos n-1 akmes..an exw mia parapanw tha auksithei to botlleneck kostos se kapoio path

~~~~~~

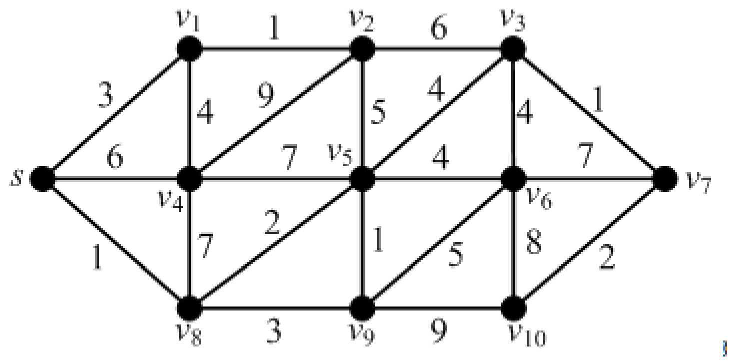
2 (2 μονάδες). Ποια είναι η υπολογιστική πολυπλοκότητα (ως συνάρτηση των n και m) του καλύτερου αλγορίθμου που γνωρίζετε (ή μπορείτε να σκεφτείτε) για τον υπολογισμό ενός τέτοιου υπογραφήματος H που περιέχει τα συντομότερα bottleneck μονοπάτια για όλα τα ζεύγη κορυφών του G;

~~~~~~`

Α' τρόπος προσέγγισης
Για τον υπολογισμό ενός bottleneck Μονοπατιού θέλουμε Θ(n). Άρα παίρνοντας για όλα τα ζεύγη αυτών κάνουμε Θ(n^3). Ωστόσο στο ίντερνετ υπάρχουν διάφορα papers που το κάνουν σε χρόνο κάτω από το κυβικό.
(Αλλά φαντάζομαι κάτι είχε συζητήσει εκείνη την χρονιά που το έβαλε επειδή δεν υπάρχει τίποτα παρόμοιο.)
~~~~~~

Στις διαλέξεις λέει πως το MST είναι η απάντηση στο Bottleneck Shortest Path για ολα τα ζευγη κορυφών+++. Προτεινόμενες Ασκήσεις 3η Σειρα Ασκηση 5
Οπότε:
O(m + nlogn) → Prim με Fibonacci Heaps(ή boruvka me prim O(mloglogn))

5.



Στο παρακάτω γράφημα, υπολογίζουμε ένα Ελάχιστο Συνδετικό Δέντρο χρησιμοποιώντας τον αλγόριθμο του Prim με αρχική κορυφή την s. Ποια (μία μόνο) από τις παρακάτω προτάσεις είναι αληθής:

○ Η ακμή (v8, v9) μπορεί να είναι η 4η ακμή που προστίθεται στο Ελάχιστο Συνδετικό Δέντρο από τον αλγόριθμο.

○ Η ακμή (s, v1) μπορεί να είναι η 2η ακμή που προστίθεται στο Ελάχιστο Συνδετικό Δέντρο από τον αλγόριθμο.

○ Η ακμή (v1, v4) μπορεί να είναι η 4η ακμή που προστίθεται στο Ελάχιστο Συνδετικό Δέντρο από τον αλγόριθμο.

○ Η ακμή (v3, v7) μπορεί να είναι η 7η ακμή που προστίθεται στο Ελάχιστο Συνδετικό Δέντρο από τον αλγόριθμο.++++++++++

6.Δίνονται 4 πίνακες, ο καθένας από τους οποίους έχει n στοιχεία και είναι ταξινομημένος σε αύξουσα σειρά. Θέλουμε να ταξινομήσουμε τα 4n στοιχείων που προκύπτουν από την ένωση των 4 πινάκων. Ο καλύτερος συγκριτικός αλγόριθμος για αυτό το πρόβλημα χρειάζεται χρόνο Θ(n log(n)).

