Δυναμικός προγραμματισμός
1η εργαστηριακή άσκηση

Ομάδα:

Αποστολίδου Μαρία (2012030114)

Κυπαρισσάς Νικόλαος (2012030112)

Μαραγκού Σοφία (20120300078)

Ζητούμενα: Το πρόβλημα που μας ζητήθηκε να επιλύσουμε με Δυναμικό Προγραμματισμό είναι η εύρεση της συντομότατης διαδρομής (από την αφετηρία στον τερματισμό) σε γράφους με την μορφή συστημάτων αποφάσεων πολλαπλών βαθμίδων. Η πρώτη (σημείο εκκίνησης) και η τελευταία (σημείο τερματισμού) βαθμίδα αποτελούνται από έναν κόμβο.

Υλοποίηση: Το πρόγραμμα αρχικά δέχεται σαν είσοδο, το πλήθος των βαθμίδων, το πλήθος των κόμβων ανά βαθμίδα, και το κόστος άμεσης μετάβασης από κόμβο σε κόμβο. Κατά την είσοδο στο σύστημα δημιουργούνται οι εξής πίνακες:

- B - πίνακας βαθμίδων, αποθηκεύει το πλήθος των κόμβων ανά βαθμίδα. Έχει μέγεθος: πλήθος βαθμίδων * 1.
- G -πίνακας γράφου, αποθηκεύει την αρίθμηση/ονομασία των γράφων. Έχει μέγεθος: πλήθος βαθμίδων*max(πλήθος κόμβων ανά γράφο). Έχει την τιμή -1 όπου δεν υπάρχει κόμβος.
- Cost -πίνακας κόστους άμεσης μετάβασης. Έχει μέγεθος: πλήθος κόμβων * πλήθος κόμβων. Έχει την τιμή -1 όπου δεν υπάρχει άμεση μετάβαση ανάμεσα στους κόμβους.

Κατά την επίλυση του προβλήματος δημιουργούνται οι παρακάτω πίνακες:

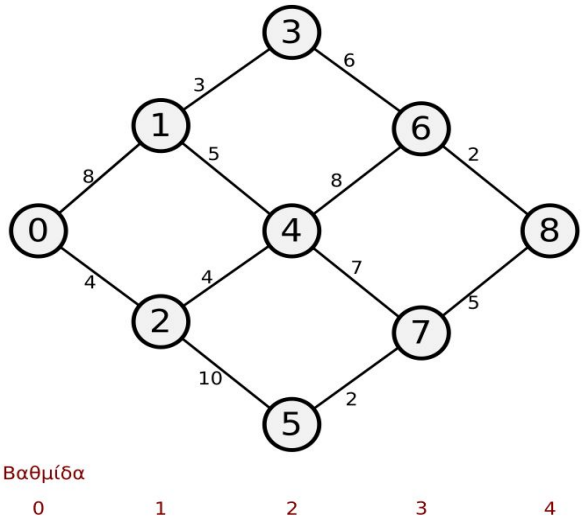
- OPT - πίνακας επίλυσης δυναμικού προγραμματισμού. Στον πίνακα αυτό αποθηκεύεται το κόστος διαδρομής. Σε κάθε κελί OPT[i,j] αποθηκεύεται το κόστος διαδρομής για να μεταβούμε από τον κόμβο i στον τελικό κόμβο μέσω του κόμβου j. Σύμφωνα με τις αρχές του δυναμικού προγραμματισμού το κόστος αυτό

υπολογίζεται ως το κόστος άμεσης μετάβασης από τον τρέχοντα κόμβο i στον κόμβο j συν τη βέλτιστη λύση του προβλήματος από τον κόμβο j και κάτω, δηλαδή το βέλτιστο κόστος διαδρομής από τον κόμβο j στον τελικό κόμβο (που έχει ήδη υπολογιστεί). (Στην τελευταία στήλη του πίνακα, αποθηκεύεται το βέλτιστο κόστος διαδρομής μέχρι τον τελικό κόμβο (το min της γραμμής)). Στο κάτω διαγώνιο μέρος του πίνακα, υπάρχει μόνο η τιμή -1, αφού δεν επιτρέπεται οπισθοδρόμηση. Τα κελιά που δεν υποδηλώνουν άμεση μετάβαση επίσης παίρνουν την τιμή -1 και η διαγώνιος έχει την τιμή 0, αφού στη συγκεκριμένη άσκηση δεν υπάρχει μετάβαση από έναν κόμβο στον εαυτό του ή, γενικότερα, το κόστος μετάβασης από έναν κόμβο στον εαυτό του είναι μηδενικό. Η λύση του συνολικού προβλήματος αποθηκεύεται στο πρώτο κελί της τελευταίας στήλης. Έχει μέγεθος: πλήθος κόμβων * πλήθος κόμβων.

