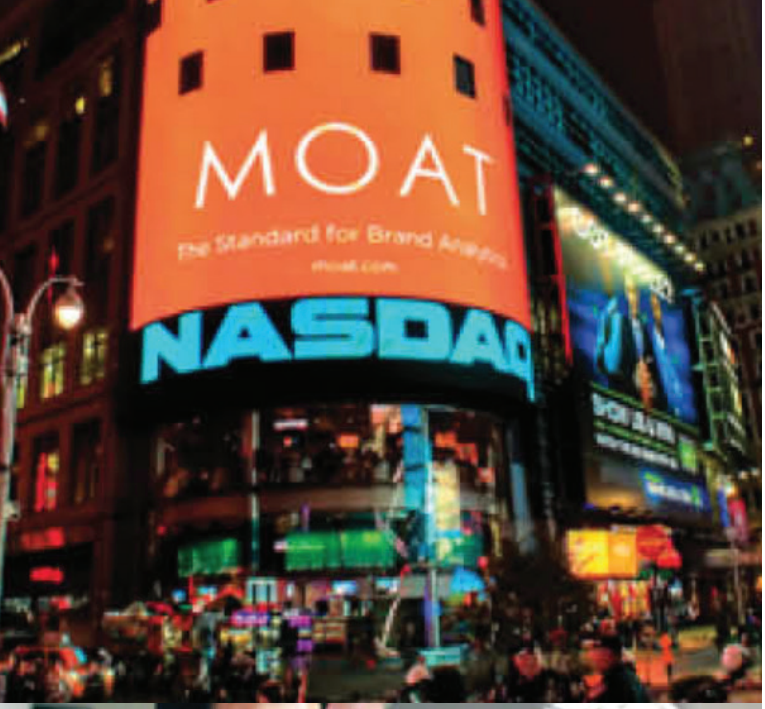
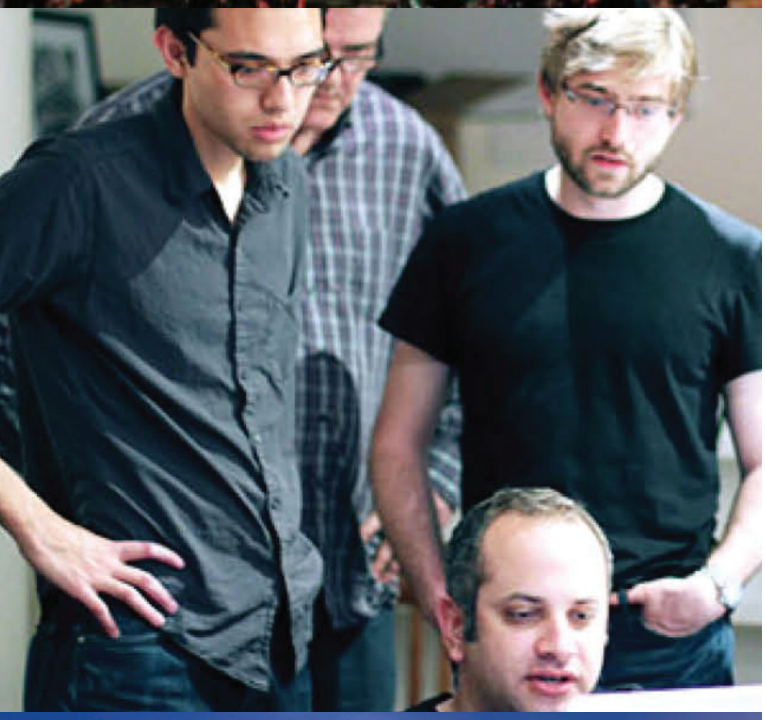




MOAT

Moat is located in the heart of the Flatiron District, the epicenter of New York's start-up scene, full of developers, entrepreneurs, and investors with great ideas.

We develop cutting-edge methods of measuring the effectiveness of online advertisements, computational methods for analyzing the resulting data, and a platform for accessing business insights from the data. We are building this product because we believe in helping publishers measure the value that online ad impressions create for their advertisers.

Our simple and elegant products help solve real problems for advertisers, publishers, and consumers, disrupting a $100B market.

We write clean code in a maintainable and robust way. We use open source tools, specialized build processes, and many homegrown systems-level and mathematical components that push around and analyze terabytes of new data each day. We regularly profile and optimize our code to visualize the web's data in real time.

More about Moat

Moat is a New York based SaaS company focused on transforming brand advertising online, establishing industry leading metrics and benchmarks for measuring both advertising and content engagement and effectiveness beyond traditional measures like clicks and social media shares.