○ Σωστό ++++- (νομιζω σωστό - C/C++ Program for Merge k sorted arrays | Set 1) του βγαινει Θ(n log(n)) γιατι ο τυπας το κανει με MinHeap για καποιο λογο

○ Λάθος+++++++++++++ (είναι ήδη ταξινομημένοι => Merge πολυπλοκότητας Θ(4n) = Θ(n)) Πηγαίντε εκεί που λέει running time (εδώ έχουμε k=4 άρα O(nlog4) = O(n))

7. (3 μονάδες) Έστω δύο πίνακες A[1..n] και B[1..n]. Ο καθένας τους περιέχει n ακέραιους αριθμούς ταξινομημένους σε αύξουσα σειρά. Ποια είναι η υπολογιστική πολυπλοκότητα (ως συνάρτηση του n) του καλύτερου αλγόριθμου που γνωρίζετε (ή μπορείτε να σκεφτείτε) για τον υπολογισμό του ενδιάμεσου (median) στοιχείου της ένωσης των δύο πινάκων (δηλ. το ενδιάμεσο στοιχείο είναι το στοιχείο που θα καταλάμβανε τη θέση n στον ταξινομημένο πίνακα που περιέχει όλα τα στοιχεία των A και B);

O(logn),+ [διάλεξη 9η] στο 58’ περίπου
~~~~~~
For k-sorted arrays, we find the median in O(klog(kn))

https://www.geeksforgeeks.org/median-of-two-sorted-arrays/

https://cs.stackexchange.com/questions/87695/to-find-median-of-k-sorted-arrays-of-n-elements-each-in-less-than-onklogk
~~~~~~

8.Μπορούμε να υπολογίσουμε τα n/100 μεγαλύτερα στοιχεία ενός (μη ταξινομημένου) πίνακα με n στοιχεία σε χρόνο O(n) στη χειρότερη περίπτωση.

○ Σωστό ++++++++++ο Ντετερμινιστικός αλγόριθμος QuickSelect (αυτος με τις 5δες) ειναι γραμμικος στη χειρότερη περίπτωση: [Επιλογή] σελ 17
https://www.geeksforgeeks.org/kth-smallestlargest-element-unsorted-array-set-3-worst-case-linear-time/
με αυτό η quickselect γινεται στη χειρότερη γραμμική
https://en.wikipedia.org/wiki/Median_of_medians
οπότε είναι σωστό

Αυτό δεν αρκει, στην ουσία το προβλημα είναι οτι θελει τα n/100 μεγαλύτερα στοιχεία του πινακα.
Οπότε βρίσκεις το (n/100)στο στοιχείο (με οποια quickselect θελουμε, ας σπουμε σε O(n)) και μετα διατρεχουμε τον πινακα και κραταμε ολα τα στοιχεια που ειναι μεγαλύτερα από αυτον τον αριθμό
Συνολο O(n+n) = O(n)

○ Λάθος++ + - - -(παραλλαγη της QuickSelect => χειρότερη περίπτωση n^2) <- ο πιθανοτικος ειναι n^2 στη χ.π.

9. (2 μονάδες) Δίνεται συνεκτικό απλό μη-κατευθυνόμενο γράφημα G(V, E, w), με n κορυφές, m ακμές και θετικό βάρος w(e) σε κάθε ακμή e. Θεωρούμε ότι όλα τα βάρη των ακμών είναι διαφορετικά μεταξύ τους. Ποια είναι η υπολογιστική πολυπλοκότητα (ως συνάρτηση των n και m) του καλύτερου αλγόριθμου που γνωρίζετε (ή μπορείτε να σκεφθείτε) για τον υπολογισμό του συνδετικού δέντρου του G με το δεύτερο μικρότερο συνολικό βάρος;
O(mlogn)+++++
(https://cp-algorithms.com/graph/second_best_mst.html)
(Παρόμοιο με 3η σειρά Γραπτών 6γ Bonus)

10.Δίνονται μη κατευθυνόμενο γράφημα G(V, E) και θετικός

φυσικός k. Είναι NP-πλήρες να αποφανθούμε αν το G έχει συνδετικό δέντρο με μέγιστο βαθμό κορυφής μεγαλύτερο ή ίσο του k.

○ Λάθος +++ (δεν αρκει να βρουμε κορυφη με βαθμο τουλαχιστον k?)

○ Σωστό ++++++
(https://en.wikipedia.org/wiki/Degree-constrained_spanning_tree) εδω ομως λεει >=k

11. Αν στον αλγόριθμο του Dijkstra, επιλέγουμε τη διαθέσιμη κορυφή με τη μέγιστη πεπερασμένη ετικέτα σε κάθε επανάληψη, τότε υπολογίζουμε τα μονοπάτια μέγιστου μήκους από την αρχική κορυφή s προς κάθε άλλη κορυφή.

