ΥΣΒΔ

Εργασία 1

Ονοματεπώνυμο: Γεωργόπουλος Ιωάννης
A.M.:1115201800026

Ονοματεπώνυμο: Δαμιανάκης Δαμιανός
A.M.:1115201800306

HP:

Γενική υλοποίηση:
Η HP αποτελείται από ένα header block που έχει το HP_info και από τα υπόλοιπα που έχουν records μέσα (μέχρι 4) άλλα και άλλες χρήσιμες πληροφορίες για το block όπως τον αριθμό των records μέσα στο block, τον αριθμό των bytes που απομένουν μέσα στο block και την θέση του επόμενου block αν υπάρχει. Αν δεν υπάρχει επόμενο block τότε η μεταβλητή θα έχει την τιμή -1.

Συναρτήσεις:
HP_CreateFile: Στο HP δημιουργούμε αρχικά ένα info header block που βρίσκεται στην θέση 0 των blocks. Αυτό περιέχει το HP info του αρχείου. Επίσης έχω προσθέσει και μια μεταβλητή (nextBlock) που περιέχει την θέση του πρώτου μπλοκ με records.

HP_OpenFile: Η open file ανοίγει το πρώτο μπλοκ του αρχείου, αντιγράφει τον HP_iinfo σε ένα δείκτη και τον επιστρέφει.

HP_CloseFile: Η close file απλά κλείνει το αρχείο.

HP_InsertEntry: Αυτή η συνάρτηση ξεκινάει ψάχνοντας με τη βοήθεια της HP_GetAllEntries αν υπάρχει ήδη κάποιο record μέσα στο αρχείο με το ιδιο id με το record που θέλουμε να βάλουμε. Αν ήδη υπάρχει τότε εκτυπώνω αντίστοιχο μήνυμα. Αλλιώς ξεκινώ την διαδικασία για να βάλω το record στο αρχείο. Αρχικά ανοίγω το header block για να πάρω τη θέση του πρώτου block με records. Αν δεν υπαρχει τότε το φτιάχνω και βάζω το record εκεί επίσης βαζω τη θέση αυτο του block που έφτιαξα στη μεταβλητή nextBlock του header block. Αν υπάρχει το πρώτο block ξεκινώ να ψάχνω κάθε μπλοκ από το πρώτο ώς το τελευταίο για κένη θέση που μπορεί να μπει το record που θέλουμε να βάλουμε. Αν δεν υπάρχει θέση σε κανένα block φτιάχνουμε ένα καινούργιο και βάζουμε το record εκεί. Επίσης στο τελευταίο μπλοκ πριν φτιάξουμε το καινούργιο βαζουμε στη μεταβλητή nextBlock την θέση του καινούργιου block που μόλις φτιάξαμε. Τέλος όπου και να μπεί το record επιστρέφουμε την θέση του block στο οποίο μπήκε.

HP_DeleteEntry: Αυτή η συνάρτηση ξεκινά ψαχνοντας με τη βοήθεια της

HP_GetAllEntries αν υπάρχει record με το δοσμένο value για id. Αν δεν υπάρχει αυτό σημαίνει ότι δεν μπορεί να σβηστεί οπότε εκτυπώνεται αντίστοιχο μήνυμα. Αν υπάρχει η HP_GetAllEntries θα επιστρέψει το id του block στο οποίο βρίσκεται το record. Η delete θα πάρει αυτό το id και θα ανοίξει το αντίστοιχο block. θα ψάξει μέσα στο block για το record. Όταν το βρει θα κοιτάξει αν βρίσκεται στην τελευταία θέση του block. Αν ναι θα μειώσει τον αριθμό των records στο block και θα αυξήσει το χώρο που υπολείπεται στο block κατα το μέγεθος του record. Με αυτό τον τρόπο ουσιαστικά το record σβήνεται. Αν όμως το record δεν βρίσκεται στην τελευταία θέση του block τότε παίρνω το τελευταίο record του block και το βάζω στη θέση αυτού που θέλω να σβήσω. Μειώνω τώρα τον αριθμό τον records στο block και αυξάνω την χωρητικότητα του block κατα το μέγεθος του record. Τέλος επιστρέφω 0 ως τιμή επιτυχίας. Αν για κάποιο λόγο δεν βρει το record στο block επιστρέφει -1.

