1η Σειρά Ασκήσεων

Φίλιππος Δουραχαλής, 3170045

Άσκηση 1)

- Μέγεθος τομέα = 4096 Bytes
- 65536 ίχνη/επιφάνεια
- 8 πλακέτες διπλής όψης = 16 επιφάνειες
- 256 τομείς/ίχνος
- SeekTime_{avg}= 10ms
- Ταχύτητα περιστροφής 7200rpm

a) Η χωρητικότητα κάθε ίχνους θα είναι:
$1\ track\ =\ 256 * 4096\ =\ 104856\ Bytes\ =\ 1MB$

Η χωρητικότητα μιας επιφάνειας του δίσκου θα είναι:
$1\ surface\ =\ 65536 * 1048576\ =\ 68719476736\ Bytes\ =\ 64GB$

Η χωρητικότητα ολόκληρου του δίσκου θα είναι
$Total\_Size\ =\ 16 * 64\ =\ 1024GB\ =\ 1TB$

b) Ο αριθμός κυλίνδρων ισούται με τον αριθμό των ιχνών κάθε επιφάνειας του δίσκου Άρα ο δίσκος έχει συνολικά 65536 κυλίνδρους.

c) Η μέγιστη καθυστέρηση περιστροφής προκύπτει όταν χρειαζόμαστε μια πλήρη περιστροφή για να βρεθεί ο σωστός τομέας κάτω από την κεφαλή. Άρα ισούται με:

$$\text{Rotational\_Delay}_{max} = {}^{60}/_{7200} = 0{,}0083\ \text{sec} = 8{,}3\ msec$$

Η μέση καθυστέρηση περιστροφής ισούται με το προηγούμενο αποτέλεσμα δια 2

$$\text{Rotational\_Delay}_{avg} = {}^{60}/_{7200} * {}^{1}/_{2} = 0{,}00415\ \text{sec} = 4{,}15\ msec$$

d) Κάθε κεφαλή διαβάζει σε μια περιστροφή 1 ίχνος, άρα όλες οι κεφαλές διαβάζουν ταυτόχρονα 16 ίχνη = 16MB
Ο δίσκος περιστρέφεται 7200 φορές το λεπτό, δηλαδή $60/7200 = 120$ φορές το δευτερόλεπτο
Άρα σε 1 sec ο ρυθμός μεταφοράς θα είναι:

$$\text{Transfe}_{\text{Rate}} = 120 * 16 * 1048576 = 2013265920\ \text{Bytes/sec} = 1{,}875\ \text{GB/sec}$$

Για την ανάγνωση του ίχνους απαιτούνται $\frac{256}{256} * 4{,}15 + 10 = 14{,}5\ msec$

Άσκηση 2)

- Μέγεθος γνωρίσματος a = 10 Bytes
- Μέγεθος εγγραφής σχέσης = 10 + 50 + 30 + 18 + 20 = 128 Bytes
- Μέγεθος σχέσης = N εγγραφές
- Μέγεθος δείκτη = 6 Bytes
- Μέγεθος σελίδας = 1024 Bytes
- Κάθε σελίδα θα περιέχει $1024/128 = 8$ $^{εγγραφές}/_{σελίδα}$

Άρα χρειαζόμαστε συνολικά $\frac{N}{8}$ σελίδες για την αποθήκευση της σχέσης.

Τέλος κάθε εγγραφή του ευρετηρίου θα έχει μέγεθος 10 + 6 = 16 Bytes.

a) Ένα πυκνό ευρετήριο θα περιέχει όλες τις τιμές του γνωρίσματος πάνω στο οποίο ο ρίζεται.
Εφόσον το γνώρισμα a είναι πρωτεύον κλειδί, περιέχει μοναδικές τιμές,
άρα το πυκνό ευρετήριο Θα περιέχει N εγγραφές.

Κάθε σελίδα μπορεί να περιέχει $^{1024}/_{16} = 64$ εγγραφές του ευρετηρίου.

Άρα τελικά για την αποθήκευσή του απαιτούνται: $\frac{\mathrm{N}}{64}$ σελίδες.

