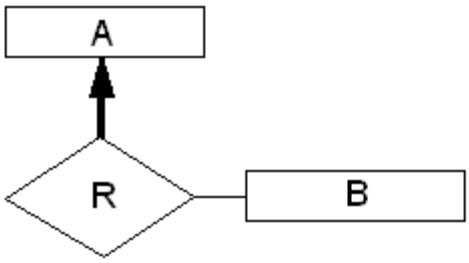
Computer Science 460
Practice Midterm Exam

Part I. Multiple-choice (2 points each)

On the actual exam, you will need to darken the bubble for each multiple-choice answer in a grid on the cover page of the exam. ***Make sure to completely fill in the appropriate bubble, and to erase any stray marks.***



1. Given the portion of an ER diagram shown above (with a thickened arrow from R to A), which of the following statements are true?

   I. R connects each entity in A to at least one entity in B
   II. R connects each entity in A to at most one entity in B
   III. R connects each entity in B to at least one entity in A
   IV. R connects each entity in B to at most one entity in A

   A. only I and II
   B. only II and III
   C. only I and IV
   D. only III and IV
   E. all four statements are true

2. Consider a relation R(a, b, c) having the following tuples:

   (1, 20, 100), (4, 20, 100), (2, 15, 100), and (1, 18, 100).

   Which of the following attributes or attribute combinations could be the primary key of the relation? (Assume that no additional tuples will ever be added.)

   A. attribute a
   B. attribute b
   C. either the combination (a, b) or the combination (a, b, c)
   D. either the combination (b, c) or the combination (a, b, c)
   E. only the combination (a, b)

3. You are given a table named `Alums` that contains the names and personal information of all graduates of the college that you work for. It includes `name` and `age` attributes, and a `state` attribute specifying the state in which a person resides.

   Consider the following SQL queries:

   I.
   ```
   SELECT name, MIN(age)
   FROM Alums
   WHERE state = "CA";
   ```

   II.
   ```
   SELECT name, MIN(age)
   FROM Alums
   WHERE state = "CA"
   GROUP BY name;
   ```

   III.
   ```
   SELECT name, age
   FROM Alums
   WHERE state = "CA"
     AND age = (SELECT MIN(age) FROM Alums);
   ```

   Which of these queries (if any) would successfully find the name and age of the youngest graduate living in California (CA)? You may assume that names are unique.

   A. only I
   B. only II
   C. only III
   D. both II and III, but not I
   E. none of them

4. You have two relations A and B. A has three tuples, and B has two. What is the maximum number of tuples that could appear in the natural join of A and B?

   A. 2
   B. 3
   C. 4
   D. 5
   E. 6

5. The hash table shown at right is being maintained using dynamic linear hashing. The key "cat" was just inserted in the table, and as a result an extra bucket (bucket 4) was added to the table. Which of the keys do we need to consider for possible movement to the new bucket?

000 = 0	dog
001 = 1	frog
010 = 2	lion
011 = 3	cat
100 = 4	

   A. only "dog"
   B. only "lion"
   C. only "cat"
   D. both "dog" and "lion"
   E. all four of the keys

PART II: Answer all three questions in the space provided.

II-1. Relational Queries *(10 points total)*

Consider the following relational schema (keys are underlined):

Product(pid, name, price, manufacturer)
Buys(cid, pid, month, day, year)
Customer(cid, cname, age)

Note that Buys(cid) references Customer(cid), and Buys(pid) references Product(pid).

a. Write **a single relational algebra query** for the following: "Find the names of all customers who have purchased products that are **not** manufactured by Acme." You may assume that "Acme" appears somewhere in the manufacturer field of all products manufactured by Acme. (5 points)

Do EITHER b or c below, and clearly indicate which one you want us to grade. (5 points)

b. Write **a single SQL query** for the following: "For each product, determine how many times it was bought in 2010, if at all." The result should be tuples of the form (product name, num times). If a product was not purchased at all in 2010, the second attribute should have a value of 0.

c. Write **a single SQL query** for the following: "Find the names of all customers who have not purchased any Acme products."

