



## Θ.Ε. ΠΛΣ50 (2007-2008) – ΓΡΑΠΤΗ ΕΡΓΑΣΙΑ Ε4

### Στόχος

Στην εργασία αυτή θα ασχοληθούμε με **δενδρικές δομές δεδομένων** σε συνδυασμό με αναδρομικές συναρτήσεις. Συγκεκριμένα καλείστε να υλοποιήσετε τον αλγόριθμο **Huffman** για να συμπίεσετε και να αποσυμπίεσετε το περιεχόμενο αρχείων κειμένου. Θα ξεκινήσουμε με την περιγραφή του αλγόριθμου Huffman και στη συνέχεια θα δώσουμε τα ζητούμενα.

### Περιγραφή Κωδικών Huffman

#### A. Εισαγωγή

Είναι γνωστό ότι για την αναπαράσταση κάποιου χαρακτήρα σε ένα υπολογιστικό σύστημα χρησιμοποιούνται πίνακες οι οποίοι ορίζουν τη δυαδική αναπαράσταση κάθε χαρακτήρα. Οι πιο γνωστοί τέτοιοι πίνακες είναι ο ASCII (7 bit), το Extended-ASCII (8 bit – οι πρώτοι 128 χαρακτήρες του δίδονται στο τέλος της εκφώνησης) και το UNICODE (16-bit). Για παράδειγμα, εάν έχουμε ένα αρχείο το οποίο περιέχει 1000 χαρακτήρες Extended-ASCII τότε το αρχείο θα έχει μέγεθος 8000 bits (δηλ., περίπου 1KByte).

Η χρήση σταθερού μεγέθους κωδικού αναπαράστασης χαρακτήρων (**constant-length codes**) κάνει πολύ εύκολη την κωδικοποίηση και αποκωδικοποίηση κάποιου αρχείου αφού κάθε όγδοο δυαδικό ψηφίο είναι η αρχή κάποιου νέου χαρακτήρα. Έτσι εάν ένα αρχείο περιέχει την ακόλουθη δυαδική ακολουθία **011000110110000101110010** τότε ο υπολογιστής μπορεί εύκολα να βρει ότι αυτό αναπαριστά την λέξη "car" (δες τον πίνακα ASCII στο τέλος της εκφώνησης). Η μέθοδος Huffman από την άλλη, η οποία επινοήθηκε το 1952 από τον τότε φοιτητή του MIT David A. Huffman, δημιουργεί μεταβλητού μεγέθους κωδικούς (**variable-length codes**) βάση του πιο κάτω συλλογισμού:

*Ένας χαρακτήρας που έχει ψηλή πιθανότητα εμφάνισης σε ένα κείμενο (π.χ. 'α', 'ε', ' ', ...), πρέπει να χρησιμοποιεί όσο το δυνατό λιγότερα bits, ενώ χαρακτήρες με χαμηλότερη πιθανότητα εμφάνισης (π.χ. '@') μπορούν να χρησιμοποιούν περισσότερα bits.*

Αυτή η μέθοδος στοχεύει στην κατά μέσο όρο μείωση του αριθμού των δυαδικών ψηφίων που χρειάζονται για την κωδικοποίηση κάποιου αρχείου. Οι κωδικοί Huffman βρίσκουν πάρα πολλές πρακτικές εφαρμογές σε προγράμματα συμπίεσης (π.χ., pack), κωδικοποίηση αρχείων (π.χ., mp3, jpg κτλ.) και στα δίκτυα.

Σημειώστε ότι οι κωδικοί Huffman μας επιτρέπουν να συμπίεσουμε το περιεχόμενο των αρχείων χωρίς να χάσουμε πληροφορία (**lossless compression**), άρα ένα αποκωδικοποιημένο κείμενο είναι το ίδιο με το αρχικό κείμενο σε αντίθεση με άλλες μεθόδους που χρησιμοποιούνται για την συμπίεση πολυμέσων και οι οποίες δεν μας επιτρέπουν να πάρουμε πίσω την αρχική πληροφορία (**lossy compression**).

#### B. Περιγραφή Αλγόριθμου Huffman μέσω Παραδείγματος

Για να κατανοήσουμε τη λειτουργία των κωδικών Huffman θα περιγράψουμε τώρα ένα αναλυτικό παράδειγμα. Υποθέστε ότι έχουμε ένα αρχείο **FILE** το οποίο αποτελείται από 100 χαρακτήρες που προέρχονται μόνο από το αλφάβητο {a,b,c,d,e,f,g}.

