Université d'Ottawa
Faculté de génie

École de science informatique
et de génie électrique

uOttawa

L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Electrical Engineering
and Computer Science

CSI 2132 and 2532 – Final Examination
Professor(s): Herna Viktor and Iluju Kiringa

25 April 2015
09:h30-12h30
Duration: 3 hrs

Closed book; no aid allowed, except one double-sided letter-size "cheat sheet". Answer all questions in **ink**. **Good luck! / Bonne Chance!**

Family name: _____

First name: _____

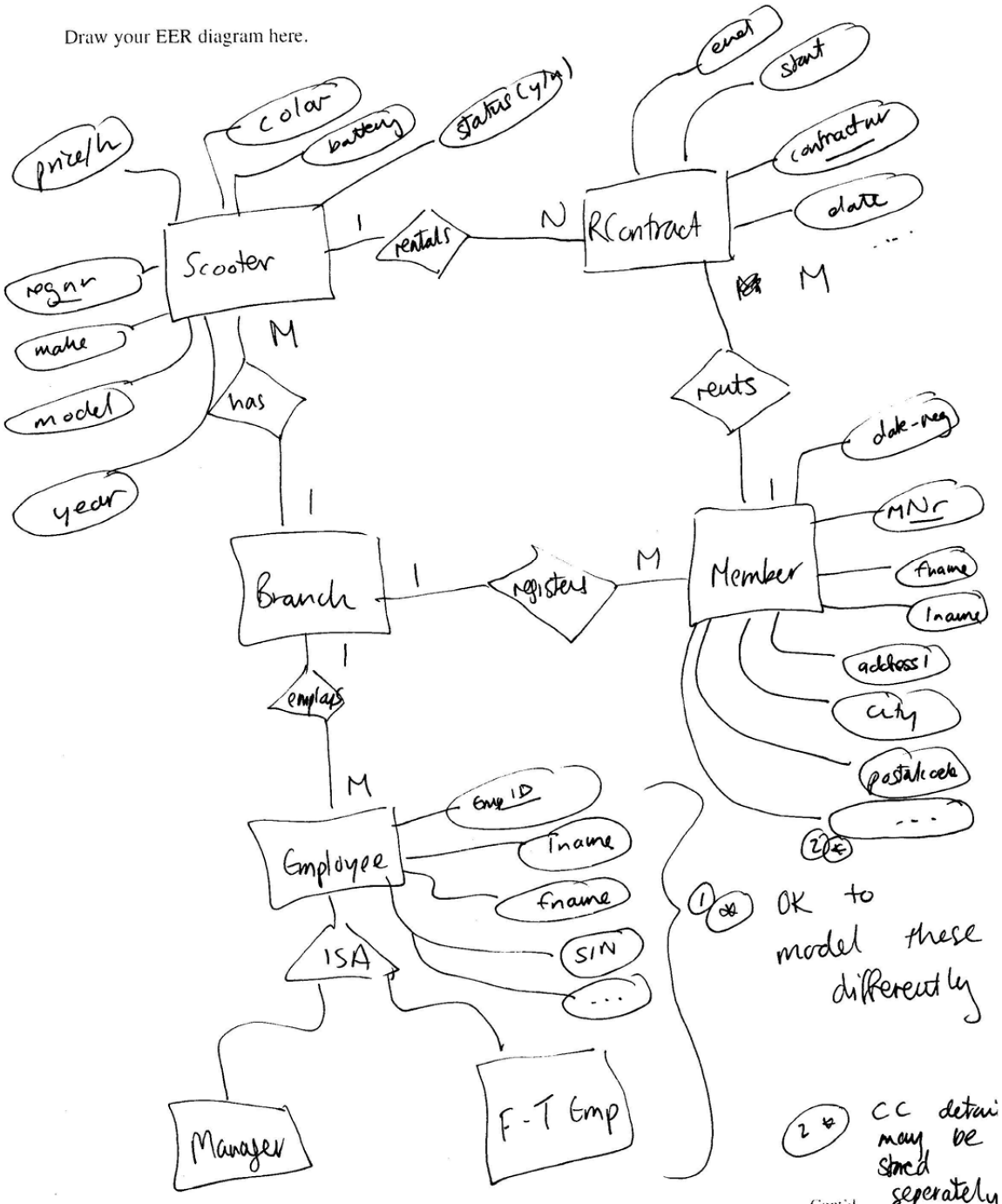
Student number: _____

There are 7 questions and a total of 100 points.