MOAT

Question 1:

QA analyst

Go to www.moat.com and tell us how you would test the Moat Ad Search engine functionality and the results page. Please provide detailed test cases you would perform and the sort of bugs you would look for.

Question 2a:

Front-End Web Programming

Create a web app that calculates the distance (in nautical miles) between two airports. The app should autocomplete the airports and should feature all airports in the U.S. only. Bonus: plot the trip on Google maps. If you are using npm, please do not include your node_modules folder and make sure that all your requirements are in package.json.

Question 2b:

User Experience Design

Design a web app that calculates the distance between two airports. We're looking for wireframes, a prototype, and a written description of how a user would interact with the site. How do the elements on the page change as the user uses the app? Include any additional features in your prototype and wireframes that would make this app more useful and improve its functionality.

Question 3:

Data Structures

Consider a function incr_dict, which takes two arguments, which behaves like this in Python:

```
>>> dct = {}
>>> incr_dict(dct, (‘a’, ‘b’, ‘c’))
>>> dct
{‘a’: {‘b’: {‘c’: 1}}}
>>> incr_dict(dct, (‘a’, ‘b’, ‘c’))
>>> dct
{‘a’: {‘b’: {‘c’: 2}}}
>>> incr_dict(dct, (‘a’, ‘b’, ‘f’))
>>> dct
{‘a’: {‘b’: {‘c’: 2, ‘f’: 1}}}
>>> incr_dict(dct, (‘a’, ‘r’, ‘f’))
>>> dct
{‘a’: {‘r’: {‘f’: 1}, ‘b’: {‘c’: 2, ‘f’: 1}}}
>>> incr_dict(dct, (‘a’, ‘z’))
>>> dct
{‘a’: {‘r’: {‘f’: 1}, ‘b’: {‘c’: 2,’f’: 1}, ‘z’: 1}}
>>>
```

except that it creates any necessary intermediate and leaf nodes.

- Implement incr_dict in Python or any language that supports dictionaries.
- Write executable tests for your function.
- Will your implementation continue to work well if the tuple is extremely long?

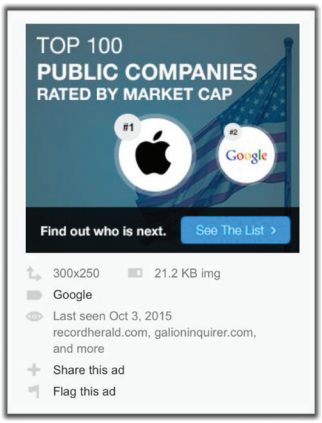
MOAT

Question 4:

Automation Of Test Data

We would like you to create a program that outputs mouse coordinates (x position, y position, and time stamp) simulating how a real user would interact with this ad: http://www.moat.com/search/results?brand[0]=google&ad=10859111

Please generate mouse data for this ad, assuming that the “Sign up now” button attracts the eye and tends to cause people to slow down around it. Please also provide an explanation for how users respond to the button in your simulation of a users mouse activity.



- Assume the user interacts for 15 seconds and provide mouse coordinates at intervals of 100 milliseconds.
- This program should run without any user input as it is simulating a user and not actually collecting data.

Question 5:

C Language

Write a C program to count blanks, tabs, open braces, closed braces, open brackets, closed brackets and newlines of a single file specified by its filename on argv and output the count in a human readable way. Run the program on its source code and include the output with your answer. If you are not confident in the C language, feel free to write this program in either Python or Javascript.

Bonus Question:

What's the current bottle neck to your answer above? How can you tell? Can you make your answer to the question above faster by changing your code? What compiler flags affect the performance the most?

Question 6:

UNIX scripting

Please complete these problems in Shell. Feel free to implement these problems in multiple languages and compare the ease of writing them along with the ease of maintaining them. Make sure that your solutions correctly handle filenames that include spaces.

a. Write a script that counts the lines of each of the Python files in the current directory. Print the countsnext to each filename.

b. Write a script that counts the lines of each of the Python files in the current directory and all descendant directories. Print the counts next to each filename.

c. Write a script that sums the total number of lines in all the Python files in the current directory and all descendant directories, even if there are many, many files. Print the total.

Question 7:

Data Structures

Implement a hashmap (with amortized constant time look-ups) in Python without using a hashmap primitive. Please include an executable testing framework for your data structure.