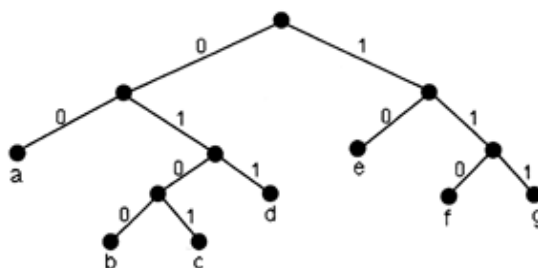
Επίσης υποθέστε ότι έχουμε αναλύσει ένα-ένα τους χαρακτήρες του FILE και βρίσκουμε ότι η πιθανότητα εμφάνισης κάθε χαρακτήρα του αλφάβητου (δηλ., πλήθος\_εμφανίσεων\_χαρακτήρα / 100) είναι αυτή που δίνεται στον Πίνακα 1 (τον οποίο ονομάζουμε **Πίνακα Πιθανοτήτων**).



Χαρακτήρας	Συχνότητα	Πιθανότητα Εμφάνισης
a	20	0.20
b	5	0.05
c	10	0.10
d	10	0.10
e	25	0.25
f	15	0.15
g	15	0.15

Πίνακας 1: Ο Πίνακας Πιθανοτήτων (πρώτη και τρίτη στήλη)

Ο πίνακας πιθανοτήτων θα αξιοποιηθεί από τον **Αλγόριθμο του Huffman** για να δημιουργηθούν κωδικοί μεταβλητού μεγέθους ανάλογα με την πιθανότητα εμφάνισης των χαρακτήρων σε ένα κείμενο. Συγκεκριμένα, ο αλγόριθμος του Huffman κατασκευάζει από τον Πίνακα Πιθανοτήτων ένα **Δυαδικό Δένδρο Huffman** όπως αυτό που παρουσιάζεται στο Σχήμα 1 (η περιγραφή του αλγόριθμου ακολουθεί στο Μέρος II.Γ για αυτό δεν χρειάζεται ακόμη να καταλάβετε πώς δημιουργήθηκε το δένδρο).



Σχήμα 1: Το Δυαδικό Δένδρο Huffman που αντιστοιχεί στις πιθανότητες του Πίνακα 1

Υποθέστε τώρα ότι λαμβάνετε τα 7 μονοπάτια από την ρίζα του δένδρου προς τους τερματικούς κόμβους του σχήματος 1, καταγράφοντας σε κάθε κίνηση την κατεύθυνση προς την οποία πηγαίνετε (αριστερά καταγράφουμε το 0 και δεξιά καταγράφουμε το 1). Αυτή η διαδικασία δημιουργεί ένα κωδικό ανά γράμμα όπως φαίνεται στον Πίνακα 2 (τον οποίο θα ονομάσουμε **Πίνακα Huffman**)

Χαρακτήρας	a	b	c	d	e	f	g
Κωδικός Huffman	00	0100	0101	011	10	110	111

Πίνακας 2: Ο Πίνακας Huffman

Όπως βλέπουμε οι χαρακτήρες 'a' και 'e' οι οποίοι είχαν την ψηλότερη πιθανότητα εμφάνισης (δηλ., 20% και 25% αντίστοιχα) έχουν και το μικρότερο μήκος κωδικού (2 bits) ενώ ο χαρακτήρας 'e' που έχει την χαμηλότερη πιθανότητα εμφάνισης (δηλ., 5%) έχει το μεγαλύτερο μήκος κωδικού (4 bits).

Για να κατανοήσουμε το πλεονέκτημα της κωδικοποίησης Huffman (και χωρίς αυτό να έχει άμεση σχέση με τα ζητούμενα της άσκησης) αναλύουμε τώρα το παράδειγμα της εκφώνησης για να βρούμε πόσο αποδοτική μπορεί να είναι η συμπίεση. Καταρχήν θυμηθείτε ότι για να κωδικοποιήσουμε ένα οποιοδήποτε κείμενο με σταθερού μεγέθους κωδικούς (constant-length codes) για το αλφάβητο μας {a,b,c,d,e,f,g} θέλουμε **3-bits ανά χαρακτήρα** (το οποίο μας δίνει  $2^3=8$  κωδικούς που είναι αρκετοί για το αλφάβητο 7 χαρακτήρων).



