UNIVERSITY OF TORONTO
Faculty of Arts and Science

April/May Examinations 1997

CSC228S

Duration — 3 hours

Aids allowed: One 8.5 by 11 inch aid sheet, as described in class

- **Make sure your examination booklet has 14 pages (including this one).**
- Write your answers in the spaces provided. Do not feel that you must use all of the space provided. If you run out of space on any question, use the back of a page, and draw an arrow to point this out.
- You will be rewarded for concise, well-thought-out answers, rather than long rambling ones.
- Write legibly.
- For questions that involve writing C code, comments are not necessary. If you need to call a standard C function but can't remember the correct order of arguments, just indicate the meaning of each argument.

Family Name: ____________ Given Names: ____________

Student #: ____________ Tutor: ____________

1.	/ 14
2.	/ 12
3.	/ 13
4.	/ 8
5.	/ 10
6.	/ 12
7.	/ 10
8.	/ 4
9.	/ 17
Total	/ 100

Question 1

[14 marks in total]

(a) [6 marks]
Consider the following definition:

```
typedef struct {
     int   ID;
     char  artist[15];
     char  title[20];
     int   year;
     float price;
     int   quantityInStock;
} CDrec;
```

Write a C function that takes an array of these structs, an integer n, and a `FILE *` that points to an already open file, and writes each of the first n structs from the array out to the file in binary. Some marks will be awarded for efficiency.
Assume that $n \geq 1$, and that nothing goes wrong.

(b) [8 marks]
Note: You may find the conversion tables at the bottom of this page helpful in answering this question.

To a UNIX system, text files and binary files look the same; they are both just sequences of bytes. The difference is in how the bytes are used to stand for things like words and numbers. For this question, assume that integers are represented by four bytes in memory. Specify any other assumptions that you need to make.

Think of an integer that takes *fewer* bytes when written in binary (using fwrite) than when written as text (using fprintf).

- Give the integer, in base 10:

- Give the sequence of bytes that would result from fprintf with this integer:

- Give the (shorter) sequence of bytes that would result from fwrite with this integer;

Think of an integer that takes *more* bytes when written in binary (using fwrite) than when written as text (using fprintf).

- Give the integer, in base 10:

- Give the sequence of bytes that would result from fprintf with this integer:

- Give the (longer) sequence of bytes that would result from fwrite with this integer:

character	ASCII value
1	49
4	52

decimal number	binary equivalent
1	1
4	100
14	1110
41	101001
49	110001
52	110100
14111	11011100011111

Question 2

[12 marks in total]

a. [2 marks]
When we say that file input is “buffered”, what does this mean?

b. [2 marks]
Why is file input buffered?

c. [2 marks]
What is the difference between a binary file and a text file?

d. [2 marks]
In a hashing scheme, what is the difference between a collision and overflow?

e. [2 marks]
What is a “pinned” record?

f. [2 marks]
What is “incremental” about incremental hashing schemes? Explain.

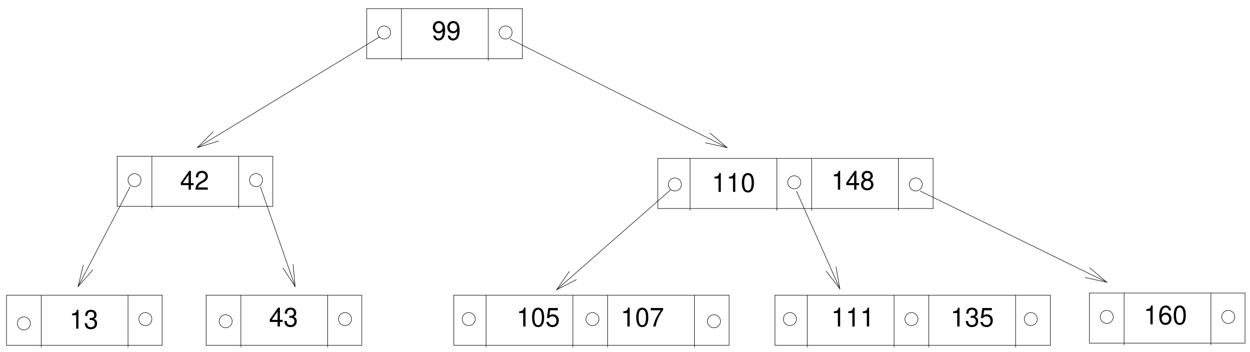
Question 3

[13 marks in total]

a. Consider a B-tree of order 3. What is the minimum number of pointers required in every internal node in such a tree?

What is the maximum?

The following tree is a valid B-tree of order 3:



Below, you will show what the tree would look like if the keys 160, 99, and 43 were deleted in that order.

b. Show what the tree would look like if the key 160 were deleted.

c. Show what the tree would look like if the key 99 were then deleted.

d. Show what the tree would look like if the key 43 were then deleted.

Question 4

[8 marks in total]

These questions concern implementation details for B-trees and hashing.

a. [4 marks]
Write C code to declare a variable called `oneNode` that is capable of holding a B-tree node. Assume that the key type is integer, and that the actual data associated with a key is in a separate data file; for each key, the B-tree simply stores a file position indicator into the data file.

