Midterm exam

Due: Thursday, October 19, 2023 6:30 am (Pacific Daylight Time)

Assignment description

Hi there, welcome to the midterm exam!

There are 10 questions below, that add up to 30 points. You only need to answer 25 points worth of them - but if you want you can answer more - ALL the points you earn (including fracti ones) will be counted (no cap), ie included in your cumulative score for the course! How cool, lol.

Fun fact: each question has '**search**' in it :)

The exam is 'closed', except for your cheatsheet. Other than that, please don't "cheattu"!!

Exam duration: 1 hour; OSAS-accommodated students get proportionately more time.

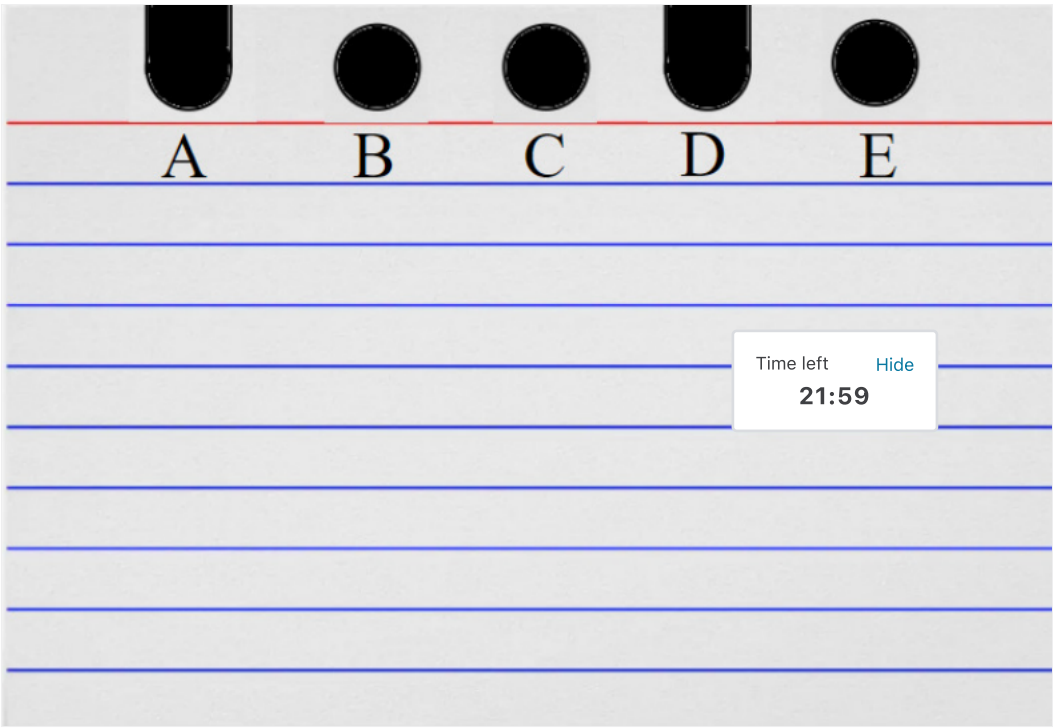
Hope you do well!

Cheers,
Teaching Team

Submit your assignment

Q1 (4 points)

Being the **search** 'geek' that you are, you decide to 'index' your collection of 1000 books, using 1000 index cards and punching holes in them and cutting out some holes, like so:



Time left Hide
21:59

In the above, which shows the index card for

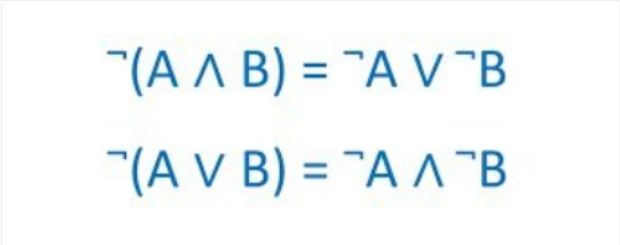
of your 1000 books, your book belongs to categories B,C,E [not punched out], and not to A,D [punched out]. Simple enough. You have 999 more such cards, where each card has 0,1,2,3 holes punched out [but not all 5, duh!!].

You are given a single long knitting needle (or chopstick!) that can pass through holes in the entire stack.

a (2 points). How would you do 'A and B'? Remember, you have just one rod!

b (2 points). How would you do 'A or B'? Again, you have to use the single rod you have.

Big hint: De Morgan's Laws, shown below! Another hint on top of the hint - you can move each negation on the left, to the right! The first line says: NOT(A AND B) === (NOT A) OR (NOT E



Edit Pre

Please enter your response to Q1

Attach files Formatting tips

Q2 (3 points)

We place data in databases (eg. Oracle, MongoDB, Redis, neo4j...), and query the data. When we **search** (a site, or Google/Bing/...), we query, as well.