Εάν, από την άλλη, κάνουμε χρήση των κωδικών Huffman (όπως αυτοί δίνονται στον Πινάκα 2), τότε θέλουμε κατά μέσο όρο μόνο:

$$\sum_{i \in \{a,b,c,d,e,f,g\}} \text{πιθανότητα\_εμφάνισης}_i * \text{μήκος\_κωδικού}_i = (0.2*2) + (0.05*4) + (0.1*4) + (0.1*3) + (0.25*2) + (0.15*3) + (0.15*3) = \mathbf{2.7 \text{ bits}} / \text{χαρακτήρα (αντί 3 bits ανά χαρακτήρα)}.$$

Σε πραγματικές εφαρμογές η εξοικονόμηση είναι πολύ μεγαλύτερη!

## Γ. Δημιουργία Δένδρου Huffman

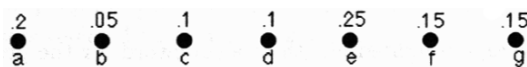
Θα περιγράψουμε τώρα σε υψηλό επίπεδο αφαιρετικότητας τον αλγόριθμο του Huffman (ο οποίος μετατρέπει τον πίνακα πιθανοτήτων του πίνακα 1 στο δυαδικό δέντρο του σχήματος 1). Στην συνέχεια θα δώσουμε ένα αναλυτικό παράδειγμα για να εμπεδώσετε την λειτουργία του αλγορίθμου:

- Αρχικά για κάθε γράμμα δημιουργήσε ένα δένδρο αποτελούμενο μόνο από τη ρίζα σημειωμένη με την αντίστοιχη πιθανότητα εμφάνισης του. Στο σχήμα 2 (επανάληψη 1) δείχνουμε 7 χαρακτήρες μαζί με την αντίστοιχη πιθανότητα εμφάνισης.
- Επανάλαβε τα ακόλουθα βήματα μέχρι να μείνει μόνο ένα δένδρο:
  1. Επέλεξε τα δύο δένδρα S1 και S2 τα οποία έχουν τη χαμηλότερη πιθανότητα εμφάνισης  $p_1$  και  $p_2$  αντίστοιχα ( $p_1 \leq p_2$ ). Στην περίπτωση ισοβαθμίας πιθανοτήτων τριών ή περισσότερων δένδρων επέλεξε όποια δένδρα επιθυμείς.
  2. Συγχώνευσε τα δένδρα S1 και S2 σε ένα δυαδικό δένδρο με ρίζα το S ως ακολούθως: τοποθέτησε το S1 στα αριστερά του S και το S2 στα δεξιά του S. Επίσης σημείωσε ως πιθανότητα εμφάνισης του S το  $(p_1 + p_2)$ .

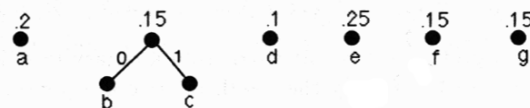


## ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

Επανάληψη 1:



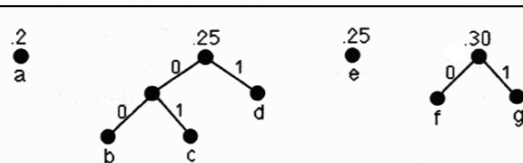
Επανάληψη 2:



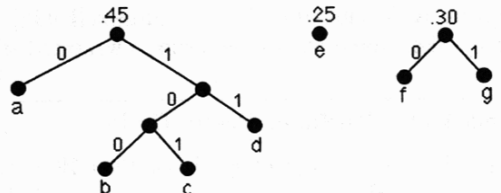
Επανάληψη 3:



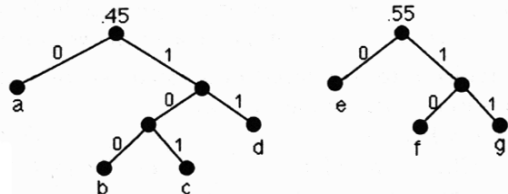
Επανάληψη 4:



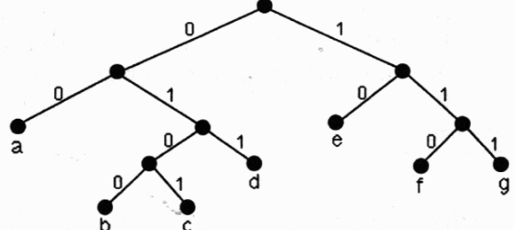
Επανάληψη 5:



Επανάληψη 6:



Επανάληψη 7:

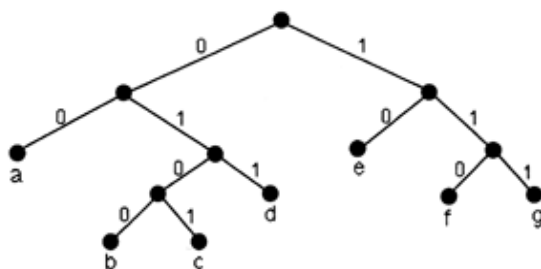


Σχήμα 2: Αναλυτικό παράδειγμα δημιουργίας Δυαδικού Δένδρου Huffman



## Δ. Μετατροπή Δένδρου Huffman σε Πίνακα Huffman

Έχοντας δημιουργήσει το δένδρο huffman τώρα περιγράφουμε πώς το επεξεργαζόμαστε αποδοτικά. Σημειώστε καταρχάς ότι ο κωδικός huffman κάποιου χαρακτήρα είναι το μονοπάτι το οποίο διανύουμε από τη ρίζα του δένδρου μέχρι να φτάσουμε στον χαρακτήρα. Ωστόσο για να βρούμε το σωστό μονοπάτι θα χρειαζόταν στη χειρότερη περίπτωση να ψάξουμε όλα τα μονοπάτια του δένδρου ενώ εάν αποθηκεύαμε το δένδρο σαν ένα πίνακα (όπως αυτός φαίνεται πιο κάτω στα δεξιά) τότε θα μπορούσαμε να γλιτώσουμε αυτή την ακριβή διαδικασία. Για να φτιάξουμε τον πίνακα Huffman μπορούμε να χρησιμοποιήσουμε μεθόδους διάσχισης του δένδρου που είδαμε στα πλαίσια της Θεματικής Ενότητας.



Το Δυαδικό Δένδρο Huffman

Χαρακτήρας	Κωδικός Huffman
a	00
b	0100
c	0101
d	011
e	10
f	110
g	111

Ο Πίνακας Huffman

## Ε. Κωδικοποίηση Αρχείου

Χρησιμοποιώντας τον Πίνακα Huffman μπορούμε τώρα να κωδικοποιήσουμε ένα αρχείο χαρακτήρων με την ακόλουθη λογική: διαβάζουμε ένα χαρακτήρα **c** από το αρχείο και στη συνέχεια βρίσκουμε από τον Πίνακα Huffman τον κωδικό που αντιστοιχεί στον **c** (έστω ότι τον ονομάζομαι **h(c)**). Τέλος γράφουμε τον **h(c)** σε ένα νέο αρχείο **outfile**.

Π.χ. για την λέξη **bag** θα δημιουργήσουμε με τον πιο πάνω πίνακα τον κωδικό **010000111**.

**Σημείωση:** Στην υλοποίησή σας ο παραγόμενος κωδικός είναι δυνατό να διαφέρει. Αυτό μπορεί να οφείλεται σε δυο παράγοντες: α) στο γεγονός ότι έχετε χρησιμοποιήσει μεγαλύτερο αλφάβητο και β) στο γεγονός ότι έχετε επιλέξει διαφορετικούς κόμβους προς συγχώνευση σε περίπτωση ισοβαθμίας πιθανοτήτων τριών ή περισσότερων κόμβων.

## Ζ. Αποκωδικοποίηση Αρχείου

Για να αποκωδικοποιήσουμε μια ακολουθία από δυαδικά ψηφία απροσδιορίστου μεγέθους αξιοποιούμε το δένδρο Huffman με την έξης διαδικασία:

- Διαβάζουμε σειριακά την ακολουθία από δυαδικούς χαρακτήρες από αριστερά προς τα δεξιά.
- Εάν βρούμε 0 προχωρούμε αριστερά στο δένδρο, ενώ στην αντίθετη περίπτωση προχωρούμε δεξιά μέχρι να βρούμε κάποιο τερματικό κόμβο οπότε τυπώνουμε και τον χαρακτήρα που είναι αποθηκευμένος σε αυτό τον κόμβο.