Για την αποθήκευση του ευρετηρίου και της σχέσης απαιτούνται συνολικά $\frac{\mathrm{N}}{8} + \frac{\mathrm{N}}{64} = \frac{9\mathrm{N}}{64}$ σελίδες.

b) Ένα αραιό ευρετήριο θα περιέχει μόνο την πρώτη τιμή του γνωρίσματος από κάθε σελίδα στην οποία αποθηκεύονται οι εγγραφές της σχέσης. Άρα θα περιέχει συνολικά $\mathrm{N}/8$ εγγραφές εφόσον κάθε σελίδα της σχέσης περιέχει 8 τιμές του γνωρίσματος a, όπως εξηγήθηκε ανωτέρω.
Άρα για την αποθήκευση του χρειαζόμαστε $\frac{\mathrm{N}}{8*64} = \frac{\mathrm{N}}{512}$ σελίδες.

Τέλος για την αποθήκευση της σχέσης και του ευρετηρίου χρειαζόμαστε συνολικά $\frac{65\mathrm{N}}{512}$ σελίδες του δίσκου.

Άσκηση 3)

1. Αναζήτηση εγγραφής με κλειδί = 41
   i. Ελέγχουμε την τιμή στη ρίζα του δέντρου. Εφόσον 41 > 13, ακολουθούμε τον δεξί pointer.
   ii. Ελέγχουμε τον δεξί κόμβο του 2ου επιπέδου. Παρατηρούμε ότι 37 < 41 < 43. Άρα ακολουθούμε τον 3ο pointer από τα αριστερά.
   iii. Καταλήγουμε στο φύλλο στο οποίο δείχνει ο pointer και διατρέχουμε διαδοχικά τις τιμές του (ή εκτελούμε binary search) μέχρις ότου βρούμε την τιμή του κλειδιού.

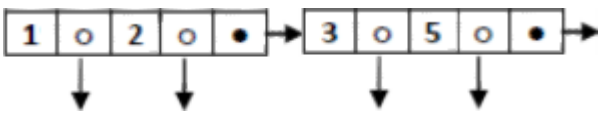
Όταν τη βρούμε, ακολουθούμε τον pointer προς την αντίστοιχη εγγραφή και την επιστρέφουμε.

2. Αναζήτηση της εγγραφής με κλειδί = 40
   i. Αντίστοιχα με προηγουμένως, ελέγχουμε την τιμή στη ρίζα. 40 > 13, επομένως ακολουθούμε και πάλι τον δεξί pointer
   ii. Μόλις βρεθούμε στον δεξί κόμβο του 2ου επιπέδου εξετάζουμε τις τιμές του. Ισχύει ότι 31 < 40 < 43, επομένως ακολουθούμε ξανά τον 3ο pointer από τα δεξιά.
   iii. Μόλις βρεθούμε στον κόμβο-φύλλο διατρέχουμε και ελέγχουμε τις τιμές του. Με γραμμική αναζήτηση θα ελέγξουμε τις τιμές 31, 37 και 41 και τότε θα σταματήσουμε, αφού το 40 αποκλείεται να βρίσκεται μετά το 41 (διότι τα φύλλα είναι διατεταγμένα) και άρα διαπιστώνουμε ότι το κλειδί όντως δεν υπάρχει, επομένως δεν επιστρέφουμε τίποτα.

3. Αναζήτηση των εγγραφών με κλειδί < 30
   i. Θεωρούμε την μικρότερη τιμή που μπορεί να πάρει το γνώρισμα και τη συγκρίνουμε με την τιμή της ρίζας. Προφανώς αυτή η τιμή θα είναι μικρότερη από 13, άρα ακολουθούμε τον αριστερό pointer προς τον αριστερό κόμβο του 2ου επιπέδου.
   ii. Επαναλαμβάνοντας την ίδια διαδικασία, θα ακολουθήσουμε πάλι τον αριστερό Pointer προς το 1ο φύλλο του δέντρου.
   iii. Τέλος διατρέχουμε όλες τις τιμές των φύλλων χρησιμοποιώντας τους ενδιάμεσους δείκτες για να περάσουμε από το ένα φύλλο στο άλλο και επιστρέφουμε τις εγγραφές για τις τιμές που ικανοποιούν το ερώτημα. Η διάσχιση τερματίζεται όταν βρούμε την τιμή 31 > 30, αφού όλες οι υπόλοιπες τιμές των φύλλων θα είναι σίγουρα μεγαλύτερες από αυτήν.

