Avamore Junior Software Developer Test

Introduction:

As you are aware, Avamore Capital is seeking to develop and implement a highly automated lending process. The project is at an early stage and, at the current time, this code is being written in Python and a web interface has been written utilising CSS (for styling), HTML (for website templates), Java (for functionality).

The objective of the test is to evaluate three sets of skills which will be relevant to the role/project: (1) Writing Python code, (2) Creating web interfaces to Python code and (3) problem solving. Further details are set out below.

Overview of Test:

There are three parts to the test, corresponding to the three sets of skills required.

Part 1: Writing Python code

Attached to this test is an excel model ("ARM Template_Simplified"). This excel model is a simplified form of what we call our 'Redemption Model'. The Redemption Model calculates how much interest is due on a loan at the time of repayment. It takes into account a number of different factors such as, the amount of the loan, the interest rate on the loan, whether the loan went into default (breached the terms of the contract) and therefore a penalty interest rate is due.

On the "Inputs" tab of the excel, you can see all the inputs which go into the Redemption Model. On the "Calculations" tab you will see (1) all the calculations that are made from these inputs and (2) total interest due on the loan in cell B3 (in red font, highlighted in yellow). This total interest due on the loan is the output of the model.

Part 1 of the test is for you to write a Python script for this Redemption Model excel, where the output is the total interest due (i.e. the equivalent of cell B3 in the "Calculations" tab of the excel model). The only flexible inputs which your Python script needs to have are the 4 cells highlighted in green in the "Inputs" sheet. Specifically, these 4 cells are:

- Cell C15 (Facility A (Land Advance))
- Cell C22 (Contractual Monthly Rate (%pm))
- Cell C28 (Beginning of Default Period)
- Cell C29 (End of Default Period: Cell)

Apart from these 4 inputs, none of the inputs need to be flexible in your Python script. When run, your Python script should provide the same total interest due on the loan as the Redemption Model excel. When any of the 4 inputs are changed in the Redemption Model excel and the Python script, both the Redemption Model and your Python Script should continue to calculate the same total interest due.

We will review your Python script by (1) reading the script to see how the code has been written and (2) testing the Python script by changing these 4 inputs in the excel model and in your Python script to see that the numbers continue to be the same.

Please provide a copy of your Python script on Github or by copying and pasting it into a word document.

Part 2: Creating web interfaces to Python code

A key part of the project is to have a web interface to the underlying code which is executing / automating the processes of the business. Part 2 of the test is for you to create a web interface to your Python script of the Redemption Model (from Part 1 of the test).

The web interface does not need to have any authentication. The only two requirements of the web interface are (1) the ability to input all 4 of your flexible inputs from your Python script of the redemption model and (2) an ability to execute your Python script from the web interface and for the web interface to show the user the "total interest due" output.

We are expecting the web interface to be written in CSS, HTML and Java. We expect the web interface to call your Python script and for the calculations to be executed by your Python script. If you feel strongly that you wish to build your web interface other than with CSS, HTML and Java then you may do so however it would be our strong preference that the web interface is written with CSS, HTML and Java.

We will review this part of the test by (1) reviewing the required functionality of your web interface (such that any combination of the 4 inputs entered provides the correct total interest due output) and (2) reviewing the code you have written to create this web interface. The format of your web interface will be noted and, naturally, a better formatted web interface will be viewed favourably. However, the formatting of your web interface is not a key aspect of the assessment.

Please provide (1) a link to your web interface and (2) a copy of your code on Github or by copying and pasting it into a word document.

Part 3: Problem solving

The Junior Software Developer (JSD) at Avamore is a broad role. The JSD will be the key team member of the software development project, supported predominantly by the co-founders of the business (but also potentially other members of Avamore as appropriate). The JSD, alongside the co-founders of the business, will be required to problem solve to address challenges with implementing the Company's visions for a highly automated lending process. Problem solving will range from software/coding challenges to challenges with data or the interpretation of data to achieve the desired results.

While this problem solving will be very much a joint task of the JSD and the co-founders, it is important that the JSD is an active contributor or driver of problem solving. Part 3 of this test is designed to understand your approach to problem solving.

The problem in this test is how to understand and interpret property prices in an area. One of the tasks Avamore undertakes on a daily basis is to evaluate the value of properties in various areas. This task is essential because Avamore provides mortgages (in other words, loans secured against property) and the value of the property is a key criteria through which loans are assessed and granted (or not granted).

We have written an internal Python script to provide us information on all properties that have been sold in a particular area. The idea behind this is that, in order for us to understand the value of a property in a particular area, we look at what similar properties in the area have previously sold for and review this comparable data.

Attached to this test is another excel file (“Sold Properties_Example”). In this excel file, we have run our internal company Python script to find all sold properties within 0.5 miles of a particular postcode (CR0 0RY) that meet three criteria:

1. sold after 31 December 2018
2. size of property sold is between 1,100 and 1,350 square feet
3. houses only (excluding flats)

As you will see from the excel, there is a wide range of price points here, ranging from £166,495 to £630,000. The properties are all of a similar size (1,100 to 1,350 square feet), so it is surprising that there is such a wide range of sold property prices.

As part of the process of automating our lending process, we need to automate our process of valuing properties in different areas. Suppose we are aiming to evaluate the price of a terraced house in the postcode CR0 0RY, which has a size of 1,200 square feet. Please answer the following questions:

1. How would you suggest we value this property based on the data provided in the excel (“Sold Properties_Example”)?
2. What challenges or risks do you see in your proposed valuation method?
3. What further information would help you have greater confidence in your valuation of the property?

Please provide a written response to the above three questions (we are not expecting any code to be written in this part of the test).

If you have any questions during the test, please email Zuahir Mirza at zm@avamorecapital.com

Once complete, please email your answers to zm@avamorecapital.com and md@avamorecapital.com cc’ing lew@avamorecapital.com