**Project: Αλγόριθμοι και Δομές Δεδομένων**

Θεωρούμε ένα supermarket στο οποίο προσέρχονται πολλοί πελάτες για τα ψώνια τους. Κάθε πελάτης μπορεί να αγοράσει διάφορα προϊόντα και σε διαφορετικές ποσότητες το καθένα. Ο διευθυντής του supermarket αποφάσισε να παίρνει ο κάθε πελάτης πόντους ανάλογα με τα προϊόντα και τα χρήματα που τους αντιστοιχούν για να επιδοτήσει έτσι τους τακτικούς πελάτες. Με τη συμπλήρωση διαφόρων ορίων σε πόντους ο πελάτης κερδίζει κάποια δώρα. Επίσης, εισήχθηκε και ο θεσμός των καρτών μιας οικογένειας ή ομάδας, δηλαδή αν πολλές κάρτες που ανήκουν σε μία ομάδα ή οικογένεια μαζέψουν όλοι μαζί τους συγκεκριμένους πόντους που αντιστοιχούν σε κάποιο δώρο, μπορούν να το αποκτήσουν. Ζητείται να γράψετε ένα πρόγραμμα το οποίο να παίρνει σαν είσοδο τα εξής αρχεία:

**Αρχείο 1 (περίπου 500000 γραμμές):**

|  |
| --- |
| (1...7) ΒΓΔΕ0134Α256789P1,2;P2,4; …  .  .  . |

Σε κάθε γραμμή υπάρχει **η ημέρα της εβδομάδας** που έγινε η αγορά, **ο κωδικός της κάρτας** και **μετά από ένα** **κενό τα προϊόντα** που αγόρασε καθώς και **οι ποσότητες** στις οποίες τα αγόρασε. Προϊόν και ποσότητα χωρίζονται **με κόμμα**, ενώ τα προϊόντα μεταξύ τους με ερωτηματικό. Ο κωδικός της κάρτας είναι 15-ψήφιος και αποτελείται από αριθμούς και γράμματα. Ένας πελάτης και συνεπώς μια κάρτα μπορεί να εμφανιστεί πολλές φορές στο κατάστημα.

**Αρχείο 2:**

|  |
| --- |
| A:χ2:P1,P6,P9, …  B:χ3:P2,P7, …  C:χ1:P3,P8, … |

Σε κάθε γραμμή αυτού το αρχείου υπάρχουν **ομάδες προϊόντων** και κάποια προϊόντα ανήκουν σε κάποιες από αυτές τις ομάδες. Ανάλογα με την κατηγορία που ανήκει ένα προϊόν έχει και διαφορετικό **συντελεστή πόντων** (δεύτερη στήλη) με τον οποίο πολλαπλασιάζονται τα συνολικά χρήματα που ξόδεψε για το συγκεκριμένο προϊόν.Έτσι υπολογίζονται οι πόντοι του κάθε πελάτη. Αν ένα προϊόν δεν ανήκει σε κάποια ομάδα ο συντελεστής πόντων είναι 0,5.

**Αρχείο 3:**

|  |
| --- |
| P1 20 1 5 7  P1 22 2 3  P2 35 5  .  .  . |

Σε κάθε γραμμή υπάρχουν **τα προϊόντα** και η **τιμή μονάδας του κάθε προϊόντος** σε ευρώκαθώς και **οι ημέρες της εβδομάδας για την τιμή αυτή**. Εάν υπάρχουν συγκρούσεις ισχύει η τελευταία τιμή και εάν δεν υπάρχει πληροφορία τιμής για κάποια ημέρα της εβδομάδας ισχύει η ακριβότερη τιμή! Εάν δεν υπάρχει ένα προϊόν η τιμή μονάδος του προϊόντος να είναι 10!

Με βάση αυτά τα 3 αρχεία πρέπει να αποθηκεύσετε τα δεδομένα σε έναν hashtable(πίνακα κατακερματισμού)μεγέθουςπερίπου 2\*Ν (must be a prime number!),όπου Ν είναι ο αριθμός των διαφορετικών καρτών που έχουμε στο αρχείο, με την διαδικασία του “LINEARPROBING”. Τα ερωτήματα προς απάντηση είναι:

* 10 κάρτες (κωδικούς) που συγκέντρωσαν τους περισσότερους πόντους
* Προϊόντα που δικαιούται σαν δώρα μία συγκεκριμένη ομάδα καρτών

Για το ερώτημα αυτό θα πρέπει την ώρα της εξέτασης να φορτώσετε 2 νέα αρχεία που θα έχουν την εξής μορφή:

**Αρχείο 4:**

|  |
| --- |
| ΑΒΓΔΕ0134627349  ΑΒΓΔΕ1234567890  0129873456ΑΒΓΔΕ  .  .  . |

Οι κωδικοί των καρτών που απαρτίζουν την ομάδα.