Για παράδειγμα για να αποκωδικοποιήσουμε την ακολουθία **0110111100** ακολουθούμε την έξης σειρά:

0	1	1	0	1	1	1	1	0	0
Left	right	right	left	right	right	right	right	left	left
				<b>B</b>			<b>A</b>		<b>G</b>

Για να είναι ο πιο πάνω τρόπος αποκωδικοποίησης λειτουργήσιμος δεν πρέπει κανένας κωδικός να είναι πρόθεμα (δηλαδή να περιέχεται σαν το αρχικό τμήμα) οποιουδήποτε άλλου κωδικού. Ευτυχώς,



αυτή η απαίτηση διασφαλίζεται πάντοτε από το γεγονός ότι οι χαρακτήρες βρίσκονται αποθηκευμένοι σαν τερματικοί κόμβοι επομένως μεταξύ οποιωνδήποτε δυο χαρακτήρων ο κωδικός Huffman θα διαφέρει πάντοτε κατά τουλάχιστο 1 bit!

## Ζητούμενα της εργασίας

Κατασκευάστε ένα πρόγραμμα C, το οποίο υλοποιεί τις ακόλουθες λειτουργίες:

- **Θέμα 1:** Υπολογισμός Πιθανοτήτων
- **Θέμα 2:** Δημιουργία Δένδρου Huffman
- **Θέμα 3:** Κωδικοποίηση Αρχείου
- **Θέμα 4:** Αποκωδικοποίηση Αρχείου

### Θέμα 1: Υπολογισμός Πιθανοτήτων

Το πρώτο ζητούμενο είναι η κατασκευή του πίνακα πιθανοτήτων των χαρακτήρων. Σας δίνεται ένα αρκετά μεγάλο αρχείο κειμένου (`sample.txt`) το οποίο πρέπει να χρησιμοποιηθεί για να υπολογιστούν οι πιθανότητες των πρώτων 128 χαρακτήρων του πίνακα ASCII.

Συγκεκριμένα, έστω Huffman η εντολή που έχει δημιουργηθεί από την μεταγλώττιση του προγράμματος σας. Το πρόγραμμα σας πρέπει να υποστηρίζει την ακόλουθη εντολή:

```
$huffman -p sample.txt probfile.txt
```

Το όρισμα `-p` δηλώνει ότι θέλετε να υπολογίσετε την πιθανότητα (probability) των χαρακτήρων. Το αρχείο `sample.txt` είναι ένα μεγάλο αρχείο κειμένου το οποίο πρέπει να χρησιμοποιηθεί για την κατασκευή του αρχείου `probfile.txt` (το αρχείο `sample.txt` μπορείτε να το προμηθευτείτε από το αρχείο `ex4-supplementary.zip` το οποίο βρίσκεται αναρτημένο στην ηλεκτρονική μας πύλη). Το αρχείο `probfile.txt` είναι το αποτέλεσμα του προγράμματος σας και περιέχει 128 πραγματικούς αριθμούς οι οποίοι αναπαριστούν την πιθανότητα εμφάνισης κάθε χαρακτήρα στην σειρά των χαρακτήρων ASCII.

#### Παράδειγμα `sample.txt`:

```
International Coffee Organization, ICO, producing countries will present a  
proposal for reintroducing export quotas for 12 months from April 1 with a  
...
```

#### Παράδειγμα `probfile.txt`:

```
0.03333333  
0.005  
...
```

### Θέμα 2: Δημιουργία Δένδρου Huffman

Το δεύτερο ζητούμενο είναι η δημιουργία του δένδρου Huffman και η εκτύπωση των παραγόμενων κωδικών στην οθόνη και σε αρχείο. Συγκεκριμένα, το πρόγραμμα σας πρέπει να υποστηρίζει την ακόλουθη εντολή:

```
$huffman -s probfile.txt
```

Σε αυτή την περίπτωση το πρόγραμμα πρέπει να εκτυπώνει στην οθόνη τους κωδικούς huffman όλων των χαρακτήρων του πίνακα ASCII στο διάστημα [32 έως 126] (αυτό επειδή οι υπόλοιποι δεν θα παρουσιαστούν ορθά στην οθόνη). Επίσης πρέπει να τυπώνει σε ένα αρχείο `"codes.txt"` όλους τους κωδικούς του πίνακα ASCII στο διάστημα [0 έως 127]. Το αρχείο `probfile.txt` περιέχει και



