INSTRUCTIONS

You can Google and consult reference materials,   but this should be your work only.
Skip any questions that are in an unfamiliar area.

Email us back a zip/tar file of written documents and source code with your answers to parts 1-4.

Please do not post the questions or solutions in public.
DO NOT post your solutions onto a GitHub repo, and DO NOT post the questions onto StackOverflow. You will be eliminated from consideration as an applicant.

DESIGN EXERCISES:

Imagine you are launching a brand new Software-As-A-Service offering. Your new product is a 4-function calculator as a service, and you are providing a REST API to allow users of your API to add, subtract,multiply, and divide numbers.   You are hosting this new API on a new domain and you control the URLs available on this domain.

For example, your new REST API service allows callers to make a request to add 2 plus 2, and will respond that the sum is 4.   It allows callers to make a request to multiply 3 times 3, and it will respond that the product is 9.

Your 4-function calculator service will be launched as a free offering so authorization and API keys are out of scope for the service.

Part 1:
Provide a written document (in text or PDF format) describing your API that you could give to your customers or a front-end developer team. Design the REST API of your product to expose these 4-function calculator operations.   What do callers need to know in order to use your API? The document should provides instructions on how to use your API to do the 4 basic math operations (add two numbers together, divide one number by another number, etc.) that your product features.

Part 2:
Provide a written test plan document (text or PDF) for ensuring the quality of your 4 function calculator service.

Please keep in mind when writing your test plan:

- What are the key positive ("happy path") test cases that should be covered in the test plan?
- In addition to positive tests, what other *kinds* of tests should be covered?
- What is the special negative test case that must be validated for one of the calculator functions?
- What else might be important to test in a production REST API service offering?

Your test plan needs to at minimum enumerate a set of test cases that should be verified for the REST API that you define in part 1.
Provide an English-language test plan. DO NOT provide actual test assets from a specific testing tool. DO NOT submit code or scripts. Your test plan should not depend on a particular implementation technology or testing tool for the test suite.

Include the documents for Part 1 and Part 2 in the zip/tar you submit to us.

CODING EXERCISES:

Please implement the solutions to the coding exercises in any of Python, Javascript, C, C++, Java, C#, or Scala.

Please provide detailed instructions on how to run your program on a Windows, Mac, or Linux machine, including any compiler and runtime versions, etc. Please do not assume we have specific IDEs or build or dependency tools (Python, JDKs, Maven, npm, etc) already installed. The source code files you supply must compile and run when we follow the instructions you provide. Your instructions must allow us to run your source code files without modification.

Please implement both exercises in the same programming language.

Part 3:
Consider a car that depreciates at 10% a year. If the car is originally worth $20,000, then after the first year it is worth 90% , ie. $18,000. After the 2nd year it depreciates by another $1800 and is worth $16,200. After the 3rd year it depreciates another $1620. And continues to depreciate year after year after that.
Implement two functions:

1. Implement a recursive function that takes as input the price of the car, and returns how many years before the car is worth $2000 or less. If the car is originally worth $2000 or less, it should return 0 years.
2. Implement the exact same function, without using recursion.

Part 4:

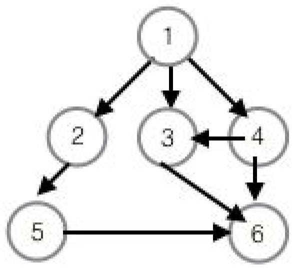
Consider a directed graph of small integer values, where each integer is a positive number and each integer is unique. Each unique integer node has zero or more child nodes.

Write 3 functions that:

1. Create a node in a graph.
2. Inserts a node as a child of an existing node.
3. Print a graph.

Do not use an existing graph library. You need to write the 3 functions listed above.
Create and Insert must ensure in a language-idiomatic way that the graph is acyclic - a node must not directly or indirectly point to itself. They must also ensure that the values are unique.

Here is an example of a directed acyclic graph:



This should print out as:

1-> 2, 3, 4
2-> 5
3-> 6
4-> 3,6
5-> 6
6-> No Children

Demonstrate that your functions are correct by including the above example. Also include a verification that a graph cannot be modified to contain cycles and a verification that a graph cannot be modified to contain duplicate nodes.

A test case that must fail due to cycles is:

Create graph node 1
Create graph node 2
Create graph node 3
Insert node 2 as a child of node 1
Insert node 3 as child of node 2
Insert node 1 as child of node 3 -> This step must not create a cycle.