How are the above (DBs vs search) similar, how are they different? You need to be specific, in both parts of the answer.

Edit Pre

Please enter your response to Q2

Attach files Formatting tips

Q3 (3 points)

To make **search** engines be efficient, we need to avoid indexing. duplicate content. In this contex

Time left Hide
21:59

pare with MD5 (or SHA etc)? How are they similar, how are the different? Explain in a few lines.

Edit Pre

Please enter your response to Q3

Attach files Formatting tips

Q4 (2 points)

A 'rogue' **search** engine crawler would simply disregard robots.txt [which specifies which parts of a site are off-limits]. Your webserver would end up serving such crawlers, possibly valu resources that your site is hosting (that you did not mean to serve].

How would you fight back, ie. 'punish' such a crawler? Be creative in your thinking! Hint: think 'SEO optimization' tricks that websites used to play, in reverse :) In other words: by examinir URL that's requested, you can tell if it should be served, or not - if it shouldn't be, what can you do/serve?

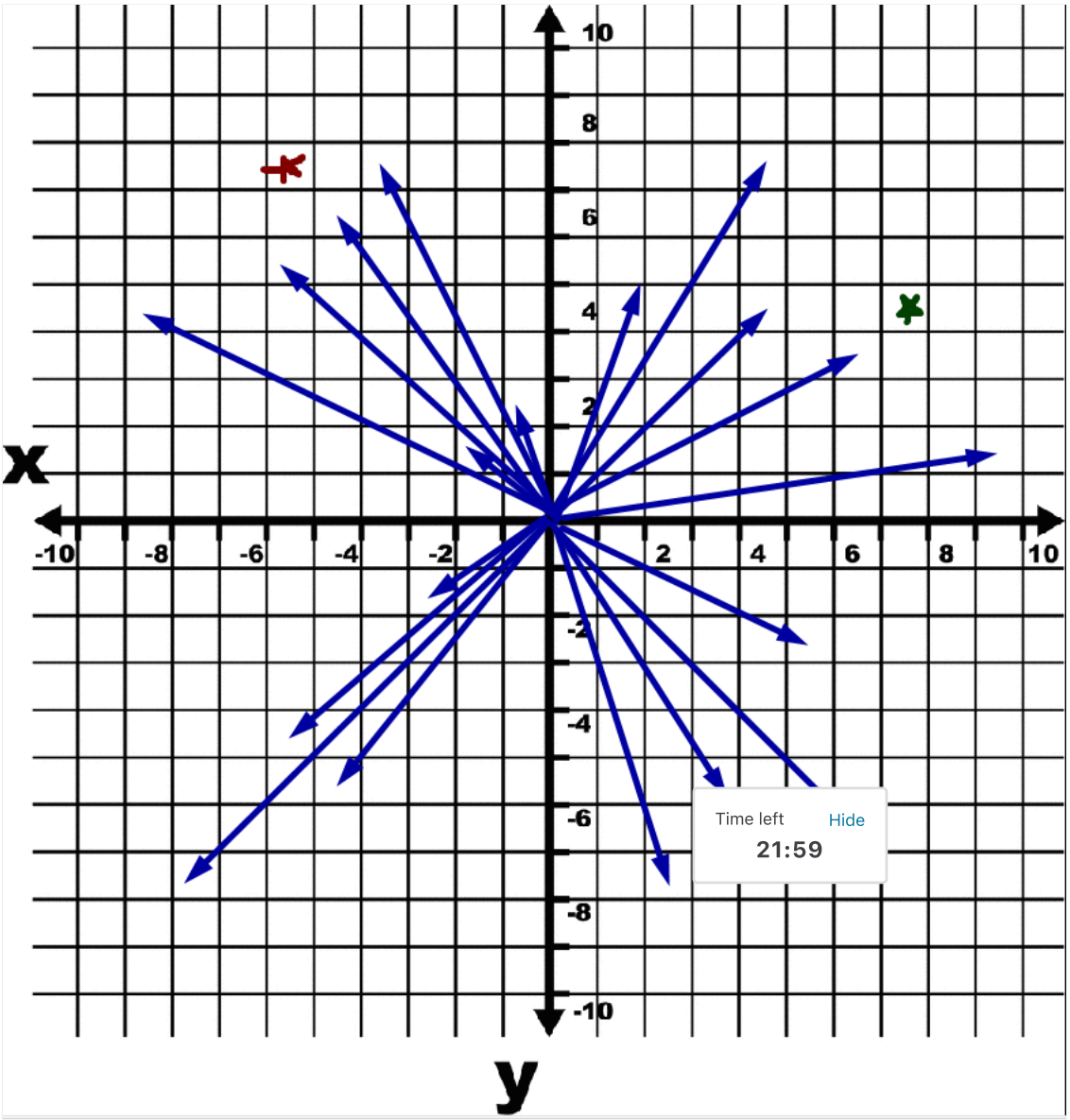
Edit Pre

Please enter your response to Q4

Attach files Formatting tips

Q5 (4 points)

