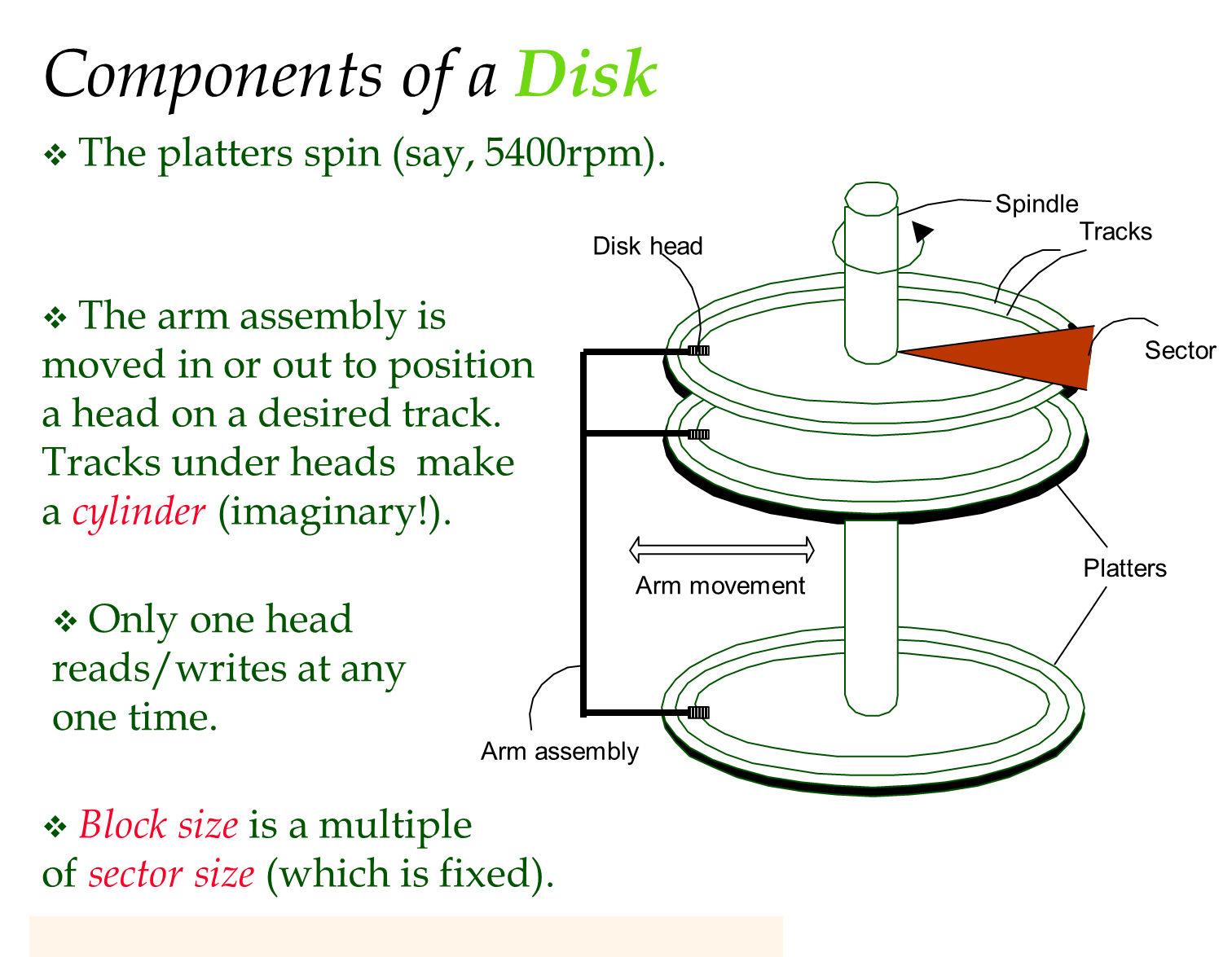
**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ PS ID # \_\_\_ COSC 3380**

**Exam 3**

**1. (10 pts.)**



Consider a disk with a sector size of **256 bytes**, **3000 tracks per surface**,

**50 sectors per track**, **five double-sided platters**, and **average seek time of 10 msec**.

1. (2 pts.) What is the **capacity of each surface**?

**ANSWER:**

1. (2 pts.) What is the **capacity of the disk**?

**ANSWER:**

1. (2 pts.) How many **cylinders does the disk** have?

**ANSWER:**

1. (2 pts.) If the disk platters rotate at 5400 rpm (revolutions per minute), what is the **maximum rotational delay**?

**ANSWER:**

1. (2 pts.) If one track of data can be transferred per revolution, what is the **transfer rate**?

**ANSWER:**

**2. (15 pts.)**

Consider the **B+ Tree** **Index** of order **d** = 2 shown below.



a) (10 pts.)

Show the **B+ Tree** **Index** that would result from **inserting** a **Data Entry** **3\*** ***into the original B+ Tree* Index**.