This exam must contain 15 pages,
including this cover page.

1 – EER Diagram	/ 15
2 – The Relational Model and SQL	/ 15
3 – Relational Algebra and Relational Calculus	/ 10
4 – Normal Forms	/ 15
5 – Disks, Files and Records	/ 15
6 – Storage and Indexing: B+ trees	/ 15
7 – Storage and Indexing: Extendible Hashing	/ 15
Total	/ 100

Draw your EER diagram here.



... Cont'd

2 The Relational Model and SQL — 15 points

Consider the following relational schema about Kids (Campers) that register for Summer Camps in Adventures in Science in Engineering at uOttawa.

Camper(*CName* : `string`, *Age* : ~~`string`~~ `integer`, *Email* : `string`, *Tshirt* : `string`, *Fee* : `real`)
Camp(*CampID* : `int`, *CampTitle* : `string`, *EmpID* : `int`, *StartDate* : `date`, *Year* : `date`)
Signup(*CName* : `string`, *CampID* : `int`)
Mentor(*EmpID* : `int`, *Name* : `string`, *EmploymentDate* : `date`, *Salary* : `currency`)

Answer the following questions.

Part A — 2 points Explain what a secondary key is and identify one in the above database schema.

An attribute that is used for retrieval purposes. It does not have to be unique.
Examples are CampTitle or Mentor.Name

Part B — 6 points Give the SQL statement to find the names and employment dates of the Mentors who earn less than any Mentor named Randy. Display this information together with the titles of the Camps these unfortunate Mentors ran (were responsible for), as well as the years they were Mentors.

```
SELECT   M.name, M.employmentDate, C.campTitle, C.year
FROM     Mentor M, Camp C
WHERE    M.EmpID = C.EmpID AND
         M.Salary <
            ( SELECT   MAX(E.Salary)
              FROM     MENTOR E
              WHERE    E.Name = 'Randy' ) ;
```

... Cont'd

Part C — 4 points Give the SQL statement to display the names, fees paid and ages of the youngest Camper(s) for all of those Camps that had an enrollment that was higher than 20. That is, these popular Camps were attended by at least 21 Campers.

```
SELECT  TEMP. Cname, TEMP. fee, Temp. MINAGE
FROM  ( SELECT  C. cname,  C. Age , C. fee
        FROM    Camper C,  Signup S
        WHERE   C. CName = S. CName  AND
                S. CampID  IN
            ( SELECT    R.CAMPID
              FROM    SIGNUP R
              -- GROUP  BY   R. CAMPID
                 HAVING  COUNT(*) > 20)) AS TEMP
  WHERE   TEMP. MINAGE =  ( SELECT   MIN(AGE)
                            FROM  TEMP);
```

Part D — 3 points Campers have to be at least 6 years old to enroll in a Camp. Provide the SQL code fragment to enforce this constraint.

```
Use  a   CHECK  OR  DOMAIN  CONSTRAINT:
     CHECK( Age >= 6)
```

(OR)

```
CREATE  DOMAIN   Ageval  INTEGER
        CHECK  ( value >=6) ;

        :

CREATE  TABLE (
              :
              Age   Ageval);
```

... Cont'd

3 Relational Algebra and Relational Calculus — 10 points

Reconsider the following extended relational schema about Kids (Campers) that register for Camps in Adventures in Science in Engineering.

Camper(*CName* : string, *Age* : string, *Email* : string, *Tshirt* : string, *Fee* : real)
Camp(*CampID* : int, *CampTitle* : string, *EmpID* : int, *StartDate* : date, *Year* : date)
Signup(*CName* : string, *CampID* : int)
Mentor(*EmpID* : int, *Name* : string, *EmploymentDate* : date, *Salary* : currency)

Part A — 7 points Provide the relational algebra statement to find the names, tshirt sizes and emails of the 10 year old Campers who never attended a Camp that was led by a Mentor named Sandy who earns a salary of $500.00.