○ Σωστό

○ Λάθος+++++++

In graph theory and theoretical computer science, the **longest path problem** is the problem of finding a simple path of maximum length in a given graph. A path is called simple if it does not have any repeated vertices; the length of a path may either be measured by its number of edges, or (in weighted graphs) by the sum of the weights of its edges. In contrast to the shortest path problem, which can be solved in polynomial time in graphs without negative-weight cycles, the longest path problem is NP-hard and the decision version of the problem, which asks whether a path exists of at least some given length, is NP-complete. This means that the decision problem cannot be solved in polynomial time for arbitrary graphs unless P = NP. Stronger hardness results are also known showing that it is difficult to approximate. However, it has a linear time solution for directed acyclic graphs, which has important applications in finding the critical path in scheduling problems.

12. Ο αλγόριθμος του Dijkstra και ο αλγόριθμος του Prim έχουν (ασυμπτωτικά) τον ίδιο χρόνο εκτέλεσης χειρότερης περίπτωσης.

○ Σωστό+++++++++++ (ο Prim πρακτικα ειναι Dijkstra με διαφορετικο κριτηριο)

○ Λάθος++++ (γιατι ???)

13.Δίνονται συνεκτικό μη κατευθυνόμενο γράφημα G(V, E, w), με θετικό βάρος w(e) σε κάθε ακμή e, αύξουσα συνάρτηση f : N -> N και θετικός φυσικός B. Είναι NP-πλήρες να αποφανθούμε αν υπάρχει συνδετικό δέντρο T του G για το οποίο το άθροισμα των τιμών f(w(e)), για τις ακμές e που ανήκουν στο T, είναι μικρότερο ή ίσο του B.

○ Λάθος ++++++
Εφοσων η f() ειναι αυξουσα, το Σf(w(e)) (οπου τα e ειναι οι ακμες το MST) θα ειναι το ελαχιστο δυνατο. Αρα αρκει να υπολογισω MST που δεν ειναι NP-complete.+

○ Σωστό ++
(How do you determine whether a graph G has a spanning tree of weight k?) Αυτό είναι για ακριβώς Κ.

14.Η βέλτιστη λύση στη διακριτή εκδοχή του προβλήματος του Σακιδίου (Knapsack) περιέχει πάντα το αντικείμενο με τον μέγιστο λόγο αξίας προς μέγεθος.

○ Σωστό

○ Λάθος++++++++

15.(2 μονάδες) Ποια είναι η υπολογιστική πολυπλοκότητα (ως

συνάρτηση των n και m) του καλύτερου αλγόριθμου που γνωρίζετε (ή μπορείτε να σκεφθείτε) για τον υπολογισμό των ισχυρά συνεκτικών συνιστωσών ενός απλού κατευθυνόμενου γραφήματος με n κορυφές και m ακμές;
O(V + E) ? +++ (Tarjan's algorithm) +++++
Αλγόριθμος Kosaraju σε γραμμικο χρόνο

16. Σε ένα s-t δίκτυο, η ροή που διέρχεται από μία συγκεκριμένη ακμή μπορεί να μειωθεί κατά την εξέλιξη του αλγόριθμου των Ford-Fulkerson.
○ Σωστό+++++++ apo tis backward edges afairoume roh, prosthetoume stis forward
Ford-Fulkerson in 5 minutes — Step by step example
○ Λάθος +T[n/2]

17.(2 μονάδες) Θεωρούμε ένα σύνολο S με n σημεία στο Ευκλείδειο επίπεδο (κάθε σημείο i δίνεται ως το διατεταγμένο ζεύγος (xi, yi) των συντεταγμένων του). Ποια είναι η υπολογιστική πολυπλοκότητα (ως συνάρτηση του n) του καλύτερου αλγόριθμου που γνωρίζετε (ή μπορείτε να σκεφθείτε) για τον υπολογισμό των δύο πλησιέστερων σημείων του S
O(n logn)++++++ ++-
Closest Pair of Points using Divide and Conquer algorithm
to exei kai stis diafaneies tou mathmatos Closest pair Kleinberg and Tardos

Notes

1) Time complexity can be improved to O(nLogn) by optimizing step 5 of the above algorithm. We will soon be discussing the optimized solution in a separate post.