85 points

You are part of a team working on an application that provides the functionality to manage many sweepstakes. Marketing firms are likely purchasers and users of the backend application, and they can choose to use your functionality however they like! It is up to you, the developer, to create the backend application to implement the logic for the functions and pieces of functionality requested below.

NOTE: All “print” and “input” function calls should only exist on the UserInterface class.

(5 points) As a developer, I want consistent commits and descriptive commit messages.

(5 points) As a developer, I want to create a Contestant class that has a first name, last name, email address, and registration number.

(10 points) As a developer, I want to create a user interface for any information the application would need to get or display for the user. One example would be to create a method called “get_user_input_string” that takes in a prompt and returns the user’s entered input.

(15 points) As a developer, I want to create a MarketingFirm class that uses the List data structure to hold its sweepstakes. The MarketingFirm class will have the following methods with full implementation (write the functionality) of each method:

- __init__(self, name)
- create_sweepstakes(self)
- select_sweepstakes(self)
- change_marketing_firm_name(self)
- menu(self)

(10 points) As a developer, I want the marketing firm menu to provide a façade interface for selecting a sweepstakes, creating a sweepstakes, changing the marketing firm name, and exiting the application.

(5 points) As a developer, once I select a sweepstakes from the select_sweepstakes method, I should be taken to that sweepstakes menu.

(15 points) As a developer, I want to create a Sweepstakes class that uses the Dictionary data structure as an underlying structure to hold contestants. The Sweepstakes class will have the following methods with full implementation (write the functionality) of each method:

- __init__(self, name)
- register_contestant(self, contestant)
- pick_winner(self) -- should return a contestant
- view_contestants(self)
- menu(self)

(10 points) As a developer, I want the sweepstakes menu to provide a façade interface for viewing contestants, registering a new contestant, picking a winner, and exiting the sweepstakes menu.

(10 points) As a developer, I want to use the observer design pattern to notify all users of the winning contestant, with the winner of the sweepstakes receiving a different message specifically congratulating

them on being the winner. This notification will be triggered within the sweepstakes pick_winner method.

Bonus Points:

(2.5 points) As a developer, if a winner is chosen for a sweepstakes that sweepstakes should not appear in the marketing firm's select_sweepstakes options.

(2.5 points) As a developer, I want to have the ability to unregister a contestant from a sweepstakes.

(5 points) As a developer, I want to send an actual email to a sweepstakes winner using an email API