ρ (TenCamps, Π_{CampID} (Signup $\bowtie_{Cname}$ ($\sigma_{Age=10}$ Camper)))

ρ (SandyCamps, Π_{CAMPID} (Camp $\bowtie_{EmpID}$ ($\sigma_{name='sandy' \wedge Salary=500}$ Mentor))

ρ (Result, $\Pi_{Cname, tshirt, email}$ (Camper $\bowtie_{CName}$ Signup $\bowtie_{CampID}$ (TenCamps − SandyCamps)))

Part B — 1 point State whether the following statement is **true** or **false**. Relational algebra is a declarative language, since we specify exactly what steps need ot be taken when a query is executed.

TRUE.

Part C — 2 points Explain the difference between tuple relational calculus and domain relational calculus.

TRC - Variables range over tuples.

$\{ P \mid \exists S \in Sailors \ (S.rating > 10) \wedge P.name = S.name \}$

DRC - Variables range over domain elements.

$\{ \langle N \rangle \mid \exists I, T, A \ (\langle I, N, T, A \rangle \in Sailors \wedge T > 10 \}$

... Cont'd

4 Formes Normales — 15 points

Considérez une relation R avec attributs $ABCD$ décrivant des livres académiques. Un exemple d'instance de R est montrée ci-bas:

BookTitle (A)	Distributor (B)	Topic (C)	Price (D)
Java Solutions	Prentice Hall	Programming	119.00
Topology	Independent	Math	120.00
Database Design	Addison W	Databases	115.00
C++	Wiley	Advanced Programming	119.00

Pour chacun des ensembles suivants de dépendances fonctionnelles, dites s'il est en 3NF ou en BCNF. Sinon, dites pourquoi.

(1) $C \longrightarrow D,\ C \longrightarrow A,\ B \longrightarrow C$

Not in 3NF. Reason: Transitive dependency: $B \to C,\ C \to AD$

Not in BCNF. Reason: BCNF $\subset$ 3NF

(2) $A \longrightarrow B,\ BC \longrightarrow D,\ A \longrightarrow C$

Not in 3NF. Reason: $BC \to D$ implies that BC does not contain a key

Not in BCNF. Reason: BCNF $\subset$ 3NF

(3) $AB \longrightarrow C,\ AB \longrightarrow D,\ C \longrightarrow A,\ D \longrightarrow B$

In 3NF.

Not in BCNF. Reason: There is no decomposition that would preserve $C \to A$.

... suite

5 Disks, Files and Records — 15 points

Reconsider the following relational schema about Kids (Campers) that register for Camps in Adventures in Science and Engineering.

Camper(*CName* : `string`, *Age* : `string`, *Email* : `string`, *Tshirt* : `string`, *Fee* : `real`)
Camp(*CampID* : `int`, *CampTitle* : `string`, *EmpID* : `int`, *StartDate* : `date`, *Year* : `date`)
Signup(*CName* : `string`, *CampID* : `int`)
Mentor(*EmpID* : `int`, *Name* : `string`, *EmploymentDate* : `date`, *Salary* : `currency`)

Suppose that the Camper table is organized as a heap file, and that it contains the records of a total of 20,000 kids (i.e. current and past Campers). A disk block has the capacity to store 1,000 records and the buffer pool contains 10 slots. On average, a Camper registers for two Camps in a year and attends the Camps for four years in a row.

Part A — 5 points Explain how you would use *one* of the two different heap file implementations, as discussed in class, in order to organize the pages of the Camper table on disk.

Student should explain either the linked list implementation or the directory file organization. Refer to textbook or slides.

... Cont'd

Part B — 10 points Assume that you wish to execute a query that displays all the personal information about the Campers in your database (i.e. `SELECT * FROM Camper`). Explain the exact process that is followed i) to locate the data on disk, ii) to transfer the data into the buffer and iii) to deal with potential buffer sizing issue.

In this query, we wants to read the entire Camper file from disk. This file contains data that span 20 pages on disk. Our buffer has the capacity to hold up to 10.

1) We will follow a file scan.
2) The first step is to locate the start of the file (first page) on disk.
3) Next, the disk manager will move the head (execute the command) to the correct position.
4) The data (1st page) is read into the buffer. The buffer manager will change the status to occupied.
5) This process will continue until the buffer is full.
6) In this case, a frame will be selected for replacement. If it is dirty, it needs to be written back. We use a replacement policy such as LRU, MRU, etc.
7) This process continues until all the data in the file has been read.