- NextNode - σε αυτόν τον πίνακα αποθηκεύεται η τιμή του j για την οποία βρέθηκε η ελάχιστη τιμή. Στον πίνακα αποθηκεύουμε σε ποιόν κόμβο της επόμενης βαθμίδας πρέπει να μεταβούμε για να κάνουμε την βέλτιστη διαδρομή. Εν ολίγοις, ποιος κόμβος έδωσε το ελάχιστον κόστος. Έχει μέγεθος: πλήθος κόμβων * 1.

Ο αλγόριθμος που υλοποιήθηκε βρίσκει ένα βέλτιστο κανόνα ελέγχου και συγκεκριμένα βέλτιστο έλεγχο κλειστού βρόγχου και όχι απλά μια βέλτιστη τροχιά.

Σαν παράδειγμα δίνεται η λύση του προβλήματος του παρακάτω γράφου:



Οι πίνακες που δημιουργούνται μετά την είσοδο του συστήματος:

B (πίνακας βαθμίδων)					
βαθμίδα	**0**	**1**	**2**	**3**	**4**
# κόμβων	1	2	3	2	1

G (πίνακας γράφου)				
0	**1**	**3**	**6**	**8**
-1	**2**	**4**	**7**	-1
-1	-1	**5**	-1	-1

Cost (πίνακας κόστους άμεσης μετάβασης)									
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
0	**0**	8	4	-1	-1	-1	-1	-1	-1
1	-1	**0**	-1	3	5	-1	-1	-1	-1
2	-1	-1	**0**	-1	4	10	-1	-1	-1
3	-1	-1	-1	**0**	-1	-1	6	-1	-1
4	-1	-1	-1	-1	**0**	-1	8	7	-1
5	-1	-1	-1	-1	-1	**0**	-1	2	-1
6	-1	-1	-1	-1	-1	-1	**0**	-1	2
7	-1	-1	-1	-1	-1	-1	-1	**0**	5
8	-1	-1	-1	-1	-1	-1	-1	-1	**0**

(δεν επιτρέπεται οπισθοδρόμηση)
(δεν υπάρχει άμεση μετάβαση)

Ενώ οι πίνακες που δημιουργήθηκαν κατά την επίλυση του προβλήματος:

OPT (πίνακας επίλυσης δυναμικού προγραμματισμού)

	0	1	2	3	4	5	6	7	8
0	0	19	18	-1	-1	-1	-1	-1	18
1	-1	0	-1	11	14	-1	-1	-1	11
2	-1	-1	0	-1	14	17	-1	-1	14
3	-1	-1	-1	0	-1	-1	8	-1	8
4	-1	-1	-1	-1	0	-1	10	12	10
5	-1	-1	-1	-1	-1	0	-1	7	7
6	-1	-1	-1	-1	-1	-1	0	-1	2
7	-1	-1	-1	-1	-1	-1	-1	0	5
8	-1	-1	-1	-1	-1	-1	-1	-1	0

NextNode

2	(σε ποιόν κόμβο της επόμενης βαθμίδας πρεπει να μεταβούμε για να κάνουμε τη βέλτιστη διαδρομή/ποιός κόμβος έδωσε το ελάχιστο κόστος)
3	
4	
6	
6	
7	
8	
8	
8	

(βέλτιστο κόστος διαδρομής από τρέχοντα κόμβο μέχρι τον τελικό κόμβο)

(λύση συνολικού προβλήματος/ βέλτιστο κόστος διαδρομής από κόμβο 0 -> τελικό κόμβο 8)

Προδιαγραφές προβλήματος:

- Οι μεταβάσεις νοούνται από την βαθμίδα κ προς την βαθμίδα κ+1. Δεν επιτρέπονται μεταβάσεις από τους κόμβους μιας βαθμίδας προς τους κόμβους μικρότερης ή ίσης βαθμίδας.
- Κάθε μετάβαση από την βαθμίδα κ γίνεται μόνο στην βαθμίδα κ+1 και όχι σε μεγαλύτερη βαθμίδα

