Terry Niner

From: Matt Christensen | Campspot <careers+f30nd36v@campspot.recruiterbox.com>
Sent: Monday, August 13, 2018 7:02 AM
To: TNINER@COMCAST.NET
Subject: Campspot Development Team Lead Programming Challenge
Attachments: test-case.json

Good Morning Terry,

Thanks for your time in the phone interview last Friday. We're pleased to let you know we're moving you to the next stage of our hiring process. At this stage we'd like you to complete the programming challenge described below. In addition, please send us 2-3 references.

We look forward to reviewing your solution.

Thanks,

Matt

==========
Campspot Programming Exercise
==========

Most destination resorts do not host guests who are willing to stay for only one night. Their guests are simply not passing through the area; they've often travelled great distance with lots of luggage specifically to stay there, and it's not worth it for them to stay just one night. As a result, a reservation system for resorts needs to prevent one night gaps in its reservation grid, or schedule of inventory. If one-night gaps occur on the grid, those vacant rooms within the gap will not be sold, and this results in lost income for the resort owner. For example, Campspot uses a one-night gap rule, as this is a common problem for camping resort owners. However, owners' gap problems aren't limited to one-night gaps. Some resorts only host guests who stay a minimum three or four nights. Those resorts may want to prevent one-, two-, and three-night gaps.

To support this use case, Campspot has a business rule called a "gap rule" that allows resort owners to define which types of gaps are not allowed.

Your task is to implement the most basic version of a gap rule. This will prevent new reservations from creating one-night gaps between themselves and existing reservations. It will accomplish this by looking at the existing reservations and the dates we are booking, and returning only campsite names that can accommodate the new booking without creating gaps.

Deliverables we expect from you include:

1. An executable program that takes in a standard JSON input file (described below) and returns which campsites are available to book

2. Tests for your program

3. Written documentation that explains:

a. How to build and run your program

b. A high-level description of your approach to solving the problem

c. Any assumptions or special considerations that were made

We've provided a sample input file (attached to this email), which you can use to test your program.

The expected output of the sample file is:

"Comfy Cabin"

"Rickety Cabin"

"Cabin in the Woods"

The JSON input file contains a single object with the following keys:

"search": the dates of the reservation we are hoping to make

"campsites": An array of the campsites that are available to book, each has an id and name.

"reservations": An array of the existing reservations that must be considered. Each has a campsite ID, start date, and end date

You can use any programming language you want, and the interface is up to you (it can be a command line program, a web interface or anything in between). There is no time limit on the exercise, but it should reasonably be completed within 2-3 hours.

Evaluation and Delivery:

Things we will be looking at when evaluating your submission:

1. Does your code work according to the provided specifications

2. Is your documentation clear and insightful

3. Is your code well-structured and maintainable

4. Are your tests well-structured, robust, and maintainable

Some questions we often ask ourselves when looking at submissions:

1. Does this code demonstrate an attention to detail?

2. How easy would it be to extend this code to add additional features?

3. How easy would it be to change any assumptions you made about the problem?

4. How easy would it be to plug this code into a different interface like a web service?

Matt Christensen

Operations Manager

p: 616-226-5500 x 108

e: matt.christensen@campspot.com