II-2. Record Formats and Tree-Based Index Structures *(12 points total)*

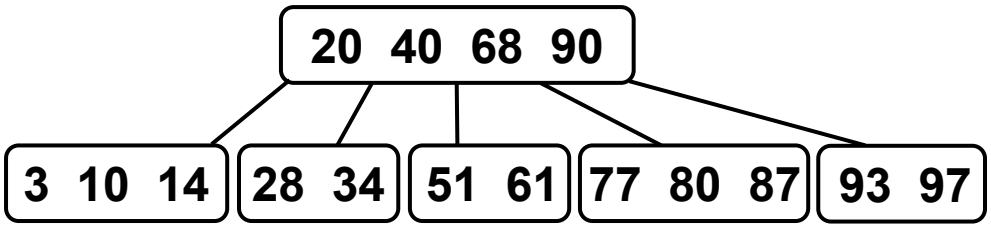
a. Assume that a DBMS is using variable-length records that begin with a header of field offsets to store tuples from a slightly simplified `Movie` relation:

```
Movie(id CHAR(7), name VARCHAR(64), year INTEGER,
  rating VARCHAR(5), runtime INTEGER, earnings_rank INTEGER)
```

If *all* of the values for a given row (including the primary key) are stored in the record, and we're using 1-byte characters, 4-byte integer data values, and 2-byte integer metadata, what would the record look like for the following tuple?

```
("1234567", "The King's Speech", 2010, "R", 118, null)
```

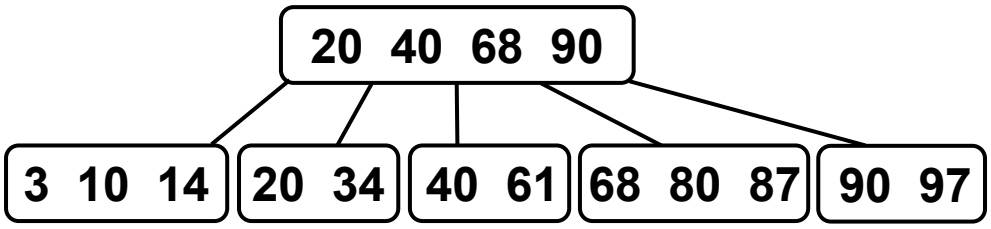
b. Consider the following B-tree of order 2:



What would the tree look like after the following sequence of operations:
insert an item whose key is 65, insert an item whose key is 79,
insert an item whose key is 85.

(continued on the next page)

c. Consider the following B+tree of order 2:



What would the tree look like after the following sequence of operations:
insert an item whose key is 65, insert an item whose key is 79,
insert an item whose key is 85.

II-3. Dynamic Linear Hashing *(8 points total)*

Consider the following hash table, which was constructed using linear hashing with the hash function h(k) = the length of the string k. For example, "True Lies" has a hash code of 9, because it is 9 characters long.

0	"Dinosaur"
1	"True Lies"
2	"Avatar", "Braveheart"
3	"Ray", "Ratatouille", "Michael Clayton"
4	"Philadelphia", "Milk"
5	"Fargo"

a. Where would "Stuart Little" (length 13) and "Batman Returns" (length 14) be inserted in this table, assuming that they can be added without growing the table?
(You may find it helpful to make use of the chart of decimal-binary equivalents below.)

b. After these two insertions, if we then increase the size of the table by one bucket, which keys would need to be rehashed, and which of them (if any) would actually be moved to a different bucket?

rehashed:

moved:

decimal-binary equivalents (for 5-bit non-negative integers):

0 = 00000	8 = 01000	16 = 10000	24 = 11000
1 = 00001	9 = 01001	17 = 10001	25 = 11001
2 = 00010	10 = 01010	18 = 10010	26 = 11010
3 = 00011	11 = 01011	19 = 10011	27 = 11011
4 = 00100	12 = 01100	20 = 10100	28 = 11100
5 = 00101	13 = 01101	21 = 10101	29 = 11101
6 = 00110	14 = 01110	22 = 10110	30 = 11110
7 = 00111	15 = 01111	23 = 10111	31 = 11111