

# Preparing For Insight Data Engineering

Insight Fellows Bridge the Gap series  
March 18, 2020



# Who is this session intended to help

- Applicants who want an overview on the Insight Data Engineering Program, application process, and coding challenge

# Who is this session not intended to help

- Applicants who want to learn how to code
- Applicants who want more information on how to do well on the data science, health data science, artificial intelligence, devops engineering, decentralized consensus, security fellowship process (Please check out the Bridge the Gap series for those programs)

# What is Insight Data Engineering?

- Intensive 7 week full-time project-based training
- Need-based scholarships available
- No education requirement
- Build scalable, distributed data pipelines
- Self-directed collaborative learning, no classes
- New York, Boston, Los Angeles, Seattle, and Silicon Valley
- Each program has separate admissions process

# INSIGHT IN A NUTSHELL



# Admissions process

1. Apply at <https://apply.insightdatascience.com/apply/programs>
2. Coding challenge with a week to complete
3. One or more 15-30 minute remote interviews
4. Decision

# What do Data Engineers do?

- Develop, construct, test and maintain architectures for big data
- Align architecture with business requirements
- Data acquisition
- Develop data set processes
- Identify ways to improve data reliability, efficiency and quality
- Use large data sets to address business issues
- Deploy sophisticated analytics programs, machine learning and statistical methods
- Prepare data for predictive and prescriptive modeling
- Use data to discover tasks that can be automated

# So you want to be a data engineer...

Data engineering is software engineering with a speciality in data

To be a good data engineer, you also must demonstrate that you can be a **good software engineer**

- You know how to code
- You can solve problems using code
- You can contribute production-quality code

# What does the Insight DE coding challenge test for?

- Can read a problem statement, understand it and solve using code
- Knowledge of basic data structures
- Able to write clean, modular code using software best practices
  - Readable, modular
  - Tested
- Knowledge of Linux and basic command-line
- Familiarity of version control tools (i.e., git)
- Communication -- able to explain how you solve a problem



# Can read coding challenge, understand & code it

- We spend a lot of time preparing and writing the problem statement presented in the coding challenge we send
- If you have questions that prevents you from completing the coding challenge (e.g., suspected error or contradiction in challenge description or input and output data) email [cc@insightdataengineering.com](mailto:cc@insightdataengineering.com)
  - But first, re-read the Readme
  - Don't email questions for the sake of contacting us -- we receive hundreds of emails and try to respond to each and every one

# Know your basic data structures

A few basic ones

- **Lists**: sequence of data that can grow
- **Arrays**: sequence of data than can be indexed by number
- **Dictionaries** (or associated arrays or key-value pairs):  
sequence of data that can be indexed by key
- **Sets**: unordered collection of unique elements
- **Trees**: recursive data structure with parent and child nodes
- **Heaps**: special tree also known as priority queues

# Know your basic data structure

**Use the right data structure for the job** Which data structure would you use?

Need to iterate through every element where order is unimportant	Array or list
Need to get one or two values based on a key	Dictionary
Need to identify connections or relationships	Trees
Need to find the median	Heap

# Clean, modular code & software best practices

- Write clean and modular code that other people beside you can read
  - consistent style
  - meaningful names for variables functions
  - If the code can be reused, create a module
  - Judiciously comment your code -- take an entire line to comment

```
int count = 0 #keeps count
```



```
# keep count of products  
int count = 0
```



- Provide exception handling and tests to ensure your code works as expected
- Luckily, many clean-code resources available online
  - Python: <https://docs.python-guide.org/writing/style/>
  - Java: <https://docs.python-guide.org/writing/style/>
  - Scala: <https://twitter.github.io/effectivescala/>
  - C++: <https://github.com/lefticus/cppbestpractices/blob/master/03-Style.md>

# Knowledge of Linux, basic command-line

- Data engineers must be comfortable navigating Unix and shell scripting commands
  - DE program is not for you if you want to work exclusively in Jupyter notebooks or IDEs
- For the challenge, you need to know only the basics.