(**Please redraw the complete resulting B+ Tree** **Index!**)

**ANSWER:**

b) (5 pts.)

**How many** **Page writes** and **Page reads** will occur?

**ANSWER:**

**3. (10 pts.)**

Consider the **Extendible Hashing** **Index** shown below. Answer the following questions about this **Extendible Hashing** **Index**:



\* \* \* \*

\* \* \*

\*

\* \* \*

\* \*

Show the **Extendible Hashing** **Index** after **inserting** a **Data Entry** with hash value **68\***.

(**Please redraw the complete resulting Extendible Hashing** **Index!**)

**ANSWER:**

**4. (15 pts.)**

Applying hash function **h** on a **key** **v** we obtain the following hashes:





Figure below shows a **Linear Hashing** **Index**.



\*

\*

\*

\*

\*

\*

\*

a) (5 pts.)

Show the **Linear Hashing** **Index** after **inserting** *sol*, where **h**(sol) = 0001010001.

**ANSWER:**

b) (10 pts.)

Show the **Linear Hashing** **Index** after **inserting** *judy*, where **h**(judy) = 1110000110.

(In the **Linear Hashing** **Index** from **a.**)

**ANSWER:**

**5. (20 pts) Worst Case Cost ANALYSIS**

Consider a **delete*****Data Record*** *specified using an* ***equality*** *condition*. For each of the

**Heap File**, **Sorted File**, **B+ Tree Index, a**nd **Hash Index File** **organizations**, what is the **cost** if ***no Data Record qualifies*** and the ***search key*** *is not a candidate key*?

Assume **B** **Data Records Pages**, **D** cost per **Page**, **R** **Data Records** per **Page**.

Assume **N** **Pages** of **Data Entries K\*** and **F** Fan Out for the **B+ Tree Index**.

Assume **N** **Pages** of **Data Entries K\*** and **H** cost for **Hash** (assume no overflow **Pages**) for the **Hash Index**.

|  |  |
| --- | --- |
| **File** organization | **Delete** **cost** |
| **Heap File** | ? |
| **Sorted File** | ? |
| **Unclustered B+ Tree Index** | ? |
| **Unclustered Hash Index** | ? |

**You must explain why for full credit**. Without explaining you will **ONLY GET half**.

**ANSWER:**

**6. (10 pts.)**

Consider a **Relation/Table** with this schema:

**Employees** *(****eid****:integer*, **ename**: string, **sal**:integer, **title**:string, **age**:integer)

Suppose that the following **Indexes**, all using **Alternative (2)** for **Data Entries k\***, exist:

* a **Hash** **Index** on eid,
* a **B+ Tree** **Index** on sal,
* a **Hash Index** on age, and
* a **Clustered** **B+ Tree** **Index** on <age, sal>.

Each **Employees Data Record** is **100 bytes long**, and you can assume that each **Index** **Data Entry k\*** is **20 bytes long**. The **Employees** **Relation** contains **10,000 Pages**.

For each of the following selection conditions, describe the best **evaluation method** **and the cost**:

a) (5 pts.)

**sal** **>** 200 **&** **age** **=** 20

**You must explain why for full credit**. Without explaining you will **ONLY GET half**.

**ANSWER:**

b) (5 pts.)

**sal >** 200 **&** **title** **=** ‘CFO’

**You must explain why for full credit**. Without explaining you will **ONLY GET half**.

**ANSWER:**

**7. (10 pts.)**

Consider a **Relation**/**Table** **R**(**A**, **B**), with the following characteristics:

* total number of **Data Records/tuples**: **1,000,000**
* 10 **Data Records/tuples** per **Page**
* Attribute **A** is a candidate key; range is 1 to 1,000,000
* **Clustered** **B+ Tree** **Index** of **depth 4** on A

Estimate the **number of page transfers** needed to evaluate **σA** < 4,000 **query** by using

a) (5 pts.)

**Sequential scan**

**You must explain why for full credit**. Without explaining you will **ONLY GET half**.

**ANSWER:**

b) (5 pts.)

**Clustered** **B+ Tree** **Index** of **depth 4** on A

**You must explain why for full credit**. Without explaining you will **ONLY GET half**.

**ANSWER:**

**8. (10 pts.)**

Suppose we have a **Relation/Table of 5,000 Data Records/tuples**.

Each **Page can hold 10 Data Records/tuples**, or **20 key-pointer pairs** (**Data**

**Entries k\***).

An **Index** is built on the **key** field of the **Relation** (thus **no duplicate search keys**) and the **File of Data Records** is **sorted** according to the **key**.

Answer the questions below about the space required for **each Index** (just the **Index**, not **the Data Records** as well).

a) (5 pts.)

How many **Pages** do we need for a **dense Index** of this **Relation**?

**You must explain why for full credit**. Without explaining you will **ONLY GET half**.

**ANSWER:**

b) (5 pts.)

How many **Pages** do we need for a **sparse Index** of this **Relation**?

**You must explain why for full credit**. Without explaining you will **ONLY GET half**.

**ANSWER:**