Name: .....

ID Number: ..... **Exam Number:** .....

Grade: 1: ... 2: ... 3: ... 4: ... 5: ... 6: ... 7: ... Total: .....

Solve **ALL the problems** in the space provided

Read the Problems CAREFULLY!

DURATION OF THE EXAM IS 90 (NINETY) minutes

There are 7 (SEVEN) pages this page included

You may tear off carefully the last page; Do not turn it in then

Read and Sign the statement below at the end of the exam

Unsigned exams will be marked with grade 0 (zero).

STATEMENT

On my honor, I pledge that I have not violated the provision of the NJIT Student Honor Code.

Sign below at the end of the exam

Signature .....

Problem 1. (75 POINTS)
Give short answers to the following questions. (References to Google are to the Google System Architecture of 1997/1998.) If a question has multiple answers write all that apply.

(1) Which one(s) of `OR`, `or`, `Or` are operators in Google?

(2) Which one(s) of `AND`, `and`, `And` are operators in Bing?

(3) Which one(s) of `NOT`, `not`, `-` are operators in Google?

(4) Which one(s) of `NOT`, `not`, `-` are operators in Bing?

(5) What is the input for the `PageRank` computations?

(6) Where does Google store the forward index?

(7) Where does Google store the (inverted) index?

(8) Does `Anchors` store information in the form of `URLs` or `docID`s?

(9) Does `Links` store information in the form of `URLs` or `docID`s?

(9) Where are web-pages stored in Google?

(10) Name/List exactly 5 data structures used by Google.

(11) Who else other than the `Indexer` generates the forward index in Google?

(12) Who writes into the short barrels in Google?

(13) Why does the `Searcher` access `Pagerank`?

(14) What is the input to the `Sorter` and what is its output?

(15) Where is the location of the input to the `Sorter` and where is the `Sorter`'s output located?

Problem 2. (25 POINTS)
Give short answers to the following questions. (References related to Google are to the Google System Architecture of 1997/1998.)

(1) Bot, robot, spider, all refer to the same thing. Give another name for it.

(2) What do we call a frequently occurring word in a text that is not a keyword?

(3) What is the difference between a keyword and an index-term. Explain and provide an example

(4) Name an operation that can map two different `keywords` into the same `index term`.

(5) What is the difference between a `word` and a `keyword` in the context of information retrieval? Give an example that highlights the difference.

Problem 3. (40 POINTS)

The following query Q1 in Google and Bing produced the following results (in millions). Assuming that both search engines support a Boolean query language, what is your expectation for the results of query Q2 and of query Q3. Explain and justify your answer.

	Query	Google Results	Bing Results
Q1.	(a AND b) OR (-a OR -b)	25,270	31,000
Q2.	(a OR b) OR (-a AND -b)		
Q3.	(a OR b) AND (-a AND -b)		

Problem 4. (60 POINTS)
For this problem 1/3= 35%, 2/3 = 65%, 4/7 = 55% , 4/9 = 45% .
For a query involving term t_1, the corpus includes the twenty documents $d_1, \ldots, d_{20}$. Five of them, $d_1, d_2, d_3, d_4, d_{10}$ contain index term t_1 and are relevant, whereas the remaining ones do not contain the index term and are irrelevant. As a response to the query involving t_1 a new Search Engine called **Boongle** returned the following 10 documents.

$$\mathbf{d_1}, \mathbf{d_2}, \mathbf{d_3}, \mathbf{d_4}, d_5, d_6, d_7, d_8, \mathbf{d_{10}}, d_{11}$$

in that specific order (left to right).
(a) What is the precision for this query of the search query? Explain.
(b) What is the recall rate for this query of the search query? Explain.
(c) What is the three point effectiveness using 40%, 60%, 80% recall levels? Explain. This means build a table as follows.

- Its first column is the documents returned (relevant or not).
- Second column contains a `y` for a document that is relevant or `-` for a document that is not relevant.
- Third and Fourth columns indicate recall and precision respectively. Observe the roundings specified.
- Then use a shorter table and record precision at recall levels $0, 20, 40, 60, 80, 100$ percent and interpolated precision.
- Compute 3-point effectiveness.

Problem 5. (30 POINTS)
Give unary and binary codes for 13.

Problem 6. (40 POINTS)
We typed the following queries in Google and Bing and recorded the following number of hits (in millions) for the two search engines. Answer the following questions with respect to the Table below.

No	Query Terms	GOOGLE	BING
Q1	⟨to⟩	5760	13600
Q2	⟨be⟩	1770	2780
Q3	⟨or⟩	1690	3810
Q4	⟨not⟩	1580	9210
Q5	⟨to be⟩	24900	1620
Q6	⟨to be or not⟩	13200	9220
Q7	⟨or not to be⟩	13200	9220
Q8	⟨to be to be or not⟩	13200	9220
Q9	⟨to be or not or not⟩	13200	9220

Table 1: Table for Problem 6

(a) For Q1, Q2, Q5 and ignoring other queries, is there any inconsistency in Google? Explain.

(b) For Q1-Q4, Q6 and ignoring other queries, is there any inconsistency in Bing ? Explain.

(c) Can you explain the results for Q6, Q7 only, for Google? Explain.

(d) Can you explain the results for Q6, Q7, Q8, Q9 only, for Bing ? Explain.