In 'similarity **search**', we look for closest points (or vectors) near a query point (or vector), eg. as shown below [the green and red *s would be the queries, and the blue arrows representi
existing points (or vectors):



It would be extremely inefficient to find nearest neighbors, by calculating distances to every existing point, then sorting them to find the closest ones.

a (2 point) How would you 'index' the blue points (vectors) so that we can avoid processing the entire collection of them? You can simply describe your strategy in a few lines (no diagram
needed). Hint: '4' :)

b (1 point). What if our vectors are in 3D (instead of the 2D ones shown), how would you index them?

c (1 point). There is a 'new' breed of DBs that make such similarity searches possible (LLMs can be hooked up to them, as well!) - what are they called?

Edit Pre

Please enter your response to Q5

Attach files Formatting tips

Q6 (3 points)

a (1 point). What is the purpose of 'discounting', in DCG for **search** results ranking?

b (1 point). The discounting factor is typically, 1/log2(rank). What if a hacker makes the discounting be this instead:

$$\mod \left[\sin\left(\frac{\pi}{4} \cdot rank\right)\right]$$

c (1 point) What would happen to the very notion of ranking, as we 'transition' from raw search results being presented, to Bard/ChatGPT summarized search?

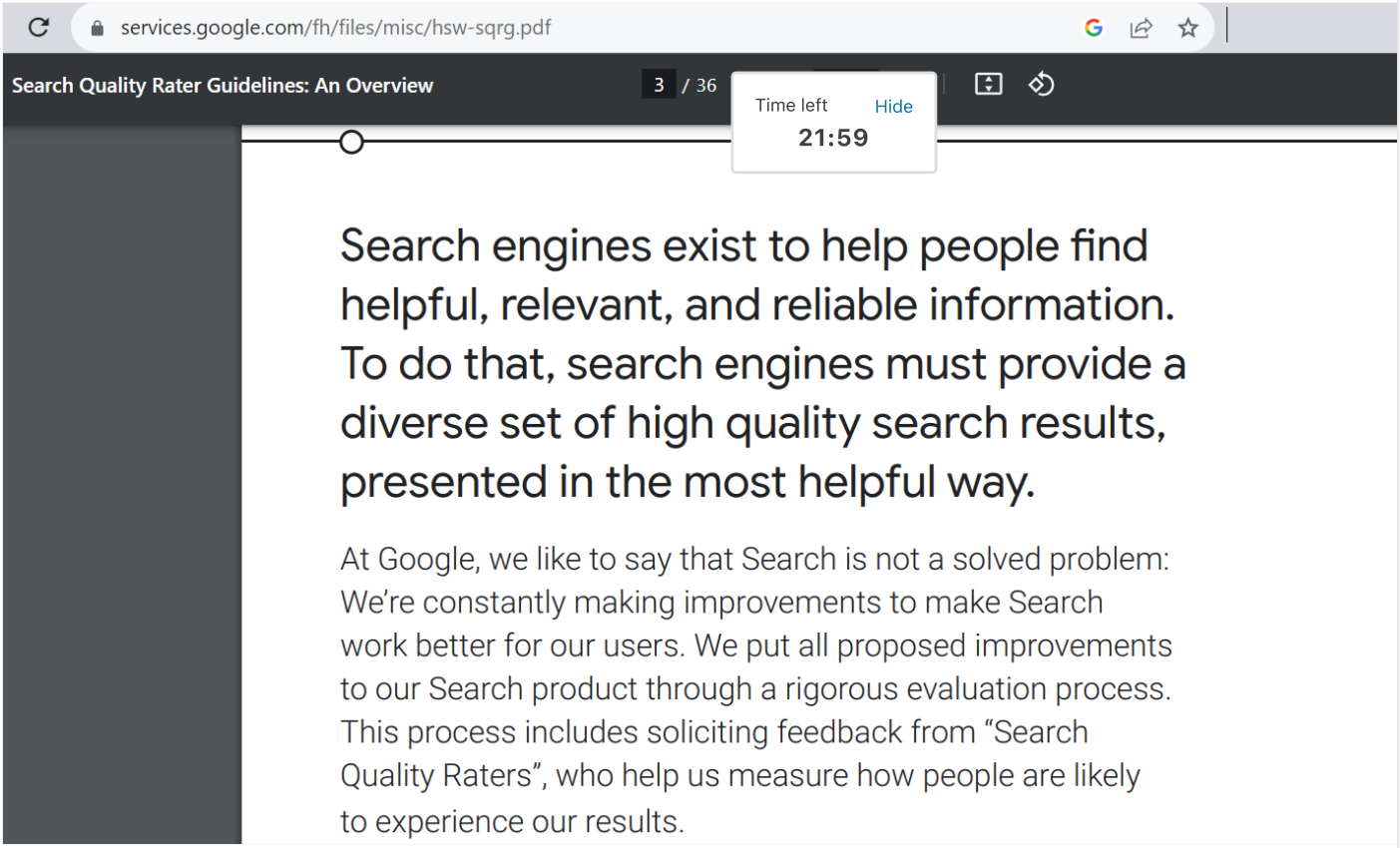
Edit Pre

Please enter your response to Q6

Attach files Formatting tips

Q7 (3 points)