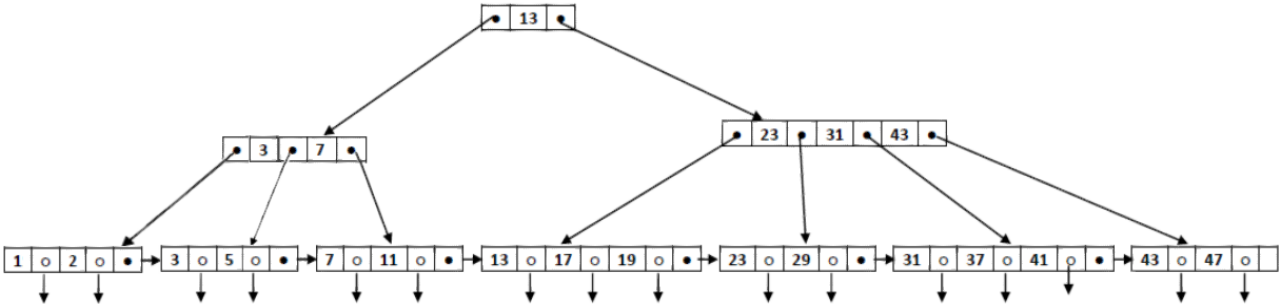
4. Αναζήτηση εγγραφών με κλειδί στο εύρος [20, 35]
   i. Θεωρούμε τη μικρότερη τιμή του εύρους, δηλαδή το 20 και τη συγκρίνουμε με την ρίζα. 20 > 13, άρα ακολουθούμε τον pointer προς το δεξί υποδέντρο.
   ii. Στον δεξί κόμβο 2ου επιπέδου, εξετάζουμε τις τιμές και παρατηρούμε ότι 20 < 23. Άρα θα ακολουθήσουμε τον πρώτο pointer προς το 3ο φύλλο.
   iii. Διατρέχουμε σειριακά τις τιμές όλων των φύλλων χρησιμοποιώντας τους pointers από το ένα στο άλλο και για κάθε τιμή που βρίσκουμε επιστρέφουμε την εγγραφή στην οποία δείχνει ο δείκτης της. Η αναζήτηση τερματίζει όταν βρούμε το πρώτο στοιχείο με τιμή μεγαλύτερη του 35, δηλαδή το 37.

5. Εισαγωγή του κλειδιού k = 1
   Εκτελώντας αναζήτηση βρίσκουμε ότι το κλειδί θα πρέπει να εισαχθεί στον πρώτο κόμβο-φύλλο, το οποίο όμως είναι γεμάτο. Επομένως για να εισάγουμε το κλειδί:
   i. Το φύλλο διασπάται στα δύο και οι τιμές μοιράζονται στις δύο νέες σελίδες που δημιουργούνται. Μετά τη διάσπαση τα φύλλα θα είναι τα εξής:

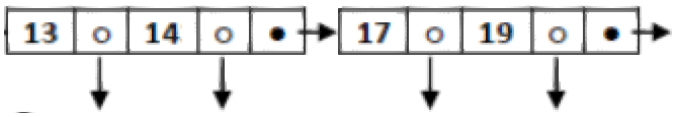


ii. Η μικρότερη τιμή του δεύτερου φύλλου προστίθεται στο γονιό, εφόσον υπάρχει διαθέσιμος χώρος σε αυτόν, ώστε να δημιουργηθεί μια “σύνδεση” με τα νέα φύλλα. Μετά την προσθήκη ο νέος κόμβος 2ου επιπέδου θα είναι ο εξής:

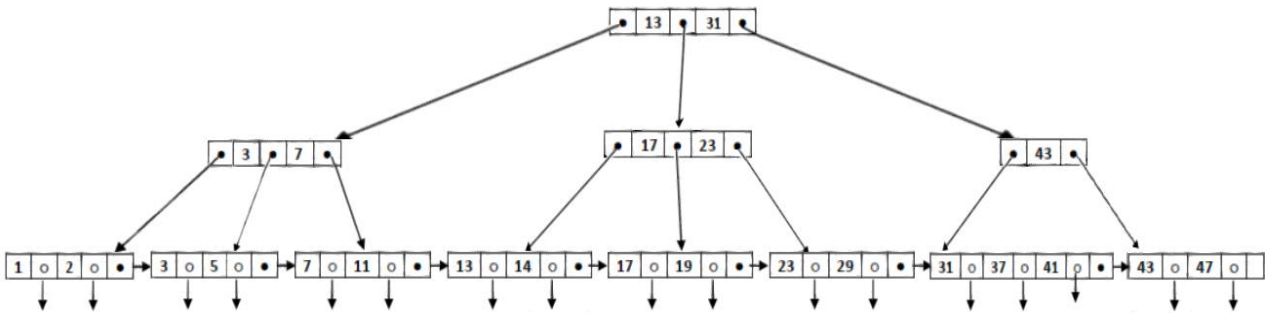
Το τελικό δέντρο που προκύπτει θα έχει την παρακάτω μορφή:



6. a) Εισαγωγή του κλειδιού k = 14
   i. Αναζητούμε το φύλλο στο οποίο πρέπει να εισαχθεί το κλειδί. Αυτό είναι το 3ο κατά σειρά φύλλο.
   ii. Το φύλλο είναι γεμάτο, επομένως διασπάται στα δύο και οι τιμές του μοιράζονται στα δύο νέα φύλλα, όπως φαίνεται παρακάτω.

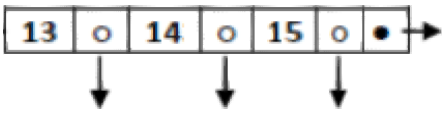


   iii. Η μικρότερη τιμή του αριστερού φύλλου που δημιουργείται (17) αντιγράφεται στον γονικό κόμβο. Ωστόσο ο κόμβος αυτός είναι ήδη γεμάτος, άρα συμβαίνει υπερχείλιση.
   iv. Ο εσωτερικός κόμβος με τη σειρά του διασπάται στα δύο με την ίδια διαδικασία και οι τιμές (δηλαδή οι pointers προς τα φύλλα) μοιράζονται στους δύο νέους κόμβους οι οποίοι θα περιέχουν τα κλειδιά {17 ,23} και {43} αντίστοιχα.
   v. Τέλος μένει να συνδεθεί ο νέος εσωτερικός κόμβος με την ρίζα. Έτσι η μικρότερη τιμή του αριστερού υποδέντρου της ρίζας, δηλαδή το 31 μεταφέρεται αυτούσιο (δεν αντιγράφεται) και προστίθεται στη ρίζα, εφόσον εκείνη δεν είναι γεμάτη. Το τελικό δέντρο θα έχει την παρακάτω μορφή:

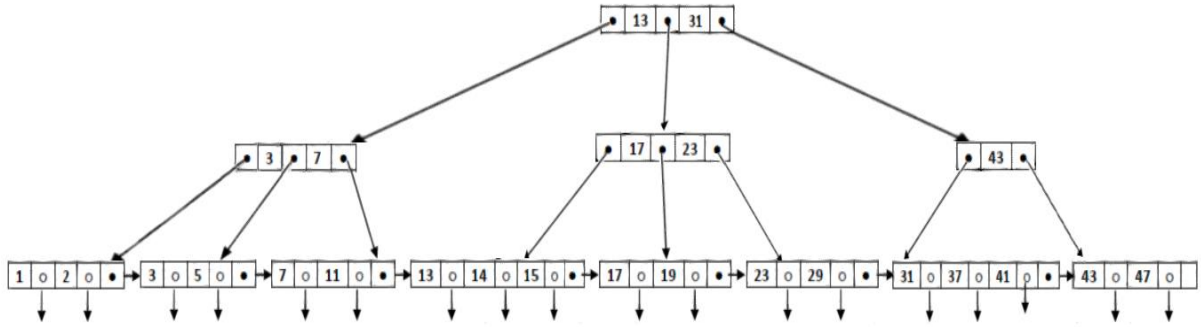


b) Εισαγωγή του κλειδιού k = 15

i. Στο προηγούμενο δέντρο αναζητούμε το φύλλο στο οποίο πρέπει να εισαχθεί το κλειδί. Αυτό είναι το 4ο φύλλο.
ii. Το φύλλο περιέχει 2 τιμές (την 13 και την 14), επομένως υπάρχει χώρος για να εισαχθεί και το κλειδί 15. Προσθέτουμε την νέα τιμή στο φύλλο χωρίς να κάνουμε κάποια περαιτέρω αλλαγή αφού δεν συνέβη υπερχείλιση. Το νέο φύλλο θα είναι:

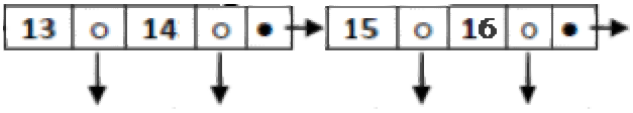


iii. Το νέο δέντρο που προκύπτει θα είναι το ακόλουθο:

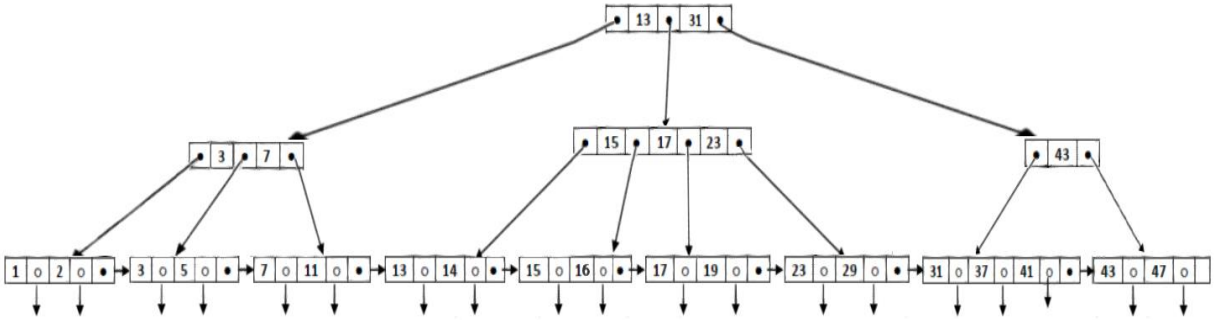


c) Εισαγωγή του κλειδιού k = 16

i. Αντίστοιχα με προηγουμένως αναζητούμε το σωστό φύλλο, το οποίο είναι ξανά το 4ο στη σειρά.
ii. Το συγκεκριμένο φύλλο παρατηρούμε ότι είναι πλήρες μετά την προηγούμενη εισαγωγή, επομένως χωρίζεται στα δύο και οι τιμές μοιράζονται όπως φαίνεται στην παρακάτω εικόνα.



iii. Η ελάχιστη τιμή του δεξιού υποδέντρου (15) αντιγράφεται στον γονικό κόμβο και εφόσον εκείνος έχει διαθέσιμο χώρο δεν προκύπτει υπερχείλιση. Το τελικό δέντρο μετά από τις τρεις εισαγωγές θα είναι το εξής:



Άσκηση 4)

- Μέγεθος σελίδας = 2048 Bytes
- Μέγεθος δείκτη = 12 Bytes
- Μέγεθος κλειδιού k = 8 Bytes

Ένας εσωτερικός κόμβος μπορεί να έχει *n* κλειδιά και *n+1* pointers προς άλλους κόμβους. Άρα το μέγιστο μέγεθος του κόμβου n θα είναι:

$$8n + 12(n + 1) = 2048$$

$$\rightarrow 20n = 2036$$

$$\rightarrow n = \frac{2036}{20}$$

$$\rightarrow n = 101$$

Δεδομένου ότι ένας δείκτης προς κάποιον κόμβο έχει το ίδιο μέγεθος με έναν δείκτη προς μια εγγραφή, κάθε κόμβος-φύλλο θα μπορεί επίσης να φιλοξενήσει 101 εγγραφές καθώς χρειάζεται να δεσμεύσουμε 12 byte για τον pointer που δείχνει στο επόμενο φύλλο και 12 byte για κάθε pointer που δείχνει στην εγγραφή του αντίστοιχου κλειδιού, δηλαδή

$$\frac{2048-12}{12+8} = 101 \text{ εγγραφές}$$

Άρα τελικά το σύνολο όλων των εγγραφών που μπορεί να ευρετηριαστούν είναι

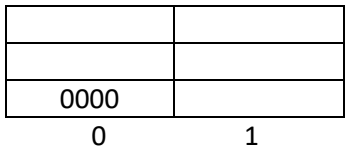
Ρίζα)	1 κόμβος = 101 εγγραφές
2ο επίπεδο)	102 κόμβοι = 102*101 = 10.302 εγγραφές
Φύλλα)	102*102 = 10.404 κόμβοι = 10.404*101 = 1.050.804 εγγραφές