Problem 7. (40 POINTS)
Assume 1GB= 10^9B, and 1MB = 10^6B. Round to the nearest million that won't exhaust your available memory; you don't need a calculator.
You have 1100MB of main memory of which 100MB are OS and programs such as those manipulating tables A and T below.

You are asked to organize the additional space to support a hash table where words are stored in a contiguous table which is an array A of characters delimited by a null character `\0`. A hash table T will store a `wordID` along with a reference (pointer or index) p to A. That way a `wordID` will be associated with a specific `word`.

The average length of a `word` is given as 11 UNICODE characters (1 UNICODE is 2B). A `wordID` and p can only be in multiples of 2B (and thus 23bits or 3B won't be an option) for efficiency. Organize T and A for maximum efficiency.

In an efficient implementation

- (a) How many words n can the scheme support,
- (b) How big would the hash table size m (number of entries) be,
- (c) How many bits or bytes for a `wordID`,
- (d) How many bits or bytes for pointer p,
- (e) How much space (bytes) will your scheme use for T,
- (f) How much space (bytes) will your scheme use for A,
- (g) What is the total space $A + T$ used by your scheme?

Justify your answers and choices. Round to nearest million for n, m, T, A but make sure your rounding doesn't exceed the amount of available memory (i.e. $A + T$ should be accommodated easily by the available memory). Make sure that you fill the following table with information. (**No calculator needed if you are careful.**)

```
   n        =
   m        =
   wordID   =
   p        =
   T        =
   A        =
 A+T        =
```

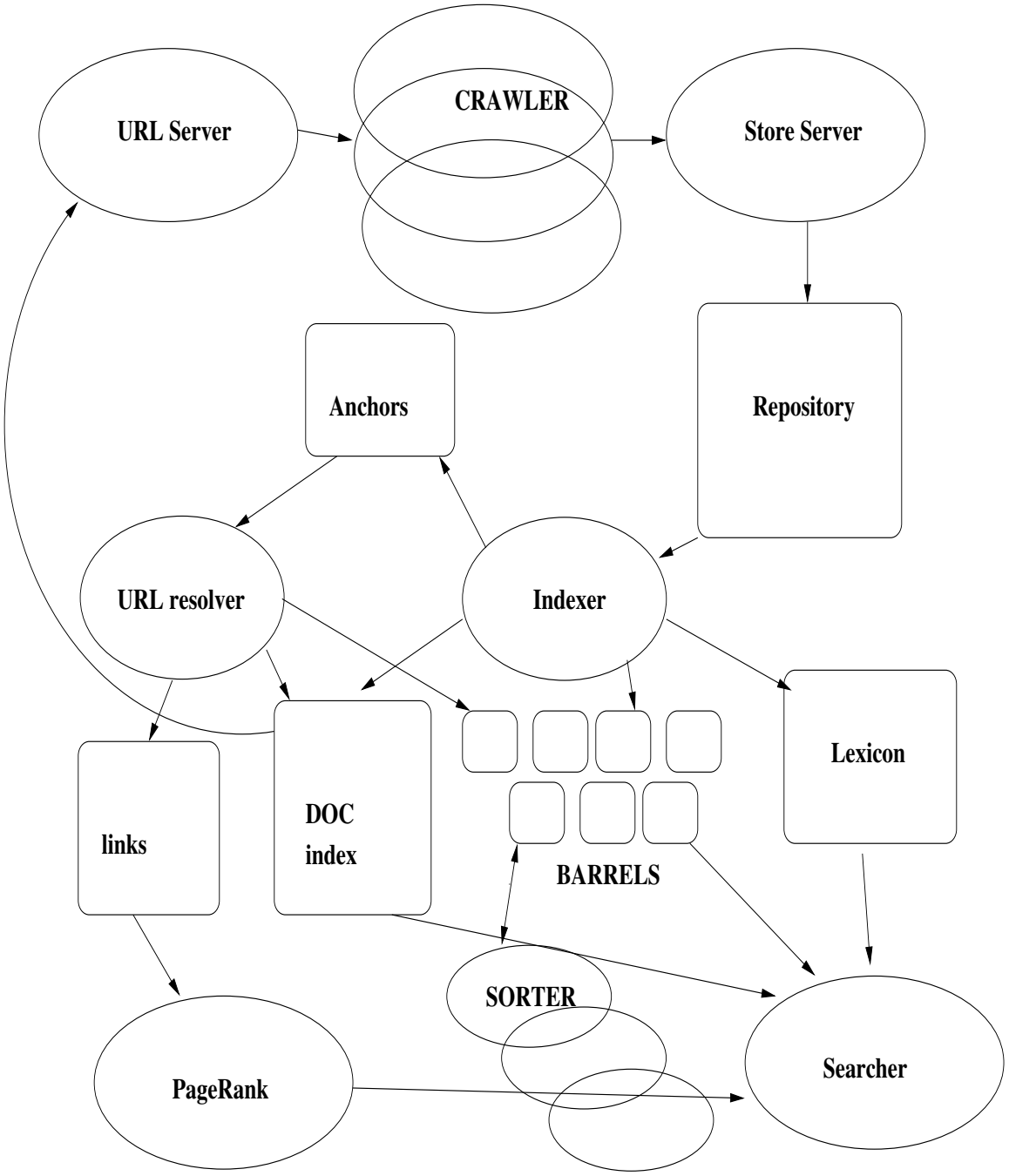

Figure 1: Google System Architecture.