Include any appropriate typedefs or #defines. Your code should work for any branching factor M. Assume that M has already been #defined.

b. [4 marks]
Assume we have a hash function that returns a bucket number. Write a C function called `goTo` that will convert a given bucket number into a file position indicator and then go to that position in a data file whose `File *` has been passed to your function. State any assumptions that you must make.

Question 5

[10 marks in total]

Recall linear hashing, one of the incremental hashing schemes. When the performance of a linear hashing scheme degrades, a bucket is “split”, starting from bucket zero.

a. [1 marks]

Suppose that we are using linear hashing, our initial hash function is $h(k) = k \ mod \ T$ and that we have already split S buckets (i.e., buckets 0 to $S - 1$ inclusive) where $S < T$. If another split is deemed necessary, which is the next bucket to be split?

b. [2 marks]

To which buckets will all of those elements go?

c. [5 marks]

Make a convincing argument that your answer to part (b) is correct.

d. [2 marks]

There is another incremental hashing scheme that is the same as Linear Hashing except in how buckets are chosen for splitting. This new scheme splits next whatever bucket was last added to. This way, we are more likely to split an overfilled bucket.

One problem with this scheme is that it’s harder to keep track of which buckets have been split. Assuming we can do this, and that we are about to split some bucket p, what other problem does this new scheme introduce?

Question 6

[12 marks in total]

Recall extendible hashing, another incremental hashing scheme. With extendible hashing, we keep a directory to the buckets. It changes in size as necessary to accommodate more or fewer buckets.

Assume that the hash function simply returns the first 8 bits of the key represented in binary; that to begin with, the first 2 of those bits are used for the index; that bucket size is 2 records; and that we begin with a single bucket.

a. Draw a diagram showing the state of affairs before any insertions. have been done. Show the directory and any buckets.

For the rest of this question, you will show the state of affairs after a sequence of insertions. The following table will be helpful:

key	h(key), i.e., value returned by hash function
108	01101100
194	11000010
1	00000001
85	01010101
127	01111111

b. Show the state of affairs if the keys 108 and 194 have been inserted, in that order.

c. Show the state of affairs if the key 1 is then inserted.

d. Show the state of affairs if the key 85 is then inserted.

e. Show the state of affairs if the key 127 is then inserted.

Question 7

[10 marks in total]

Below is a series of bad ideas to do with hashing. In some cases, it may be impossible to implement the idea, and in others it may be possible but inefficient. For each one, explain what is wrong with the idea and why.

a. A simple (non-incremental) hashing scheme in which the buckets have variable size.

b. A hashing scheme in which the buckets are stored as a linked list within the file.

c. A hashing scheme in which we choose a bucket size large enough that we won't have to handle bucket overflow.

d. A hashed file of medical records (hashed by OHIP number) that is also accessible according social insurance number by using a second hash function that operates on the social insurance number.

e. A hashed file for current student registration records, whose hash function is based on the first 4 digits of UofT student numbers.

Question 8

[4 marks in total]

We want to store information about movie rentals. One way to do this is to use a single file, with fields for: rental date; name, phone number, and credit card number of the customer who rented; and title, ID number and category of the movie they rented. Below are some sample records.

Data	Name	PhoneNumber	CreditCardNumber	Title	IDNumber	Category
1995	Brennan Schoeler	604-1234	2934 0123 2344 4205	Aladdin	1414141	Children's
1996	Brennan Schoeler	604-1234	2934 0123 2344 4205	Snow White	9877117	Children's
1991	Kathy Knill	769-9823	9234 8575 9672 3423	Star Wars	8686868	SciFi

... etc.

We have talked about three kinds of update anomaly. These occur when an update to a file introduces some problem with the meaning of the file. Give specific examples of two different kinds of update anomaly that could occur if we were to use this file format.

Question 9

[17 marks in total]

A video rental store has a relational database to keep track of their customers, videotapes, etc. Their database includes the files below. (Notice that they have organized things better than in question 8.) The RENTALS file keeps a history of all past rentals, up to the current date.

file VIDEOS:

IDNumber	Title	Category
8818398	Sleepless in Seattle	Romance
1234562	Rambo	Action
2222222	Friday the 13th	Horror

... *etc.*

file CUSTOMERS:

MembershipNumber	Name	PhoneNumber	CreditCardNumber
12345	Diane Horton	555-1212	5191 8244 4568 9999
15111	Jane Davies	787-3724	5191 1111 2222 3333
99999	Lydia Veen	788-0924	5224 9813 3543 0153

... *etc.*

file RENTALS:

IDNumber	MembershipNumber	Year
8818398	99999	1996
8818398	12345	1995
2222222	15111	1995

... *etc.*

For this question, you will write a series of queries on this database. You may use either relational algebra or SQL.

a. Write a query to extract all Action Movies.

b. Write a query to extract the titles of all Action movies.

c. Write a query to extract all customers who rented in 1996.

d. Write a query to extract all customers who rented in 1996 and in 1985.

e. Write a query to extract the titles of all Horror movies that were rented by Diane Horton.

END OF EXAM; Total pages = (14)