**Αρχείο 5:**

|  |
| --- |
| G1 50  G2 100  .  .  . |

Σε κάθε γραμμή φαίνεται πόσους πόντους χρειάζεται να συγκεντρώσει ο πελάτης για να πάρει το αντίστοιχο δώρο.

* Αριθμός συγκρούσεων στον πίνακα κατακερματισμού για 3 διαφορετικές συναρτήσεις κατακερματισμού!

**Παράδοση του projectμαζί με 4-σέλιδη αναφορά: 21/5/2013**

(λεπτομέρειες για την σχεδίαση και τους αλγορίθμους σε όλα τα θέματα της εργασίας π.χ. hashing function, sorting methods, data structures, I/O functions, …)

Η πρώτηγραμμήείναι ο βαθμός στο projectκαι η δεύτερηγραμμήείναι η αντίστοιχη επιρροή του στην τελικήβαθμολογία του μαθήματοςμαζί με το γραπτό τελικό διαγώνισμα το οποίοπρέπει να είναιτουλάχιστον 5 για να μετρήσει και το project.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| -2,5 | -2,0 | -1,5 | -1 | -0,5 | 0 | 0,5 | 1 | 1,5 | 2 | 2,5 |

Πρωτότυπο συνάρτησης Περιγραφή συνάρτησης

double atof( const char \*nPtr ); Μετατρέπει τη συμβολοσειράn Ptr σε double.

int atoi( const char \*nPtr ); Μετατρέπει τη συμβολοσειρά string nPtr σε int.

long atol( const char \*nPtr ); Μετατρέπει τη συμβολοσειρά nPtr σε long int.

double strtod( const char \*nPtr, char \*\*endPtr ); Μετατρέπει τη συμβολοσειρά nPtr σε double.

**Συναρτήσεις της** <stdio.h>

Πρωτότυπο συνάρτησης Περιγραφή συνάρτησης

int getchar( void );Λαμβάνει τον επόμενο χαρακτήρα από την standard είσοδο και τον επιστρέφει σαν έναν ακέραιο.

char \*gets( char \*s );Λαμβάνει χαρακτήρες από την standard είσοδο,

τους εισάγει στον πίνακα s μέχρι να συνατήσει ένα χαρακτήρα αλλαγής γραμμής ή τέλους αρχείου (end-of-file character). Ένας κενός χαρακτήρας τέλους επισυνάπτεται στον πίνακα.

int putchar( int c ); Τυπώνει το χαρακτήρα που είναι αποθηκευμένος στη μεταβλητή c.

int puts( const char \*s );Τυπώνει τη συμβολοσειρά (το αλφαριθμητικό)s ακολουθούμενη από ένα χαρακτήρα αλλαγής γραμμής.

int sprintf( char \*s, const char \*format, ... ); Ισοδύναμη με την printf, με τη διαφορά ότι η έξοδος αποθηκεύεται στον πίνακα s αντί να τυπωθεί στην οθόνη.

int sscanf( char \*s, const char \*format, ... );Ισοδύναμη με τη scanf, με τη διαφορά ότι η είσοδος «διαβάζεται» (λαμβάνεται) από τον πίνακα s αντί να διαβαστεί από το πληκτρολόγιο.

char \*strtok( char \*s1, const char \*s2);Διαδοχικές κλήσιες της strtok έχουν ως αποτέλεσμα τον κατακερματισμό του s1 στα λεγόμενα “tokens”— λογικά τμήματα όπως λέξεις σε μια γραμμή κειμένου — που διαχωρίζονται από χαρακτήρες που περιέχονται στο αλφαριθμητικό s2. Η πρώτη κλήση της strtok περιέχει το s1 ως το πρώτο όρισμα, και διαδοχικές κλήσεις για τον κατακερματισμό του ίδιου αλφαριθμητικού περιλαμβάνουν το NULL ως το πρώτο όρισμα. Με κάθε κλήση επιστρέφεται ένας δείκτης στο τρέχον κομμάτι (token). Όταν δεν υπάρχουν πλέον περισσότερα κομμάτια (tokens) όταν καλείται η υνάρτηση, επιστρέφεται NULL.

/\* Synarthsh poy diabazei to prwto arxeio me ta basika dedomena \*/

void diabase\_arxeio\_1()

/\* Sth **cl\_ptr** tha sosoyme oloys toys kwdikoys gia na doyme poioi einai monadikoi. \*/

cl\_ptr = **init\_code\_vector**(MAX\_CODES);

**init\_code\_vector**( Arxikopoiei mia domh poy anaparista pinaka apo kwdikoys, me bash to zhtoymeno plithos kwdikwn )

**cl\_ptr** = malloc(sizeof(struct code\_vector));

struct code\_vector {

int length;

code\_string \* codes;// char code\_string[CODE\_STRING\_LENGTH+1];

};

**cl\_ptr->codes** = malloc(sizeof(code\_string)\*num\_codes);

return cl\_ptr;

/\* Epilogh onomatos arxeioy kai anoigma toy gia anagnwsh \*/

printf("\n1.Ypologismos arithmoy diaforetikwn kwdikwn gia kartes.... ");

\* Agnooyme an yparxoyn kenes grammes

\* Swzoyme th grammh se mia domh product\_data

data = **analyze**(line);

\* Hmera ebdomadas.