HP_GetAllEntries: Αυτή η συνάρτηση ξεκινάει ανοίγοντας το header block και παίρνοντας την θέση του πρώτου block. Αν αυτή είναι -1 αυτό σημαίνει ότι το αρχείο δεν έχει records άρα επιστρέφει -1. Αν υπάρχει το πρώτο block ξεκινώ να ψάχνω κάθε μπλοκ από το πρώτο ώς το τελευταίο και κάθε record μέσα σε αυτά τα block. Αν βρω το record τότε επιστρέφω το το id του block στο οποίο βρίσκεται.

HT:

Γενική υλοποίηση:

Η HT αποτελείται από ένα header info block που έχει το HT_info το οποίο δείχνει στο πρώτο header block. Τα header blocks περιέχουν τις διευθυνσεις τών buckets αλλά και τον αριθμό των bytes στο block και τη θέση του επόμενου header block. Επίσης ο συνολικος αριθμός των buckets που μπορουν να αντιπροσωπεύει ένα header block είναι (512-8)/4=126 όπου 8 είναι ο χώρος που παίρνουν τα αλλα στοιχεία του header block, 512 ο συνολικός χώρος του header block και 4 το μέγεθος ενός integer. Τέλος υπάρχουν και τα buckets όπου έχουν 1 η περισσότερα blocks. Σε αυτά τα blocks των buckets αποθηκεύονται τα records και άλλες χρήσιμες πληροφορίες για το block όπως τον αριθμό των records μέσα στο block, τον αριθμό των bytes που απομένουν μέσα στο block, το κλειδί του bucket και την θέση του επόμενου block αν υπάρχει. Αν δεν υπάρχει επόμενο block τότε η μεταβλητή θα έχει την τιμή -1.

Συναρτήσεις:

HT_CreateIndex: Στο HT δημιουργούμε αρχικά ένα info header block που βρίσκεται στην θέση 0 των blocks. Αυτό περιέχει το HT_info του αρχείου. Επίσης έχω προσθέσει και μια μεταβλητή (first_header_block) που περιέχει την θέση του πρώτου header block. Έπειτα δημιουργεί το πρώτο header block και βαζει τη θέση του στη μεταβλητή firtst_header_block του header info block που φτιαχτηκε προηγουμένως. Έπειτα φτιάχνω όσα buckets χρειάζονται ανάλογα με τα buckets που έχει δώσει ο χρήστης από την γραμμή εντολών και βάζω τις θεσεις των buckets στα header blocks επειδή όπως είπαμε το κάθε header block χωράει 126 θέσεις buckets αν δεν χωράνε οι θέσεις των buckets στο 1 header block δημιουργούνται και άλλα header blocks. Όταν γίνεται αυτό βάζω την θέση του καινούριου header block και στη μεταβλητή nextHeaderBlock του προηγούμενου.

HT_OpenIndex: Η open index ανοίγει το πρώτο μπλοκ του αρχείου, αντιγράφει τον HT_info σε ένα δείκτη και τον επιστρέφει.

HT_CloseIndex: Η close index απλά κλείνει το αρχείο.

HT_InsertEntry: Αυτή η συνάρτηση ξεκινάει ψάχνοντας με τη βοήθεια της HT_GetAllEntries αν υπάρχει ήδη κάποιο record μέσα στο αρχείο με το ιδιο id με το record που θέλουμε να βάλουμε. Αν ήδη υπάρχει τότε εκτυπώνω αντίστοιχο μήνυμα. Αλλιώς ξεκινώ την διαδικασία για να βάλω το record στο αρχείο. Συνεχίζουμε υπολογίζοντας το κλειδί του value που δόθηκε. Παίρνοντας το div, του κλειδιού δια το 126 (αριθμό θέσεων bucket που υποστηρίζει ένα header block) παίρνουμε ποιο header block έχει τη θέση του bucket με κλειδι το key που υπολογίσαμε. Και έπειτα παίρνουμε το key mod 126 για να δούμε που μέσα στο header block βρίσκεται η θέση του bucket με κλειδι το key που υπολογίσαμε. Αφου βρούμε τη θέση του bucket που αντιστοιχεί στο κλειδί ξεκινώ να ψάχνω κάθε μπλοκ μέσα στο bucket για κένη θέση που μπορεί να μπει το record που θέλουμε να βάλουμε. Αν δεν υπάρχει θέση σε κανένα block του bucket τότε φτιάχνουμε ένα καινούργιο και βάζουμε το record εκεί. Επίσης στο τελευταίο μπλοκ του bucket πριν φτιάξουμε το καινούργιο βαζουμε στη μεταβλητή nextBlock την θέση του καινούργιου block που μόλις φτιάξαμε. Τέλος όπου και να μπεί το record επιστρέφουμε την θέση του block στο οποίο μπήκε.

