University of Toronto

csc228S - File Structures and Data Management, Fall 2000

Midterm Test

Monday, October 30, 2000

Duration: 50 minutes

Aids allowed: None

Family Name: ____________________ Given names: ____________________

Student #: ____________________ Tutor: ____________________

- There are 10 pages, including this one. The test is out of 100 marks and the value of each question is provided; please use this information to manage your time effectively.
- Answer all questions directly on the exam in the space provided. If you need more space, use the back of the exam sheet and indicate clearly where your answer is continued.

Part A:	______	/ 15
Part B:	______	/ 12
Part C:	______	/ 20
Part D:	______	/ 21
Part E:	______	/ 16
Total	______	/ 84

Part A [15 marks in total]

Fox Mulder and Dana Scully (characters from the X-files show, in case you don't watch it) come to you for help. They're receiving some kind of strange alien transmission, and in order to analyze it, they need to first store it on their hard disk. Unfortunately, they have only a small PC at their disposal, and will need to compress the file to make it fit. The transmission is in the form of an image whose size is 10 x 10 pixels and whose values range from 0 through 254:

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	254	254	254	128	128	128	64	0	64
0	254	0	0	128	0	128	64	0	64
0	254	254	0	128	0	128	0	64	0
0	254	0	0	128	0	128	0	64	0
0	254	0	0	128	0	128	64	0	64
0	254	0	0	128	128	128	64	0	64
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

1. (3 points) The above image requires one byte per pixel, yielding an uncompressed size of 100 bytes. Can you think of a more compact notation that, for the above particular image in particular, would yield a compressed size of 25 bytes?

   ```
   ANSWER

   2 bit code per pixel: 00=0 01=64 10=128 11=254
   ```

2. (4 points) We discussed a run-length encoding scheme in class which copies the input file (image) to the output file (compressed image) except where the same pixel value occurs more than once in succession. When two or more values occur in succession, 3 bytes are entered: a run-length code indicator, the pixel value, and the pixel count. Choose an appropriate run-length code indicator, justify your choice, and show the contents of the resulting run-length encoded compressed image file that describes the first four rows (40 pixels) of the image.

   ```
   ANSWER

   Choose 255 as the 1 byte run delimiter, since 255 doesn't
   occur in the image:

   255 0 21 255 254 3 255 128 3 64 0 64 0 254 255 0 2 128 0 128 64 0 64
   ```

3. (2 points) Describe the general principal of variable-length coding.

   ```
   ANSWER

   Variable-length encoding schemes assign shorter codes to the most
   frequently occurring strings (in this case, pixels) in the file.
   ```

4. (2 points) With a variable-length coding scheme, you save space at the expense of giving up a particular kind of access to the image data. What kind of access are you giving up, and what file structure could you introduce that would help alleviate the problem?

```
ANSWER

You lose direct access to the image data. With the construction of an
index, i.e., a second file of fixed-length fields that provide byte
offsets for the variable length records (image rows) in the compressed
image file, you would provide direct access to image rows but not
to pixels within a row.
```

5. (4 points) The next image that Mulder and Scully receive from the aliens is very large, with dimensions 10,000 x 10,000. If we assume that each of the 10,000 rows is a fixed-length record, that the number of bytes per sector on the disk is 200, that the number of sectors per track is 25, and that the number of tracks per cylinder is 32, how many cylinders are required to hold the uncompressed file, assuming 1 byte per pixel?

```
ANSWER

10,000 records at 10,000 bytes/record
10,000 bytes/record / 200 bytes/sector = 50 sectors/record
10,000 records * 50 sectors/record = 500,000 sectors
500,000 sectors / 25 sectors/track = 20,000 tracks
20,000 tracks / 32 tracks/cylinder = 625 cylinders
```

Part B [12 marks in total]

1. (6 points) In Assignments 1 and 2, you processed an input file whose fields were fixed in length. Name three other methods for maintaining a field.

```
ANSWER

-length indicator
-field delimiter
-keyword-value expression
```

2. (6 points) For each method you listed in the previous question, provide (very briefly) one advantage and one disadvantage.

```
ANSWER

length indicator
advantages:
-efficient use of space through variable length fields
-can jump directly to the next field without scanning
disadvantages:
-overhead of byte count
-byte count must be fixed in length, resulting in internal fragmentation for
small byte counts (small fields).

field delimiter
advantages:
-efficient use of space through variable length fields
-no byte count overhead
disadvantages:
-overhead of delimiter
-must scan to get to next field

keyword-value expressions
advantages:
-fields can be in any order
-fields can be missing
-file more readable (self-describing)
-efficient use of space through variable length fields
disadvantages:
-overhead of keyword
-overhead of keyword parsing mechanism
```

Part C [20 marks in total]

1. (10 points) List 5 methods for organizing records in a file.

   ```
   ANSWER

   - a record is fixed-length
   - a record has a fixed number of fields
   - a record has a length indicator
   - a record has a delimiter
   - a second file containing a fixed-length byte address for each record
   ```

2. For each of the scenarios listed below, indicate which method for organizing records (from your list in the previous question) is best. Note that you only need to list the method, not justify it. Also, note that there may be more than one correct answer, and you only need to list one of them.

   a) (2 points) Your application must read the entire contents of each record once in an address file. Space efficiency is of the utmost importance, and fields (which are never empty) are variable-length.