... Cont'd

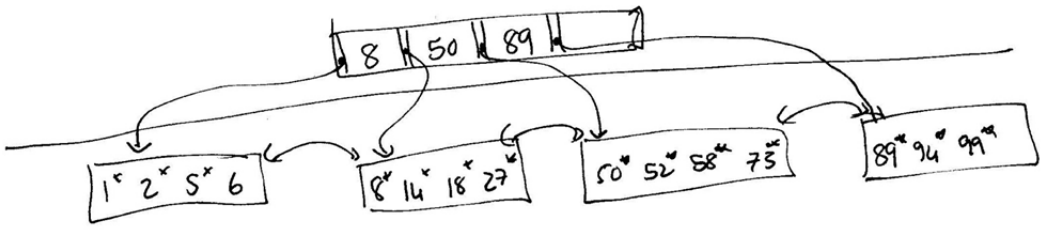
6 Storage and Indexing: B+ trees — 15 points

Consider the following table that contains the information about the prices (in $) and ratings of products sold in a convenience store.

ProductID	Type	Price	Rating
120	Gum	1	5
121	Camera	50	2
122	Candy	8	4
123	7Up 24 Box	18	5
124	Kitkat	2	3
125	Coke 24 Box	18	5
126	Nachos	6	4
127	Hat	73	5
128	Jacket	94	5
129	Boots	99	4
130	Backpack	27	3
131	Camera	52	5
132	Walking Stick	58	2
133	Aero	5	4
134	Cheese	14	2
135	Carpet	89	4

Assume that the DBMS constructs a B+ tree with an order d of 2 on the Price attribute. You decide to store the data entries using Alternative 3.

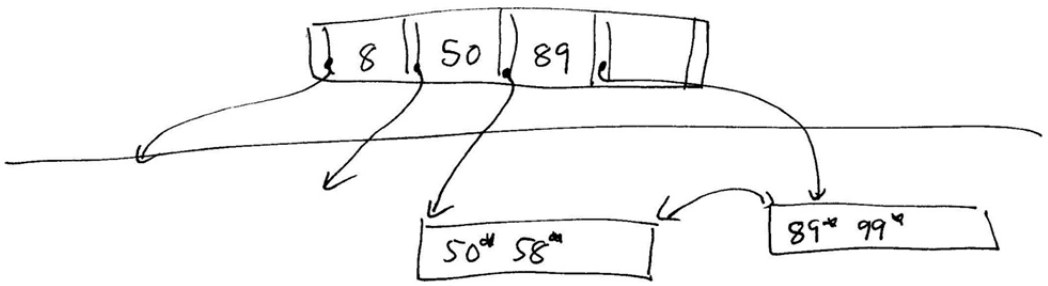
Part A — 10 points Show the final tree after you have inserted **all** the data using the bulk loading algorithm.



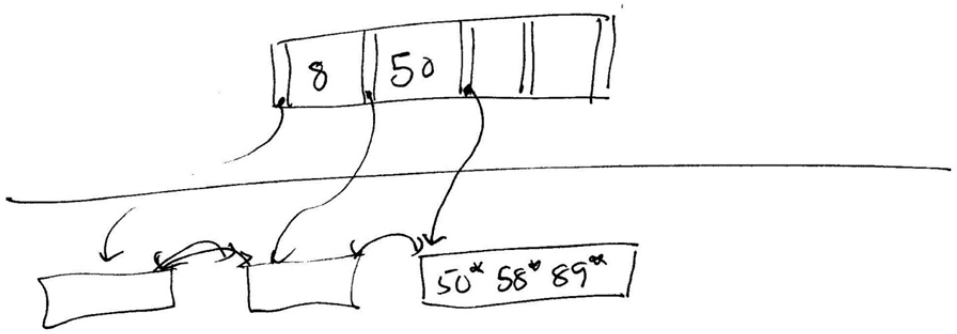
... Cont'd

Part B — 5 points Suppose that the products with the following ProductIDs are removed from the tree, in this order: 131, 128, 127 and 129. Show the resultant tree after you have deleted these four entries. 52 94 73 99