Google uses humans, to evaluate the quality of **search**:



Why? Briefly explain the process in your own words.

Edit Pre

Please enter your response to Q7

Attach files Formatting tips

Q8 (2 points)

Both precision (P) and recall (R) are useful measures, for characterizing the results of a **search**. We combine these measures into a single 'F score' value, as follows:

$$F = \frac{2RP}{(R+P)}$$

Why do we combine P and R this way?

Edit Pre

Please enter your response to Q8

Attach files Formatting tips

Q9 (3 points)

Time left Hide
21:59

a (1 point). How have videos (eg. at YouTube) traditionally been indexed [where does the indexing

b (2 points). Going forward, given "recent advances in tech", how would/should videos be indexed, for a better **search** experience? Explain in a few lines, using an example or two.

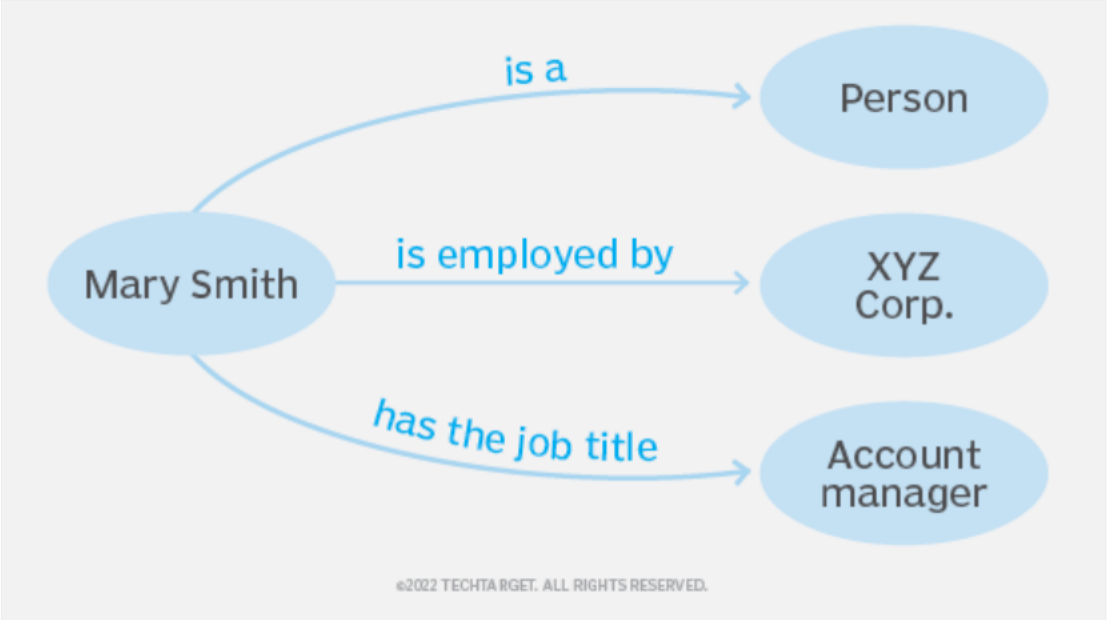
Edit Pre

Please enter your response to Q9

Attach files Formatting tips

Q10 (3 points)

Here is a simple 'RDF triple':



a (1 point). In the context of **search**, what are RDFs are used for?

b (2 points). RDFs are usually stored as XML, but they can also be stored as JSON (using a standard format called 'JSON-LD'). Disregarding JSON-LD's specifics, how would you express above RDF in your own simple JSON format? As a reminder, valid JSON looks like this [a set of key:value pairs, enclosed in a container { }]:

```
{
  "menu": {
    "id": "file",
    "value": "File",
    "popup": {
      "menuitem": [
        {
          "value": "New",
          "onclick": "CreateNewDoc()"
        },
        {
          "value": "Open",
          "onclick": "OpenDoc()"
        },
        {
          "value": "Close",
          "onclick": "CloseDoc()"
        }
      ]
    }
  }
}
```

Time left Hide
21:59

Edit Pre

Please enter your response to Q10

Attach files Formatting tips