

## PROBLEM 3.3.5

# GeoQuest




### INTRODUCTION

Have you ever gone on a scavenger hunt at a birthday party? For your final Unit 3 problem, you will create GeoQuest, an app that is like a scavenger hunt except it collects virtual objects instead of physical ones. The app will use the camera on your tablet or other mobile device to capture a variety of items in and around your school and include the items' geolocation.

#### Materials



- Computer with Android™ Studio and Genymotion™ installed
- Android™ tablet and USB cable
- Tools with which to create visual representations of your code
- Tools for creating and using a sprint task list and product backlog
- Presentation software

### RESOURCES

-  **GeoQuest Requirements**  
Resources available online
-  **Problem 3.3.5 Rubric**  
Resources available online
-  **Problem 3.3.5 One-Page Rubric**  
Resources available online

## Procedure


### Part I: Plan

- 1 Before you begin, read through this entire problem definition. You need to be well acquainted with all requirements in  **Problem 3.3.5 GeoQuest Requirements.**
- 2 Reference the  **3.3.5 Problem Rubric** throughout the development process.

- 3 The app will need the following information:
  - Items to find
  - Picture of each item found
  - Geolocation of the item
  - Record of items found
  - Indicator that all items have been found
- 4 Brainstorm a variety of designs, creating sketches or prototypes.
- 5 Define categories of items to find and some specific feature you will record about each item.
- 6 Consider security issues for your app. Just like the social networking app you created earlier, your GeoQuest app may be open to abuse. If permissions are not defined properly, an app may allow significant security breaches to the device, networked computers, and most critically, back-end databases. The following cybersecurity scenarios represent real life situations that could result in serious consequences, such as a failing grade, expulsion, criminal prosecution, fines, a lifelong ban on tech careers, even jail time.

*Cybersecurity scenario A:* Your partner asks for your password in order to collaborate on a document for the project. Describe:

- Possible unethical behavior
- Consequences of committing an offense
- An ethical solution

*Cybersecurity scenario B:* You accidentally discover a  “**backdoor**” into your teacher’s gradebook and you can see everyone’s grades in the class. Describe:

- Possible unethical behavior
- Consequences of committing an offense
- An ethical solution

*Cybersecurity scenario C:* You are hired as an intern at a high-tech company and a month into the job, a competitor offers to pay you \$50,000 for access into your company’s secure data server. Describe:

- Possible unethical behavior
- Consequences of committing an offense
- An ethical solution

## Part II: Implement

---

- 7 Use the Agile development methodology in developing your app.
- 8 Create Java classes and subclasses as necessary. Your list of objects should be a polymorphic list of items from each category.
- 9 Randomly select a subset of items to find from a larger list of items. Recall the use of the static method `Math.random()`:  

```
Math.random(); // generates a double between 0 and .9999999999999999
(int)(Math.random()*10); // generates an integer number between
0 and 9
(int)(Math.random()*100) + 1; // generates a number between
1 and 100
```
- 10 Reference Unit 1's LetMeTakeASelfie app for using the camera.
- 11 Reference TripTracker for the geolocation features.
- 12 When all objects in the find list have been found with a picture and geolocation, indicate a completed state of the game.

## Part III: Additional Challenge

---

- 13 Create a Backendless back-end service for your GeoQuest so you can stop a game and pick it up later.

# GeoQuest Requirements

The GeoQuest project requires documentation, specific app functionality, and a team presentation.

## Documentation Requirements

1. Problem Statement
2. Brainstorm List
3. Product Backlog
4. Sprint Task List
5. Source code for the product, including at least:
  - a. An original abstract class
  - b. Original subclasses of the abstract class
  - c. A polymorphic list of subclass items
  - d. Javadocs for all original methods
6. Narrative on the three cybersecurity consequences (Step 5 in Problem 3.3.5 GeoQuest)
7. Documentation of any features the team adds that go above and beyond the following functionality

## Functionality Requirements

1. Navigation between `Fragments` and/or `Dialogs`
2. Selection from a randomly generated list of items
3. Photo-captured items
4. Geolocation-captured items
5. A record of all found items
6. A “completed state” of game notification