\* kwdikos kartas

\* Lista agorasmenwn proiontwn

\* An den exei tipota eimaste OK

\* Alliws prepei na doyme posa diaforetika proionta dhlwnontai

\* Metrame ton arithmo apo ';' poy emfanizontai sto trexon str

\* kai to swzoyme sto p\_list.length \*/

\* Desmeyoyme xwro gia ta proionta (kwdikos, posothta)

**result.p\_list.products** = calloc(1, sizeof(struct product) \* result.p\_list.length);

/\* Apothikeyoyme ton kwdiko gia epeksergasia meta. \*/

**append\_code**(cl\_ptr, data.code);//Prosartei ena kwdiko se mia domh

/\* Boithitikh synarthsh sygkrishs gia klhsh ths qsort \*/

**int codecmp**(const void \* p1, const void \* p2) {

return strcmp( (const char \*)p1, (const char \*)p2);

}

/\* Ypologismos twn monadikwn kwdikwn se mia domh code\_vector \*/

**int find\_unique\_codes**(struct code\_vector \* cl\_ptr) {

int i;

int num\_unique = 0;

/\* Kanoyme taksinomhsh sth lista twn kwdikwn \*/

**qsort**(cl\_ptr->codes, cl\_ptr->length, sizeof(code\_string), codecmp);

/\* Elegxoyme apo 0 ews N-1 an enas kwdikos X[i] me ton X[i+1] einai

\* diaforetikoi, kai oso einai diaforetikoi ayksanoyme to num\_unique, alliws

\* paraleipoyme tis epanalipseis. \*/

for (i=0; i<cl\_ptr->length-1; i++){

num\_unique++;

while( (i < cl\_ptr->length-1) && (strcmp(cl\_ptr->codes[i], cl\_ptr->codes[i+1]) == 0) )

i++;

}

/\* Diorthonoyme th metrhsh gia toys kwdikoys N-1, N \*/

if (strcmp(cl\_ptr->codes[i], cl\_ptr->codes[i+1]) != 0)

num\_unique++;

return num\_unique;

}

free(data.p\_list.products);

}printf("Egine\n");

/\* Ypologismos twn monadikwn kwdikwn se mia domh code\_vector \*/

**Unic= find\_unique\_codes**(struct code\_vector \* cl\_ptr)

/\* Kanoyme taksinomhsh sth lista twn kwdikwn \*/

**qsort**(cl\_ptr->codes, cl\_ptr->length, sizeof(code\_string), codecmp);

/\* Elegxoyme apo 0 ews N-1 an enas kwdikos X[i] me ton X[i+1] einai

\* diaforetikoi, kai oso einai diaforetikoi ayksanoyme to num\_unique, alliws

\* paraleipoyme tis epanalipseis. \*/

/\* Ypologismos megethoys pinaka katakermatismoy (>2\*N kai prwtos)\*/

hashlen = **compute\_hash\_length**(unique);

/\*init\_hash\_table()-Arxikopoiei to hash table mas, na exei hashlen xwrhtikothta \*/

my\_hash = **init\_hash\_table**(hashlen);

/\* Fortwma dedomenwn ston pinaka katakermatismoy \*/

printf("\n1.Ypologismos arithmoy diaforetikwn kwdikwn gia kartes.... ");

Diabasma dedomenwn arxeioy 1

{

\* Diabasma mias grammhs apo to arxeio sth metablhth line

\* Agnooyme an yparxoyn kenes grammes

\* Swzoyme th grammh se mia domh product\_data

data = **analyze**(line);

\* Apothikeyoyme ton kwdiko gia epeksergasia meta.

**put\_hash\_table**(data.code, data);

{

/\* Ypologizoyme poy prepei na mpoyn ta data \*/

index = **hash\_func**(code);//synarthsh katakermatismoy

/\* Einai o kwdikos sto hash\_table[index] idios me ton dedomeno? \*/

/\* Bres me linear probing adeia thesh.

\* Oso o apothikeymenos kwdikos den einai to keno string,

\* aykshse to index kai an kseperasei to my\_hash->length,

\* pame pali apo to 0.