HT_DeleteEntry: Αυτή η συνάρτηση ξεκινά ψαχνοντας με τη βοήθεια της HT_GetAllEntries αν υπάρχει record με το δοσμένο value για id. Αν δεν υπάρχει αυτό σημαίνει ότι δεν μπορεί να σβηστεί οπότε εκτυπώνεται αντίστοιχο μήνυμα. Αν υπάρχει η HT_GetAllEntries θα επιστρέψει το id του block στο οποίο βρίσκεται το record. Η delete θα πάρει αυτό το id και θα ανοίξει το αντίστοιχο block. θα ψάξει μέσα στο block για το record. Όταν το βρει θα κοιτάξει αν βρίσκεται στην τελευταία θέση του block. Αν ναι θα μειώσει τον αριθμό των records στο block και θα αυξήσει το χώρο που υπολείπεται στο block κατα το μέγεθος του record. Με αυτό τον τρόπο ουσιαστικά το record σβήνεται. Αν όμως το record δεν βρίσκεται στην τελευταία θέση του block τότε παίρνω το τελευταίο record του block και το βάζω στη θέση αυτού που θέλω να σβήσω. Μειώνω τώρα τον αριθμό τον records στο block και αυξάνω την χωρητικότητα του block κατα το μέγεθος του record. Τέλος επιστρέφω 0 ως τιμή επιτυχίας. Αν για κάποιο λόγο δεν βρει το record στο block επιστρέφει -1.

HT_GetAllEntries: Αυτή η συνάρτηση ξεκινάει υπολογίζοντας το κλειδί του value που δόθηκε. Παίρνοντας το div του κλειδιού δια το 126 (αριθμό θέσεων bucket που υποστηρίζει ένα header block) παίρνουμε ποιο header block έχει τη θέση του bucket με κλειδι το key που υπολογίσαμε. Και έπειτα παίρνουμε το key mod 126 για να δούμε που μέσα στο header block βρίσκεται η θέση του bucket με κλειδι το key που υπολογίσαμε.Όταν βρούμε τη θέση του bucket ψάχνουμε σε όλα τα blocks του bucket για το record με id == value. Αν βρω το record τότε επιστρέφω το το id του block στο οποίο βρίσκεται. Αν δεν υπάρχει σε κανένα από τα blocks του bucket τότε επιστρέφω -1.

Μεταγλώττιση και εκτέλεση:

Για την μεταγλώττιση πρέπει στο φάκελο που είναι το Makefile να πατήσετε make.
Αυτό μεταγλωτίζει τον κώδικα και το εκτελέσιμο αρχείο τοποθετείται στο φάκελο bin.
Να σημειωθεί ότι η HT και η HP μεταγλωτίζονται μαζί και μετά για την εκτέλεση του προγράμματος ο χρήστης επιλέγει την μέθοδο από τη γραμμή εντολών. Ειδικότερα για την εκτέλεση του HP εκτελείται η εντολή "./bin/main (int numOfRec) HP" όπου το (int numOfRec) είναι ένας ακέραιος και συγκεκριμένα μπορεί να πάρει τιμές (1,5,10,15) και καθορίζει το αρχείο από το οποίο θα διαβάσει το πρόγραμμα μας. Δηλαδή αν θέλω να τρέξω το πρόγραμμα με αρχείο εισόδου /examples/records1K.txt και μέθοδο HP η εντολή θα είναι "./bin/main 1 HP" αν θέλω το αρχείο εισόδου /examples/records5K.txt με μέθοδο HP θα γράψω ./bin/main 5 HP . Για να τρέξω την HT θα κάνω το ίδιο αλλά θα προσθέσω ένα ακόμα όρισμα που είναι ο αριθμός των buckets. Δηλαδή αν θέλω να τρέξω το πρόγραμμα με μέθοδο HT με αρχείο εισόδου το /examples/records10K.txt και 500 buckets θα γράψω ./bin/main 10 HT 500 . Το τελευταίο όρισμα (500) είναι ο αριθμός των bucket και πρέπει να είναι ακέραιος αριθμός θετικος.