# Team Presentation Requirements

1. An explanation of the team's contributions of original code to the product solution, as well as the structure of the solution
2. A demonstration of the features of the app that the team created
3. An example of a particularly challenging debugging situation accompanied by an explanation of the process involved to fix the bug
4. An explanation of how the product development process benefited from both integration testing and unit testing
5. A description of the version control system that the team chose, as well as rationale for choosing it
6. A description of the current state of the product with respect to the problem statement, product backlog, and sprint task list
7. A description of next steps for development

# Problem 3.3.5 GeoQuest Rubric

## RESOURCES



**Problem 3.3.5 Rubric**  
Resources available online

## Presentation Criteria

Criteria	4	3	2	1	Total
<b>Explanation of original work</b>	The tools selected for communicating about the project structure allow for an exceptionally clear view of the team's original work.	The tools selected for communicating about the project structure allow for a mostly clear view of the team's original work.	The tools selected for communicating about the project structure allow for a somewhat unclear view of the team's original work.	The tools selected for communicating about the project structure allow for an unclear view of the team's original work.	
	The relationships between the team's original classes, fields, and methods is exceptionally clear.	The relationships between the team's original classes, fields, and methods is mostly clear.	The relationships between the team's original classes, fields, and methods is somewhat unclear.	The relationships between the team's original classes, fields, and methods is unclear.	
	The reason for the creation of each of the team's original classes, fields, and methods is exceptionally clear.	The reason for the creation of each of the team's original classes, fields, and methods is mostly clear.	The reason for the creation of each of the team's original classes, fields, and methods is somewhat unclear.	The reason for the creation of each of the team's original classes, fields, and methods is unclear.	

Criteria	4	3	2	1	Total
	One complex method was exceptionally well explained.	One complex method was mostly well explained.	One complex method was somewhat well explained.	One complex method was not well explained.	

Criteria	4	3	2	1	Total
<b>Product Demonstration</b>	The tools selected for communicating about the project structure allow for an exceptionally clear view of the team's original work.	The tools selected for communicating about the project structure allow for a mostly clear view of the team's original work.	The tools selected for communicating about the project structure allow for a somewhat clear view of the team's original work.	The tools selected for communicating about the project structure allow for an unclear view of the team's original work.	
	Demonstration of the product highlights at least two original features of the product.	N/A	Demonstration of the product highlights one original feature of the product.	N/A	
	Demonstration of the product includes an exceptionally clear, complete explanation of events that trigger functionality.	Demonstration of the product includes a mostly clear explanation of events that trigger functionality.	Demonstration of the product includes a somewhat clear explanation of events that trigger functionality.	Demonstration of the product includes an unclear explanation of events that trigger functionality.	

Criteria	4	3	2	1	Total
	Demonstration of the product includes complete clarification of which event handlers are being triggered.	Demonstration of the product includes mostly complete clarification of which event handlers are being triggered.	Demonstration of the product includes somewhat complete clarification of which event handlers are being triggered.	Demonstration of the product includes incomplete clarification of which event handlers are being triggered.	

Criteria	4	3	2	1	Total
<b>Coding Process Comments</b>	The team discusses a challenging debugging situation that occurred during development. The team...				
	Thoroughly explains how they detected the bug.	Mostly explains how they detected the bug.	Somewhat explains how they detected the bug.	Minimally explains how they detected the bug.	
	Thoroughly explains why they chose the debugging techniques that they did.	Mostly explains why they chose the debugging techniques that they did.	Somewhat explains why they chose the debugging techniques that they did.	Minimally explains why they chose the debugging techniques that they did.	
	Thoroughly explains how they corrected the bug once it was detected.	Mostly explains how they corrected the bug once it was detected.	Somewhat explains how they corrected the bug once it was detected.	Minimally explains how they corrected the bug once it was detected.	