}

}

printf("Egine\n")

void diabase\_arxeio\_2()

/\* Epilogh onomatos arxeioy kai anoigma toy gia anagnwsh \*/

printf("\n2.Diabasma arxeioy(arxeio2.txt) omadwn proiontwn .....");

/\* Diabasma mias grammhs apo to arxeio sth metablhth line \*/

/\* Agnooyme an yparxoyn kenes grammes \*/

/\* Diabasma mias grammhs apo to arxeio sth metablhth line \*/

str = **strtok**(line, ":");

**strcpy**(team\_vector.data[k].team\_code, str);

str = **strtok**(NULL, ":");

**team\_vector.data[k].factor** = **atof**(str);

str = **strtok**(NULL, ":");

c = str;

while (\*c != '\0')

{

if (\*c == ',')

team\_vector.data[k].pcv.length++;

c++;

}

team\_vector.data[k].pcv.length++;

**team\_vector.data[k].pcv.codes**=**calloc**(team\_vector.data[k].pcv.length,sizeof(product\_code\_string); die\_if(team\_vector.data[k].pcv.codes == NULL, "Den exoyme mnhmh");

str = **strtok**(str, ",");

i = 0;

while (str != NULL)

{

strcpy(team\_vector.data[k].pcv.codes[i], str);

str = **strtok**(NULL, ",");

i++;

}

k++;

}

**printf("Egine\n\n");**

**void diabase\_arxeio\_3()**

/\* Epilogh onomatos arxeioy kai anoigma toy gia anagnwsh \*/

printf("\n3.Diabasma arxeioy(arxeio3.txt) me times proiontwn .....");

/\* Diabasma mias grammhs apo to arxeio sth metablhth line \*/

/\* Agnooyme an yparxoyn kenes grammes \*/

pd = **analyze\_price**(line);

**price\_vector\_add**(pd);

max\_price = price\_vector.data[i].price[day];

printf("Egine.\n");

void diabase\_arxeio\_4()

/\* Epilogh onomatos arxeioy kai anoigma toy gia anagnwsh \*/

printf("Diabasma arxeioy kwdikwn omadas pelatwn .....");

/\* Diabasma mias grammhs apo to arxeio sth metablhth line \*/

/\* Agnooyme an yparxoyn kenes grammes \*/

card\_team.codes = **realloc**(card\_team.codes, card\_team.length \* sizeof(code\_string));

strcpy(card\_team.codes[card\_team.length-1], line);

}

printf("Egine.\n");

void diabase\_arxeio\_5()

/\* Epilogh onomatos arxeioy kai anoigma toy gia anagnwsh \*/

printf("Diabasma arxeioy dedomenwn dwrwn .....");

/\* Diabasma mias grammhs apo to arxeio sth metablhth line \*/

/\* Agnooyme an yparxoyn kenes grammes \*/

prize\_vector.data = realloc(prize\_vector.data, prize\_vector.length \* sizeof(struct prize\_data));

printf("Egine.\n");

void bres\_dwra\_omadas()

{

double synolo\_pontwn;

int i;

synolo\_pontwn = 0;

printf("\n7.H omada poy apoteleitai apo tis kartes:\n");

printf("========================================\n");

for (i=0; i<card\_team.length; i++){

synolo\_pontwn += pontoi\_kartas(card\_team.codes[i]);

printf("%s\n", card\_team.codes[i]);

}

printf("Diakaioyta ta dwra: ");

for (i=0; i<prize\_vector.length; i++)

if (synolo\_pontwn > prize\_vector.data[i].required\_points)

printf("%s ", prize\_vector.data[i].prize);

printf("\n");

}

void ypologise\_sygkroyseis()

{

printf("\nH synarthsh apo to paradeigma hash\_func\n");

dokimase\_hash\_function( hash\_func );

printf("\nH synarthsh fnv\_32\_hash\n");

dokimase\_hash\_function( fnv\_32\_hash );

printf("\nH synarthsh djb2 hash\n");

dokimase\_hash\_function( hash );

}

void anazhthsh()

{

struct product\_data \* pd;

code\_string code;

char buf[BUFFER\_LENGTH];

int k;

if (my\_hash == NULL)

{

printf("Den exete fortosei to arxeio1!!!\n");

return;

}

printf("Kwdikos? ");

diabase\_keimeno(buf, sizeof(buf));

if (strlen(buf) != 15)

{

printf("Parakalw eisagete 15 xarakthrwn kwdiko.\n");

return;

}

strcpy(code, buf);

pd = get\_hash\_table(code);

if (pd == NULL)

{

printf("De brethike eggrafh me kwdiko %s\n", code);

return;

}

printf("Dokimh anakthshs. kwdikos: %s\n", code);

while (pd != NULL)

{

printf("Hmera: %d, Proionta:", pd->day\_of\_week);

for (k=0; k < pd->p\_list.length; k++)

printf("%s,%d;", pd->p\_list.products[k].pcode, pd->p\_list.products[k].quant);

printf("\n");

pd = pd->next;

}

}