2019 Level UP: Take Home Assignments

Created by Melissa Geneva Martinez, last modified by Andrea Maria Simon on Sep 13, 2019

The Level UP Staff Engineer/Architect nomination, process requires you to take a Take Home Assignment. The **SLA for completing this assignment is 72 hours,** and will be due at the date and time committed by you and indicated on your **Level UP Take Home Assignment** email.

To submit your solutions, please upload your code utilizing Bitbucket/GitHub. Include any supporting documents, files, links to repositories, etc. No PDFs. *Be sure to grant all panelists access to any restricted pages or repositories.* The panelist list can be located Level UP Panelists.

Assessments are Level UP specific and should be done **individually without any collaboration**. **Do not share assessment topics with your colleagues or people external to American Express.**

Below are the available assessments. Choose **ONE (1)** assessment in the domain that you are most confident and email your selection to Level UP (TechnologyLevel-Up@aexp.com), with your selection provide the date when you can commit to start taking the assessment. **Note:** Level UP needs **7 business days** for the selected Panelists scheduled for your interview to review, so please revert to your interview calendar date when determining when you can take this test. However, **it is high suggested that you take this assessment sooner than later**. Once Level UP receives this information, access to the selected test will be granted on the date you've committed for test taking.

Assessment	Assessment Description
C#_Java Quaternary Tree Exercise #1 (backend)	In a language of your choice, design and implement a reusable class for Quaternary Trees.
C#_Java Amicable Arrays Exercise #2 (backend)	In a language of your choice implement a function which takes two square arrays and returns true if the arrays are "amicable".
C#_Java Snakes and Dragons Exercise #3 (backend)	Implement a function in a language of your choice which takes a 2 dimensional array as input and returns an integer which represents length of a snake found in the array.
C#_Java Plane Heading Exercise #4 (backend)	Design and implement in a language of your choice two executables, "Plane" and "Pilot".
C#_Java Hexagon Array Exercise #5 (backend)	Design all the interfaces and implement a reusable class for a Hexagon Array of integers.
COBOL (Python) SQL Parser Exercise #1	Create a SQL parser that reads an SQL statement input through SYSIN (or a file).
COBOL (Python) Binary Conversion Exercise #2	Create a general purpose COBOL module that converts the binary content of any EBCDIC string into BASE64 characters.
PYTHON Merchants and Cardholders Clusters Exercise #3	Create a function for a platform to divide their cluster of merchants and card holders into clusters where each cluster serves a group of merchant and card members so that it has no chattiness with other clusters.
COBOL (Python) Sort Input File Exercise #4	Write a general purpose COBOL program to sort an input file.
COBOL (Python) Number Print Program Exercise #5	Write a general purpose number print program
COBOL (Python) Highest Number of Merchant Transactions Exercise #6	Write a main program A and a sub program B. The main program reads input transaction file. Each record has Account#, Date and Amount. Program A calls program B for each record to help accumulate and keep track of top 5 merchants that has highest number of transactions.
Java (Python) Applied to Big Data Distributed Mode Output Exercise #1	Write a program that will run in a distributed mode.
Java (Python) Applied to Big Data Parquet Output Format Exercise #2	Generate output data in the Parquet output format.
Java (Python) Applied to Big Data Word Mapping Exercise #3	Generate an output file that is effectively a mapping of the word, to the file names in which you found them.
Java (Python) Applied to Big Data JSON Output Files Exercise #4	Develop an output format, that can be used to write the given input from one file, as JSON output files.
Java (Python) Applied to Big Data Average of Tenure Exercise #5	Generate output using Parquet formatting.
Language Agnostic Applied to API_Microservices Exercise #1 (backend)	Create a gRPC service that will simply route a Transaction message to specific backend gRPC services based on the "Destination" field.
Language Agnostic Applied to API_Microservices Exercise #2 (backend)	Create a backend API server for a HR system as well as a Client that can pull an extract of all employees within the HR API.

Remember that all Panelists must have access to your solutions, and **MUST** be received by Level UP no later than date and time indicated in your **Level UP Take Home Assignment** email.

For any questions, please contact Level UP (TechnologyLevel-Up@aexp.com).

Be successful!

No labels