πάλι τις πιθανότητες για κάθε επί μέρους χαρακτήρα του πίνακα ASCII (υποθέτουμε ότι έχει δημιουργηθεί από το θέμα 1)

Παράδειγμα `codes.txt`:

```
111
01101
0111
...
```

### Θέμα 3: Κωδικοποίηση Αρχείου

Το τρίτο ζητούμενο είναι η κωδικοποίηση ενός αρχείου κειμένου `data.txt` χρησιμοποιώντας το δένδρο Huffman το οποίο δημιουργεί το πρόγραμμα σας (με τις συναρτήσεις που υλοποιήθηκαν στο θέμα 2). Συγκεκριμένα, το πρόγραμμα σας πρέπει να υποστηρίζει την ακόλουθη εντολή:

```
$huffman -e probfile.txt data.txt data.txt.enc
```

Το αρχείο `probfile.txt` περιέχει τις πιθανότητες για κάθε επί μέρους χαρακτήρα του πίνακα ASCII. Το αρχείο κειμένου `data.txt` περιέχει μια απροσδιορίστου μεγέθους ακολουθία χαρακτήρων ASCII. Τέλος το αρχείο κειμένου `data.txt.enc` περιέχει την κωδικοποιημένη ακολουθία ως μια σειρά από "1" και "0".

Παράδειγμα `data.txt.enc` :

```
0101111111001001011100010101110111111110101001001010001001....
```

### Θέμα 4: Αποκωδικοποίηση Αρχείου

Το τέταρτο ζητούμενο είναι η αποκωδικοποίηση ενός αρχείου κειμένου `data.txt.enc` χρησιμοποιώντας το δένδρο Huffman το οποίο δημιουργεί το πρόγραμμα σας βάση του `probfile.txt` (με τις συναρτήσεις που υλοποιήθηκαν στο θέμα 2). Συγκεκριμένα, το πρόγραμμα σας πρέπει να υποστηρίζει την ακόλουθη εντολή:

```
$huffman -d probfile.txt data.txt.enc data.txt.new
```

Το κωδικοποιημένο αρχείο `data.txt.enc` θα περιέχει το συμπιεσμένο αρχείο (από το θέμα 3) σαν μια ακολουθία ψηφίων "1" και "0" ενώ το αρχείο `data.txt.new` θα πρέπει να περιέχει το αποσυμπιεσμένο αρχείο (αν η υλοποίηση σας είναι σωστή τότε το `data.txt.new` περιέχει τις ίδιες πληροφορίες με το `data.txt`)

Η εντολή `diff`, ελέγχει στο UNIX εάν δυο αρχεία περιέχουν τα ίδια στοιχεία

```
$diff data.txt.new data.txt => δεν πρέπει να επιστρέφει τίποτα.
```



## Κριτήρια αξιολόγησης

Θέμα 1 (Υπολογισμός Πιθανοτήτων)	10
Θέμα 2 (Δημιουργία Δένδρου Huffman)	50
Θέμα 3 (Κωδικοποίηση Αρχείου)	15
Θέμα 4 (Αποκωδικοποίηση Αρχείου)	15
Γενική εικόνα (ευανάγνωστος κώδικας, σχόλια, δομή, αποδοτικότητα κλπ.)	10
<b>ΣΥΝΟΛΟ</b>	<b>100</b>

## Τρόπος – Ημερομηνία Παράδοσης

Η εργασία σας θα πρέπει να έχει φτάσει στον Καθηγητή-Σύμβουλό σας μέχρι την Κυριακή 6/1/2008.

Περιμένουμε όλες οι εργασίες να αποσταλούν μέσω Email και να είναι γραμμένες σε επεξεργαστή κειμένου MSWord (διευκρινήσεις, σχόλια και παραδοχές που θεωρείτε απαραίτητα να συμπεριλάβετε στην εργασία σας). Τα τμήματα κώδικα και τα τμήματα δεδομένων θα βρίσκονται σε ξεχωριστά αρχεία και θα υπάρχουν αναφορές σ' αυτά από το κείμενο της εργασίας. Στον Καθηγητή-Σύμβουλό σας, σε κάθε περίπτωση, στέλνετε ΕΝΑ μόνο αρχείο (συμπιεσμένο).