Eg: `cd`, `chmod +x filename.sh`

- Luckily, many resources available online
  - <https://www.howtogeek.com/412055/37-important-linux-commands-you-should-know/>
  - <https://www.guru99.com/must-know-linux-commands.html>
  - <https://www.geeksforgeeks.org/introduction-linux-shell-shell-scripting/>

# Familiarity of version control i.e., git

- Git is the most popular open-source version control software
- Version control allows you to track changes over time and is essential for team collaboration
- Documentation available: <https://git-scm.com/docs>

# Communicate how you solved it

- Data engineers need to communicate with other engineers, data scientists, business users and others
- Documenting the code that you wrote is essential for others to understand what you did

# How the Insight coding challenge is organized

- Applicants typically receive challenge via Github link
- Challenge consists of problem statement via Readme
- In the past, applicants asked to read input file(s), do some processing and write results to output file(s)
- Applicants expected to provide Github link to their codebase
- For best results, code in Python, Java, Scala or C++
  - Do not use R, Matlab, C#
  - Do not submit Jupyter notebook



# General Insight coding challenge directory structure\*

```
|-- Readme.md
|-- run.sh
|-- src
|-- input
|-- output
|-- insight_testsuite
    |-- tests
        |-- test_1
            |-- input
            |-- output
```

\* Subject to change in the future

# Coding challenge directory structure

```
|-- Readme.md
|-- run.sh
|-- src
|-- input
|-- output
|-- insight_testsuite
    |-- tests
        |-- test_1
            |-- input
            |-- output
```

While the coding challenge Readme contains instructions on what we are asking you to program, when you submit your solution, you should re-write the Readme to clearly describe your approach to solving the challenge

# Coding challenge directory structure

|-- Readme.md

|-- run.sh

|-- **src**

|-- **input**

|-- **output**

|-- **insight\_testsuite**

    |-- **tests**

        |-- **test\_1**

            |-- **input**

            |-- **output**

This subdirectory is where the code you write should reside

# Coding challenge directory structure

|-- Readme.md

|-- run.sh

|-- **src**

|-- **input**

|-- **output**

|-- **insight\_testsuite**

    |-- **tests**

        |-- **test\_1**

            |-- **input**

            |-- **output**

This subdirectory is where your code should expect to read any input files

# Coding challenge directory structure

```
|-- Readme.md
|-- run.sh
|-- src
|-- input
|-- output
|-- insight_testsuite
    |-- tests
        |-- test_1
            |-- input
            |-- output
```

This subdirectory is where your code should expect to write any output files

# Coding challenge directory structure

```
|-- Readme.md
|-- run.sh
|-- src
|-- input
|-- output
|-- insight_testsuite
    |-- tests
        |-- test_1
            |-- input
            |-- output
```

You must modify this script to run/invoke your code,  
and compile it if necessary (e.g., Java, C++)

# Coding challenge directory structure

```
|-- Readme.md
|-- run.sh
|-- src
|-- input
|-- output
|-- insight_testsuite
    |-- tests
        |-- test_1
            |-- input
            |-- output
        |-- write_your_own_test
            |-- input
            |-- output
```

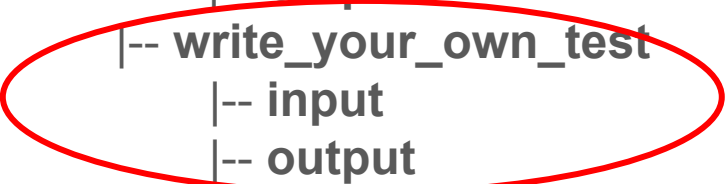
This subdirectory is where tests can be found and added to. Insight usually provides test\_1

But you can add a new directory with your own test cases

# How to add your own tests and test your code

Create a new subdirectory under **tests** with additional input and output subdirectories. In this case it's named **write\_your\_own\_test** but re-name it to a more appropriate name for your test

```
|-- insight_testsuite
    |-- run_tests.sh
    |-- tests
        |-- test_1
            |-- input
            |-- output
            |-- write_your_own_test
                |-- input
                |-- output
```





# How to add your own tests and test your code


In the **input** directory, create a file (of if the challenge calls for it, multiple files) with the data that you want to use as input to test your code

```
|-- insight_testsuite  
    |-- run_tests.sh  
    |-- tests  
        |-- test_1  
            |-- input  
            |-- output  
        |-- write_your_own_test  
            |-- input  
            |-- output
```

# How to add your own tests and test your code

In the **output** directory, create a file (of if the challenge calls for it, multiple files) with the data you would expect your code to output if it were correct/pass your test

```
|-- insight_testsuite
    |-- run_tests.sh
    |-- tests
        |-- test_1
            |-- input
            |-- output
        |-- write_your_own_test
            |-- input
            |-- output
```



# Common questions / issues

- Follow instructions on how to submit code
- Do not check-in Jupyter notebooks with your code
- Use only built-in or standard libraries
  - Do not use pandas or add-on libraries that need to be downloaded and installed locally (e.g., in Python, if you need to 'pip install' it first then it's non-standard)
- `run.sh` has been modified to execute your program (e.g., make sure the command to run your code is not commented out)
- Test your code using `...`

Questions?