δηλαδή **1.050.804 εγγραφές** της σχέσης R

Άσκηση 5)

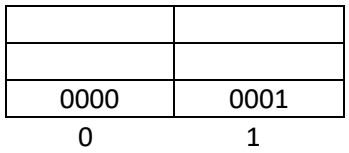
- $3\ ^{\text{εγγραφές}}/_{\text{σελίδα}}$
- Αρχικά m=1, i=1
- Μέγιστο Utilization = 70%

1. Εισαγωγή 0000



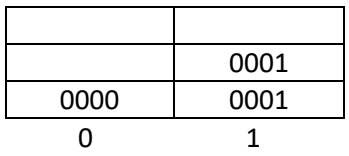
Utilization = 16,5%

2. Εισαγωγή 0001



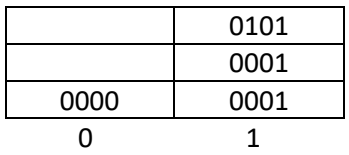
Utilization = 33%

3. Εισαγωγή 0001



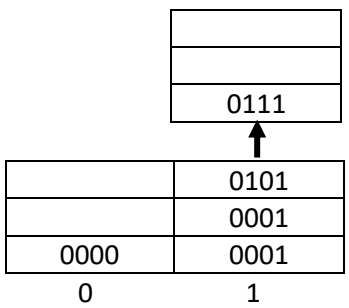
Utilization = 50%

4. Εισαγωγή 0101



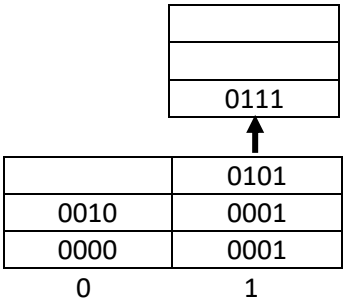
Utilization = 66,5%

5. Εισαγωγή 0111



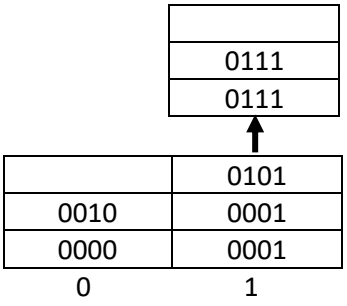
Utilization = 5/9 = 55%

6. Εισαγωγή 0010



Utilization = 6/9 = 66%

7. Εισαγωγή 0111

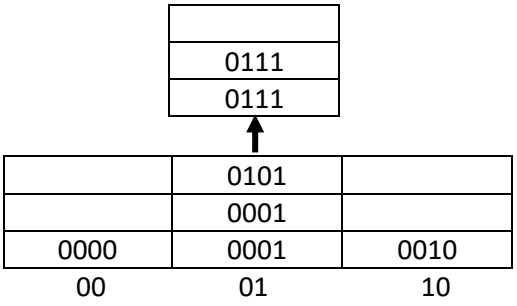


Utilization = 7/9 = 77% > 70%

Χρειάζεται να αυξήσουμε τους κάδους.

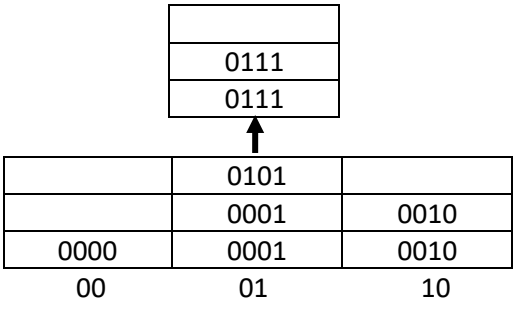
Το m αυξάνεται, **m=10**. Εφόσον το m ξεπερνάει το $2^i - 1$, το i αυξάνεται άρα **i=2**