   ```
   ANSWER

   Since each record must be read, neither direct access nor record
   skipping is necessary. That leaves a record delimiter or field
   count. Since space is at a premium, the record delimiter is
   superfluous, and we can exploit the fixed number of fields, all of
   whom must be populated.
   ```

   b) (2 points) The same scenario above, except that fields are variable length and some fields may be missing.

   ```
   ANSWER

   A record delimiter supports both variable length fields and missing
   fields without incurring the cost of a record byte count.
   ```

   c) (2 points) For the image shown in Part A, you want to be able to quickly read a specified record (row) from the uncompressed image.

   ```
   ANSWER

   fixed length records provides direct access.
   ```

   d) (2 points) You have variable-length fields in your record and want to be able to quickly jump to the next record if the first field contains a particular code.

   ```
   ANSWER

   you should have a record length indicator at the beginning of each record.
   ```

   e) (2 points) For the image shown in Part A, you want to be able to quickly read a record (row) from the image when it is compressed using a variable-length code.

ANSWER

should use second file containing byte offset of records, allowing direct access.

Part D [21 marks in total]

We saw in class an effecive method for retrieving reusable space from deleted fixed-length records. This method, which embeds a common data structure directly in the file, allows us to not only determine immediately if there are empty records in the file, but allows us to jump immediately to an empty record if one exists.

1. (5 points) Describe the method.

   ANSWER

   Embed an available record stack directly in the file itself. It will require the addition of a head of stack pointer which can be placed in the header of the file. When the first record is deleted, the head pointer is set to the RRN of the deleted record, and a ''*'' (or other suitable flag) is placed in the first column of the deleted record. The deleted record must also contain the RRN of the next deleted record in the stack (or -1 if it's the bottom of the stack).

2. Consider the following file with 10 records:

```
record #     1     2     3     4     5     6     7     8     9     10
contents:    A...  B...  C...  D...  E...  F...  G...  H...  I...  J
```

   Using the method described in the previous question, show the contents of the file after each of the following operations is performed. Note that the operations are cumulative, e.g., apply the third operation to the contents resulting from the application of the first and second operations. If you have added any additional data structures to implement the space reclamation method, then be sure to show their contents.

   a) (2 points) delete record 3

   ANSWER

```
head pointer: 3
record #     1     2     3     4     5     6     7     8     9     10
contents:    A...  B...  *-1.  D...  E...  F...  G...  H...  I...  J
```

   b) (2 points) delete record 8

   ANSWER

```
head pointer: 8
record #     1     2     3     4     5     6     7     8     9     10
contents:    A...  B...  *-1.  D...  E...  F...  G...  *3..  I...  J
```

   c) (2 points) delete record 2

   ANSWER

```
head pointer: 2
record #     1     2     3     4     5     6     7     8     9     10
contents:    A...  *8..  *-1.  D...  E...  F...  G...  *3..  I...  J
```

d) (2 points) add record K

```
ANSWER

head pointer: 8
record #     1    2    3    4    5    6    7    8    9    10
contents:    A... K... *-1. D... E... F... G... *3.. I... J
```

e) (2 points) add record L

```
ANSWER

head pointer: 3
record #     1    2    3    4    5    6    7    8    9    10
contents:    A... K... *-1. D... E... F... G... L... I... J
```

3. (2 points) What simple change to the above method do you need to make to handle the deletion of variable-length records?

```
ANSWER

replace the RRN's with byte offsets
```

4. (2 points) What additional information do you need to store in a record to handle the addition of variable-length records?

```
ANSWER

you need to insert a record length indicator, so that you can
determine if the deleted record is large enough to accommodate the new
record.
```

5. (2 points) What placement strategy minimizes search through an available list of deleted records?

```
ANSWER

a worst-fit placement stragegy sorts the deleted records in decreasing
order of size. Thus, if the first record is large enough you take
it. If it's not large enough, you stop. You never search beyond the
first available record in the list.
```

Part E [16 marks in total]

You are asked to design an indexing system for a file of variable-length records that contains the following fields:

```
athlete
nation
event
final rank in event
```

and that:

a) allows access to athletes by nation.

b) allows direct access to any of the top 10 finishers for a specified event, e.g., "get me the athlete who finished 7th in the 100m."

c) allows access to an athlete by name (you may assume that no two athletes have the same name).

d) does not rearrange the records in the file.

1. (12 points) Briefly describe a primary and secondary key indexing system that supports the above three types of indexing and is built on top of a primary key indexing system (4 points each).

   ```
   ANSWER

   First build a primary key index whose entries are sorted by primary
   key. The primary keys must be unqiue and could be defined any number
   of ways.

   Build a secondary key index, sorted by nation, that maps each nation
   to a linked list of primary key indices (of athletes belonging to that
   nation). A linked list is essential since different nations have
   vastly different numbers of athletes.

   Build a secondary key index, sorted by event, that maps each event to
   an array of 10 primary key indices corresponding to the top 10
   athletes finishing the event and sorted in increasing rank. Since the
   number of primary key indices for a given secondary key index is
   fixed, and since direct access is required, an array is ideally
   appropriate in which to store the primary key indices.

   Build a secondary key index sorted by athlete name, that contains a
   single primary key, i.e., the primary key for that athlete.
   ```

2. (4 points) Why not replace the primary keys in a secondary key indexing system with the byte offsets of the individual records?

   ```
   ANSWER
   ```

deleting or updating a record localizes any changes to the primary key index, thereby insulating all secondary key indices from such changes. Deleted records (i.e., primary keys) pointed to by secondary keys are caught by the primary key indexing system. Thus, the secondary key indexes (there may be multiple indexes) do not need to be re-sorted each time a change is made.