1) Remove 52* 94* 73* 99*



2) Remove 99* : Merge with Sibling



... Cont'd

7 Stockage et Indexes: Hachage Extensible — 15 points

Dans cette question, utilisez les assomptions et notations suivantes:

- $k*$ dénote une entrée de données correspondant à la clé de recherche k.
- h est la fonction de hachage qui retourne les derniers bits de la représentation binaire du code ASCII de k.
- Chaque compartiment contient un maximum de 4 entrées de données.
- L'ensemble des clés de recherche disponibles, ainsi que leur code ASCII, est donné dans la table ci-bas.

binary ASCII code	key	binary ASCII code	key
1000001	A	1001110	N
1000010	B	1001111	O
1000011	C	1010000	P
1000100	D	1010001	Q
1000101	E	1010010	R
1000110	F	1010011	S
1000111	G	1010100	T
1001000	H	1010101	U
1001001	I	1010110	V
1001010	J	1010111	W
1001011	K	1011000	X
1001100	L	1011001	Y
1001101	M	1011010	Z

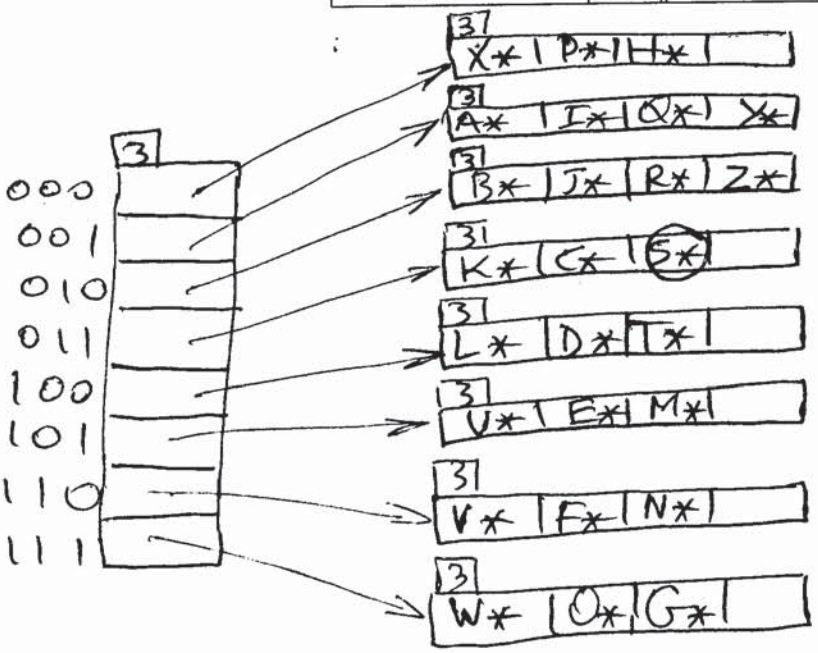


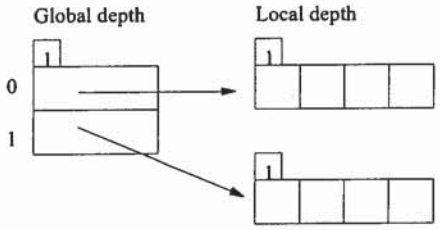
Table after insertion of R*, S*, T*, U*, V*, W*, X*, Y*, Z*.

... suite

Partie A — 5 points – Table de Hachage à Partir du Début

Considerez l'index à hachage initial suivant.

Note: any insertion that causes a structural change in the tree is marked with a circle.



Montrez les étapes de l'algorithme du hachage extensible en insérant toutes les clés obtenues de la table ASCII ci-dessus (Suggestion: suivre la méthode utilisée dans l'exemple du livre, où une nouvelle figure était introduite seulement si vous devez doubler la taille du répertoire).

Table after insertion of A*, B*, C*, D*, E*, F*, G*.

Table after insertion of the data entries above, plus I*.

Table after insertion of the above, plus J*, K*, L*, M*, N*, P*.