Criteria	4	3	2	1	Total
<b>Development Process Comments</b>	The team thoroughly and clearly explains how product development benefited from both unit testing and integration testing.	The team mostly explains how product development benefited from both unit testing and integration testing.	The team somewhat explains how product development benefited from both unit testing and integration testing.	The team minimally explains how product development benefited from both unit testing and integration testing.	
	The team thoroughly and clearly explains their rationale for the version control system that they chose.	The team mostly explains their rationale for the version control system that they chose.	The team somewhat explains their rationale for the version control system that they chose.	The team minimally explains their rationale for the version control system that they chose.	
	The team thoroughly and clearly explains next steps for development of the product.	The team mostly explains next steps for development of the product.	The team somewhat explains next steps for development of the product.	The team minimally explains next steps for development of the product.	
	All original methods include complete Javadoc documentation.	Most original methods include complete or near complete Javadoc documentation.	Many original methods include complete Javadoc documentation, some may be incomplete.	A minimal number of original methods include Javadoc documentation.	

# Product Criteria

Criteria	4	3	2	1	Total
<b>User Interface</b>	User interface is exceptionally logically arranged.	User interface is mostly logically arranged.	User interface is somewhat logically arranged.	User interface is illogically arranged.	
	Navigation between fragments and/or dialogs is exceptionally easy.	Navigation between fragments and/or dialogs is mostly easy.	Navigation between fragments and/or dialogs is clumsy or difficult.	Navigation between fragments and/or dialogs is non-existent.	

Criteria	4	3	2	1	Total
<b>Functionality</b>	The list of items to find is randomly generated from a larger, list of items.	The list of items to find is generated from a larger list of items.	A list of items to find is present.	A list of items to find is incorrect or causes errors.	
	The list of items to find is composed of polymorphed items.	N/A	The list of items to find is composed of homogenous items.	N/A	

Criteria	4	3	2	1	Total
	The list of found items is presented in a very logical and fluent manner.	The list of found items is presented in somewhat logical and fluent manner.	The list of found items is difficult to find or awkwardly navigable.	The list of found items is incomplete or incorrect.	
	The selected item is captured with a picture and the picture is stored with the item in an object.	The selected item is captured with a picture and the picture is stored.	The selected item is captured with a picture.	The camera is accessed.	
	The selected item is captured with its geolocation and is stored with the item in an object.	The selected item is captured with its geolocation and is stored.	The GeoQuest does not indicate a completed state.	The geolocation feature is incorrect or incomplete.	
	The GeoQuest correctly denotes a completed state.	N/A	The GeoQuest incorrectly denotes a completed state.	N/A	

# Problem 3.3.5 One-Page Rubric

## RESOURCES



**Problem 3.3.5 One-Page Rubric**  
Resources available online

Explanation of original work	Product Demonstration	Coding Process Comments	Development Process	User Interface	Functionality
The tools selected for communicating about the project structure allow for an exceptionally clear view of the team's original work.	The tools selected for communicating about the project structure allow for an exceptionally clear view of the team's original work.	In a debugging situation, the team thoroughly explains how they detected the bug.	The team thoroughly and clearly explains how product development benefited from both unit testing and integration testing.	User interface is exceptionally logically arranged.	The list of items to find is randomly generated from a larger list of items.
					The list of items to find is composed of polymorphic items.
The relationships between the team's original classes, fields, and methods are exceptionally clear.	Demonstration of the product highlights at least two original features of the product.	In a debugging situation, the team thoroughly explains why they chose the debugging techniques that they did.	The team thoroughly and clearly explains their rationale for the version control system that they chose.	Navigation between fragments and/or dialogs is exceptionally easy.	The list of found items is presented in a very logical and fluent manner.
					The selected item is captured with a picture and the picture is stored with the item in an object.

Explanation of original work	Product Demonstration	Coding Process Comments	Development Process	User Interface	Functionality
The reason for creation of all of the team's original classes, fields, and methods is exceptionally clear.	Demonstration of the product includes an exceptionally clear, complete explanation of events that trigger functionality.	In a debugging situation, the team thoroughly explains how they corrected the bug once it was detected.	The team thoroughly and clearly explains next steps for development of the product.	The selected item is captured with its geolocation and is stored with the item in an object.	
One complex method was exceptionally well explained.	Demonstration includes complete clarification of event handlers are being triggered.		All original methods include complete Javadocs.		The GeoQuest correctly denotes a completed state.