Hyperiondev

TASK

Capstone Project III

Visit our website

Copyright © 2021 HyperionDev. All rights reserved.

Introduction

WELCOME TO THE FINAL CAPSTONE TASK!

You've made it to the final Capstone Project for this level! Great job! In this Capstone Project, you will be consolidating all the knowledge that you have gained in this level of the Bootcamp. You will extend and complete the program that you have been working on in the previous Capstone Projects for this level. By the end of this project, this program must be ready to add to your developer portfolio. This project should make it clear to a prospective employer that you can follow a development process to create an object-oriented program that is debugged, tested, refactored and documented program that meets a client's specification!

Get in touch

Connect for support

Remember that with our courses, you're not alone! You can contact an expert code reviewer to get support on any aspect of your course.

The best way to get help is to login to **www.hyperiondev.com/portal** to start a chat with a code reviewer. You can also schedule a call or get support via email.

Our expert code reviewers are happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

Copyright © 2021 HyperionDev. All rights reserved.

DEVELOPER PORTFOLIO

By the end of this project, you will be adding the program that you have been developing throughout this level to your developer portfolio. This project should make it clear to a prospective employer that you can follow a development process to create an object-oriented program that is debugged, tested, refactored and documented and meets a client's specification! Put extra time and effort into this project to make sure that it really showcases the skills you have acquired!

THE TASK AT HAND

You have been asked to create a project management system for a small structural engineering firm called "Poised". Poised does the engineering needed to ensure the structural integrity of various buildings. They want you to create a Java program that they can use to keep track of the many projects on which they work.

Poised stores the following information for each project that they work on:

- Project number.
- Project name.
- What type of building is being designed? E.g. House, apartment block or store, etc.
- The physical address for the project.
- ERF number.
- The total fee being charged for the project.
- The total amount paid to date.
- Deadline for the project.
- The name, telephone number, email address and physical address of the architect for the project.
- The name, telephone number, email address and physical address of the contractor for the project.
- The name, telephone number, email address and physical address of the customer for the project.

Poised wants to be able to use your program to do the following:

- Capture information about new projects. If a project name is not provided when the information is captured, name the project using the surname of the customer. For example, a house being built by Mike Tyson would be called "House Tyson" and an apartment block owned by Jared Goldman would be called "Apartment Goldman".
- Update information about existing projects. Information may need to be adjusted at different stages throughout the lifecycle of a project. For

Copyright © 2021 HyperionDev. All rights reserved.

example, the deadline might change after a meeting with various stakeholders.

- Finalise existing projects. When a project is finalised, the following should happen:
    - An invoice should be generated for the client. This invoice should contain the customer's contact details and the total amount that the customer must still pay. This amount is calculated by subtracting the total amount paid to date from the total fee for the project. If the customer has already paid the full fee, an invoice should not be generated.
    - The project should be marked as "finalised" and the completion date should be added. All the information about the project should be added to a text file called "Completed project".
- See a list of projects that still need to be completed.
- See a list of projects that are past the due date.
- Find and select a project by entering either the project number or project name.

Before you begin

A key focus of this project will be ensuring that your code is correct, well-formatted and readable. In this regard, make sure that you do the following before submitting your work:

1. Make sure that your code is readable. To ensure this, add comments to your code, use descriptive variable names and make good use of whitespace and indentation. See **this style guide** to see how classes and methods should be named and how your program should be formatted.

2. Make sure that your code is as efficient as possible. How you choose to write code to create the solution to the specified problem is up to you. However, make sure that you write your code as efficiently as possible.

3. Make sure that all output that your program provides to the user is easy to read and understand. Labelling all data that you output (whether in text files or to the screen) is essential to make the data your program produces more user-friendly.

Copyright © 2021 HyperionDev. All rights reserved.

Compulsory Task

Follow these steps:

- Design your program to meet the specifications given by the client. Extend the program that you have written in previous Capstone Projects so that this program also:
    - Reads details about existing projects from a text file and uses this information to create a list of project objects.
    - Allows a user to add new objects to this list.
    - Allows a user to select and update or finalise any project on the list.
    - Allows a user to see a list of projects that still need to be completed.
    - Allows a user to see a list of projects that are past the due date.
    - Writes the updated details about the projects to the text file when the program ends.

- Besides meeting the above criteria, you should also do the following:
    - Make sure that your program includes exception handling. Use try-catch blocks wherever appropriate. This should include ensuring that your program handles exceptions related to writing or reading to/from text files and exceptions related to acquiring user input.
    - Make sure that you have completely removed all errors from your code. Take extra care to ensure that logical and runtime errors have been detected and removed.
    - Make sure that your code has been adequately refactored.
    - Make sure that your code is adequately documented. Adhere to the style guide found **here**.
    - Use Javadoc to generate API documentation from documentation comments for your program.

- Submit your fully debugged and refactored code and the documentation for your project to your mentor.

- After receiving feedback from your mentor and improving your code based on this feedback, add your program to Github.

Copyright © 2021 HyperionDev. All rights reserved.

Completed the task(s)?

Ask an expert to review your work!

Review work

Rate us

Share your thoughts

HyperionDev strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved or think we've done a good job?

Click here to share your thoughts anonymously.

Copyright © 2021 HyperionDev. All rights reserved.