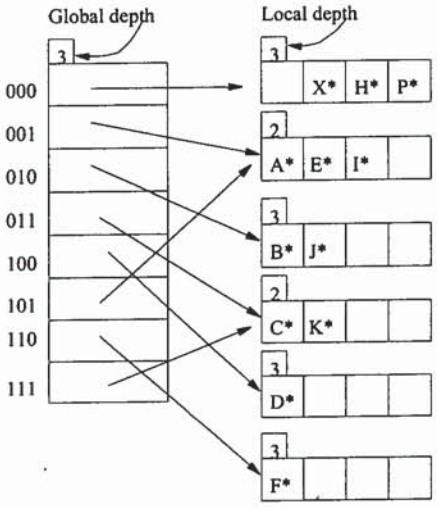
Table after insertion of Q*.

... suite

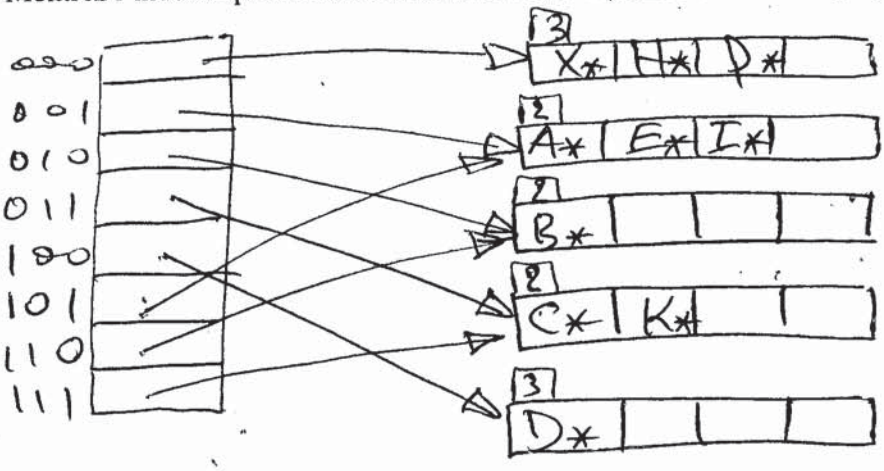
Answer Continued on Previous Page

Partie B — 10 points – Modification d'une Table de Hachage Existante

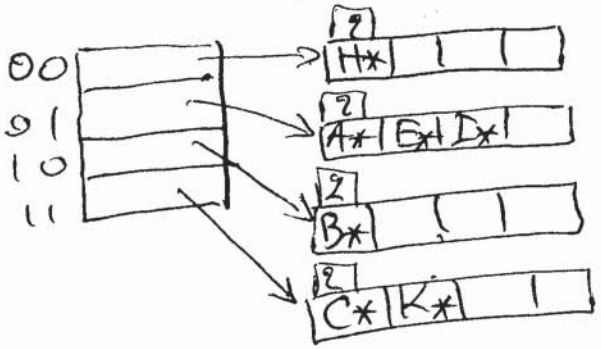
Considerez l'index à hachage extensible ci-bas.



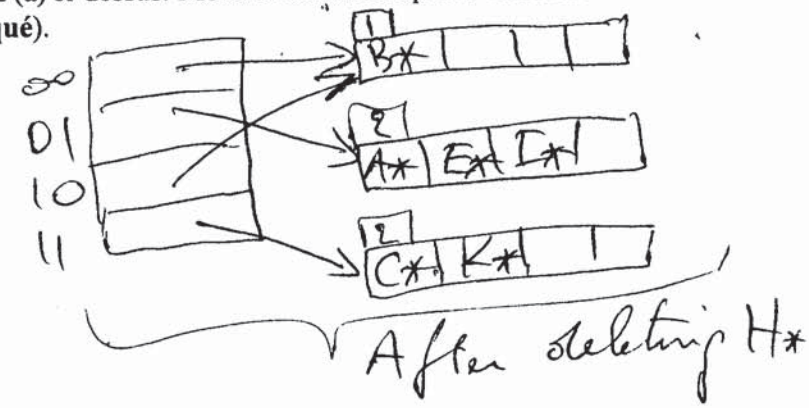
(a) Montrez l'indexe après l'effacement de $J*$ et $F*$ **(dans l'ordre indiqué)**.



(b) Considerez l'index obtenu à la fin de l'étape (a) ci-dessus. Montrez cet index après l'effacement de $X*$, $P*$, $D*$ et $H*$ **(dans l'ordre indiqué)**.



after deleting X*, P*, D*



After deleting H*

... Fin de l'Examen Final