Δεν θα δοθεί παράταση στην παράδοση της εργασίας πέραν της ως άνω αναφερόμενης ημέρας και ώρας, για κανένα λόγο. Την Τρίτη 8/1/2008 ώρα 13:00, θα δημοσιευθεί ενδεικτική επίλυση της εργασίας στο διαδίκτυο (portal).

**Καλή Επιτυχία!!!**





## ΠΑΡΑΡΤΗΜΑ: Ο Πίνακας Extended ASCII (πρώτοι 128 χαρακτήρες)

000 00000000	NUL (Null char.)	064 01000000	@ (AT symbol)
001 00000001	SOH (Start of Header)	065 01000001	A
002 00000010	STX (Start of Text)	066 01000010	B
003 00000011	ETX (End of Text)	067 01000011	C
004 00000100	EOT (End of Transmission)	068 01000100	D
005 00000101	ENQ (Enquiry)	069 01000101	E
006 00000110	ACK (Acknowledgment)	070 01000110	F
007 00000111	BEL (Bell)	071 01000111	G
008 00001000	BS (Backspace)	072 01001000	H
009 00001001	HT (Horizontal Tab)	073 01001001	I
010 00001010	LF (Line Feed)	074 01001010	J
011 00001011	VT (Vertical Tab)	075 01001011	K
012 00001100	FF (Form Feed)	076 01001100	L
013 00001101	CR (Carriage Return)	077 01001101	M
014 00001110	SO (Shift Out)	078 01001110	N
015 00001111	SI (Shift In)	079 01001111	O
016 00010000	DLE (Data Link Escape)	080 01010000	P
017 00010001	DC1 (XON) (Device Control 1)	081 01010001	Q
018 00010010	DC2 (Device Control 2)	082 01010010	R
019 00010011	DC3 (XOFF)(Device Control 3)	083 01010011	S
020 00010100	DC4 (Device Control 4)	084 01010100	T
021 00010101	NAK (Negative Acknowledgement)	085 01010101	U
022 00010110	SYN (Synchronous Idle)	086 01010110	V
023 00010111	ETB (End of Trans. Block)	087 01010111	W
024 00011000	CAN (Cancel)	088 01011000	X
025 00011001	EM (End of Medium)	089 01011001	Y
026 00011010	SUB (Substitute)	090 01011010	Z
027 00011011	ESC (Escape)	091 01011011	[ (left/opening bracket)
028 00011100	FS (File Separator)	092 01011100	\ (back slash)
029 00011101	GS (Group Separator)	093 01011101	] (right/closing bracket)
030 00011110	RS (Request to Send)(Rec Separator)	094 01011110	^ (caret/cirumflex)
031 00011111	US (Unit Separator)	095 01011111	_ (underscore)
032 00100000	SP (Space)	096 01100000	`
033 00100001	! (exclamation mark)	097 01100001	a
034 00100010	" (double quote)	098 01100010	b
035 00100011	# (number sign)	099 01100011	c
036 00100100	\$ (dollar sign)	100 01100100	d
037 00100101	% (percent)	101 01100101	e
038 00100110	& (ampersand)	102 01100110	f
039 00100111	' (single quote)	103 01100111	g



# ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

040 00101000	( (left/opening parenthesis)	104 01101000	h
041 00101001	) (right/closing parenthesis)	105 01101001	i
042 00101010	* (asterisk)	106 01101010	j
043 00101011	+ (plus)	107 01101011	k
044 00101100	, (comma)	108 01101100	l
045 00101101	- (minus or dash)	109 01101101	m
046 00101110	. (dot)	110 01101110	n
047 00101111	/ (forward slash)	111 01101111	o
048 00110000	0	112 01110000	p
049 00110001	1	113 01110001	q
050 00110010	2	114 01110010	r
051 00110011	3	115 01110011	s
052 00110100	4	116 01110100	t
053 00110101	5	117 01110101	u
054 00110110	6	118 01110110	v
055 00110111	7	119 01110111	w
056 00111000	8	120 01111000	x
057 00111001	9	121 01111001	y
058 00111010	: (colon)	122 01111010	z
059 00111011	; (semi-colon)	123 01111011	{ (left/opening brace)
060 00111100	< (less than)	124 01111100	(vertical bar)
061 00111101	= (equal sign)	125 01111101	} (right/closing brace)
062 00111110	> (greater than)	126 01111110	~ (tilde)
063 00111111	? (question mark)	127 01111111	DEL (delete)