Question 8:

Wiki Crawler

Starting from a random Wikipedia article (example: http://en.wikipedia.org/wiki/Art) and clicking on the first non-italicized link not surrounded by parentheses in the main text and then repeating the process for subsequent articles usually leads to http://en.wikipedia.org/wiki/Philosophy. Please write a program that models this behavior and answers the following questions, while making as few http requests as possible.

Questions:

- What percentage of pages lead to philosophy?
- Using the random article link (found on any wikipedia article in the left sidebar), what is the distribution of path lengths for 500 pages, discarding those paths that never reach the Philosophy page?

Question 9:

Automation Engineer

Please write a small handful of Selenium tests in Python against http://www.moat.com (http://www.moat.com/):

1. Verify that the “Try These” links are random and that they work.
2. Verify that the “Recently Seen Ads” are no more than half an hour old.
3. Verify the ad count on the search result page is correct, even if there are more than 100 ads in the result set.
4. Verify the “Share this Ad” feature.

Question 10:

Six degrees of Wikipedia

Make a web app that allows a user to query for the shortest path between two pages in Wikipedia.

Data:

Here's a SQLite database file that contains id->name mappings for each page in Wikipedia and id-to-id mappings of each page link:

https://s3.amazonaws.com/search-archives/interview/wiki.sql

If you don't want to use SQLite, feel free to use another datastore.
Here's a dump of the page names (the page id is the line number + 1):
http://search-archives.s3.amazonaws.com/interview/pages.txt

And here's a dump of the page links (the format for each line is source_page_id: destination_page_id_1 destination_page_id_2 ...):
http://search-archives.s3.amazonaws.com/interview/links.txt

If you are using a different datastore, please include scripts that load data from these files so we can replicate your work.

What we're looking for:

- Clean, bug free code
- An intuitive UI for the user
- A clearly defined runtime environment (i.e. virtualenv, Docker, npm + package.json, etc.)
- Good performance

MOAT

Question 11:

Data Processing

The statement of the problem:

When a digital ad loads on a page, Moat's code runs in the browser and measures the ad experience. It sends messages to Moat's servers when the ad loads (known as an ad impression), when the ad becomes visible on the user's screen (known as a viewable impression), and periodically to update the sum of in-view time on screen. Moat's data pipeline parses and aggregates these messages to produce reports for its clients. The client dashboard shows total impressions, total viewable impressions, and average in-view time (defined as sum of in-view time divided by viewable impressions).

Write a working program in a language of your choosing to transform and aggregate the event logs defined below into the TSV format specified below. Discuss your design decisions and how you would envision your program to be run in production.

Here is the template of an event log:

<obfuscated IP address> [<timestamp>] "GET /pixel.gif?<query string> HTTP/1.1" 200 43 "<user agent string>"

The query parameters are:

- event_type: The type of event that triggered the message being sent from the browser to our servers. Possible values are "ad_load" and "heartbeat". Remember that a given impression in general corresponds to multiple event logs.
- in_view_time: The time (in milliseconds) since the ad load. Only present when e="heartbeat".
- impression_id: The value of this field is a random number, generated at the start of the ad impression,
- that we may assume uniquely identifies the ad impression. All logs for an individual impression have the same impression_id.
- account_id: Uniquely identifying label of the account whose metrics the event log will be used to compute.
- campaign_id: A label identifying the specific campaign the ad is part of. Beware that multiple accounts
- may make use of the same label for a campaign ID.
- creative_id: A label identifying the specific creative. The same caveat as for campaign_id (above) applies to creative_id.

TSV Columns

- account_id
- campaign_id
- creative_id
- date
- impressions
- viewable_impressions
- sum_in_view_time

Sample data may be downloaded at: bit.ly/moat-pipeline-data

MOAT

Question 12:

Auto-Refresh

The statement of the problem:

Some web pages auto-refresh, i.e. they reload themselves and/or the ads they contain periodically, independent of user activity. It's worthwhile to detect when pages auto-refresh because an advertiser might consider auto-refresh ad impressions less valuable than impressions that happen on the initial page load.

We provide impression-level data from ads served on a set of web pages across a single domain. (The domain and web page URLs have been anonymized.) Your task is to define a quantifiable measure of the likelihood that a given page auto-refreshes that could realistically be used in a real-time production environment to detect auto-refreshing from a stream of logs of ad impressions served on a page. We will test your code on the provided data set to see how well it classifies pages with respect to auto-refresh.

Please provide:

- A description of your approach, with any necessary definitions.
- The results of your analysis.
- Working code (notebook or code that can be run from the command line) that can be used to reproduce your results.

Link to the data: bit.ly/moat-autorefresh-data