Επίλυση του γράφου στο δεύτερο μέρος της άσκησης:

Έξοδος τερματικού όπως εμφανίζεται κατά την εκτέλεση του προγράμματος που δημιουργήθηκε:

```
Number of nodes: 16
B table:
    1     2     3     4     3     2     1
Graph's nodes:
    0       1       3       6      10      13      15
            2       4       7      11      14
                    5       8      12
                            9
Cost table:
    0      11      12      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1
   -1       0      -1      13       5      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1
   -1      -1       0      -1       9      11      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1
   -1      -1      -1       0      -1      -1      10       7      -1      -1      -1      -1      -1      -1      -1      -1
   -1      -1      -1      -1       0      -1      -1       8       9      -1      -1      -1      -1      -1      -1      -1
   -1      -1      -1      -1      -1       0      -1      -1       7       6      -1      -1      -1      -1      -1      -1
   -1      -1      -1      -1      -1      -1       0      -1      -1      -1      16      -1      -1      -1      -1      -1
   -1      -1      -1      -1      -1      -1      -1       0      -1      -1      13      12      -1      -1      -1      -1
   -1      -1      -1      -1      -1      -1      -1      -1       0      -1      -1       8      13      -1      -1      -1
   -1      -1      -1      -1      -1      -1      -1      -1      -1       0      -1      -1      10      -1      -1      -1
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0      -1      -1       5      -1      -1
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0      -1      15       7      -1
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0      -1       6      -1
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0      -1      11
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0      14
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0
Please give starting node:
0
The optimal cost from node 0 to the end: 53
The best path from node 0: 1 4 7 10 13 15
OPT table:
    0      53      58      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      53
   -1       0      -1      49      42      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      42
   -1      -1       0      -1      46      47      -1      -1      -1      -1      -1      -1      -1      -1      -1      46
   -1      -1      -1       0      -1      -1      42      36      -1      -1      -1      -1      -1      -1      -1      36
   -1      -1      -1      -1       0      -1      -1      37      38      -1      -1      -1      -1      -1      -1      37
   -1      -1      -1      -1      -1       0      -1      -1      36      36      -1      -1      -1      -1      -1      36
   -1      -1      -1      -1      -1      -1       0      -1      -1      -1      32      -1      -1      -1      -1      32
   -1      -1      -1      -1      -1      -1      -1       0      -1      -1      29      33      -1      -1      -1      29
   -1      -1      -1      -1      -1      -1      -1      -1       0      -1      -1      29      33      -1      -1      29
   -1      -1      -1      -1      -1      -1      -1      -1      -1       0      -1      -1      30      -1      -1      30
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0      -1      -1      16      -1      16
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0      -1      26      21      21
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0      -1      20      20
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0      -1      11
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0      14
   -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1      -1       0
NextNode table:
1       4       4       7       7       9       10      10      11      12      13      14      14      15      15      15
```

Όπως φαίνεται και στην εικόνα, η βραχύτερη διαδρομή από τον κόμβο 0 στον κόμβο 15, (δηλαδή από το σημείο Α στο σημείο Β) κοστίζει 53 μονάδες χρόνου. Η βραχύτερη διαδρομή αποτελείται από τους κόμβους $0 \rightarrow 1 \rightarrow 4 \rightarrow 7 \rightarrow 10 \rightarrow 13 \rightarrow 15$.

Παράρτημα - παρατηρήσεις πάνω στον κώδικα:

- Το πρόγραμμα είναι υλοποιημένο σε γλώσσα C.
- Κατά την δημιουργία του πίνακα OPT χρειάστηκε η σύγκριση με έναν αυθαίρετα μεγάλο αριθμό. Έτσι η αρχικοποίηση της μεταβλητής min έγινε στο 999999999.
- Η δέσμευση των πινάκων γίνεται με δυναμικό τρόπο.
- Μετά το τέλος της χρήσης των πινάκων γίνεται αποδέσμευση της μνήμης που δεσμεύτηκε.
- Τα αρχεία της εργασίας και περισσότερες πληροφορίες: github.com/nkyparissas/Dynamic_Programming_Exercise_1.git