Το αρχείο μεταβάλλεται ως εξής:



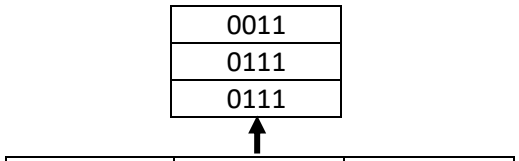
Utization = 7/12 = 58%

8. Εισαγωγή του 0010



Utilization = 8/12 = 66%

9. Εισαγωγή 0011



Utilization = 9/12 = 75% > 70%

Αυξάνουμε το αρχείο προσθέτοντας ένα νέο bucket.

m = 11, **i=2**.

	0101		0011
	0001	0010	0111
0000	0001	0010	0111
00	01	10	11

Utilization = 9/12 = 75% > 70%

Παρατηρούμε πως αν και προσθέσαμε ένα καινούργιο κάδο, το utilization παρέμεινε σταθερό καθώς μετά από την αναδιοργάνωση του αρχείου δεν υπάρχει πλέον η σελίδα υπερχείλισης. Άρα αυξάνουμε εκ νέου το μέγεθος.

m = 100, i=3.

	0101		0011	
	0001	0010	0111	
0000	0001	0010	0111	
000	001	010	011	100

Utilization = 9/15 = 60%

10. Εισαγωγή 0100

	0101		0011	
	0001	0010	0111	
0000	0001	0010	0111	0100
000	001	010	011	100

Utilization = 10/15 = 66%

Άσκηση 6)

- Μέγεθος εγγραφής ευρετηρίου = 4 Bytes
- Μέγεθος εγγραφής = 400 Bytes
- Μέγεθος κάδου = 2400 Bytes
- Ολικό βάθος d = 10
- $\frac{2400}{400} = 6\ {}^{\text{εγγραφές}}/_{\text{κάδο}}$

a) Το ολικό βάθος είναι 10, που σημαίνει ότι χρησιμοποιούνται 10 bits για τον προσδιορισμό των κάδων, άρα υπάρχουν συνολικά $2^{10} = 1024$ εγγραφές του ευρετηρίου που δείχνουν προς του κάδους .

b) Το μέγεθος του ευρετηρίου θα είναι $1024 * 4 = 4096$ Bytes = 4 KB

c) Κάθε κάδος του ευρετηρίου μπορεί να έχει έως 6 εγγραφές. Αν χρησιμοποιούνται στο μέγιστο και οι 1024 διαθέσιμοι κάδοι, το αρχείο μπορεί να έχει έως $1024 * 6 = 6044$ εγγραφές.

Άσκηση 7)

Η επιλογή της συγκεκριμένης συνάρτησης κατακερματισμού δύναται να δημιουργήσει προβλήματα καθώς στην πράξη μπορεί να κατανείμει ανομοιόμορφα τα κλειδιά του γνωρίσματος στους κάδους. Αυτό αποτελεί πρόβλημα ιδιαίτερα στην περίπτωση όπου μεγαλύτερο ποσοστό των υπαλλήλων έχουν τον ίδιο μισθό ή όταν οι χαμηλόμισθοι υπάλληλοι είναι αρκετά περισσότεροι από τους υψηλόμισθους (όπως συμβαίνει σε αρκετές επιχειρήσεις). Κάτι τέτοιο θα έχει ως επακόλουθο όχι μόνο να υπάρχουν αρκετές συγκρούσεις, αλλά και ορισμένοι λίγοι κάδοι να έχουν δυσανάλογα περισσότερες εγγραφές σε σχέση με τους υπόλοιπους, αφού η πιθανότητα το hash ενός υπαλλήλου να αντιστοιχεί σε έναν από τους πρώτους κάδους θα είναι αρκετά μεγαλύτερη. Έτσι το ευρετήριο θα επεκτείνεται συχνά καθώς προστίθενται σταδιακά πιο πολλές εγγραφές στους λίγους αυτούς κάδους, με αποτέλεσμα μεγαλύτερο κόστος σε μνήμη. Γίνεται λοιπόν σαφές ότι η συνάρτηση αυτή δεν είναι κατάλληλη και πρέπει να επιλεγεί κάποια η οποία θα πετυχαίνει μια πιο ομοιόμορφη κατανομή.