

# OOP PROJECT

**Due Date: February 8th, 2pm**

The OOP Project is an opportunity for bootcamp students to

- Begin to pull together a number of different things they've learned so far, including classes and polymorphism, user input, arrays and Lists, and exceptions;
- Work on a larger and more complicated case study than lab exercises as a prelude to the Final Project;
- Collaborate with teammates on a software project, experiencing the need for proper object-oriented design, documentation, and version control.

Every student will participate in this project in an assigned group. The majority of 3 days will be committed to the project (although outside work may still be necessary) and there will be multiple check-ins during those days to make sure groups are on-track.

The entire group will work together on one of the three projects. Take a little time to decide which one you want to tackle, but once you get started stick to that project—no turning back!

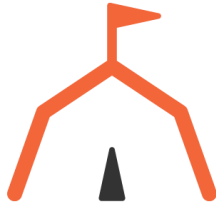
Possible projects:

- Point-of-Sale Terminal (a cash register/ordering terminal for someplace like a store, coffee shop, or fast-food restaurant)
- Library System
- Fitness Center

See the following pages for more information on each project. Please recognize that the descriptions are minimum versions; it's hoped each group will go beyond these requirements and incorporate features of interest to them.

**By the end of the day, email the following to [justin.jones@grandcircus.co](mailto:justin.jones@grandcircus.co) and [kristen.harrell@grandcircus.co](mailto:kristen.harrell@grandcircus.co)**

- **Your group's Github Repo**
- **Your group's Chosen Project**



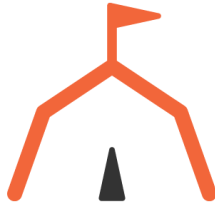
## Point of Sale TERMINAL

Write a cash register or self-service terminal for some kind of retail location. Obvious choices include a small store, a coffee shop, or a fast food restaurant.

- Your solution must include some kind of a product class with a name, category, description, and price for each item.
- 12 items minimum; stored in a list.
- Present a menu to the user and let them choose an item (by number or letter).
  - Allow the user to choose a quantity for the item ordered.
  - Give the user a line total (item price \* quantity).
- Either through the menu or a separate question, allow them to re-display the menu and to complete the purchase.
- Give the subtotal, sales tax, and grand total. (Remember rounding issues the Math library will be handy!)
- Ask for payment type—cash, credit, or check
- For cash, ask for amount tendered and provide change.
- For check, get the check number.
- For credit, get the credit card number, expiration, and CVV.
- At the end, display a receipt with all items ordered, subtotal, grand total, and appropriate payment info.
- Return to the original menu for a new order. (Hint: you'll want an array or List to keep track of what's been ordered!)

Optional enhancements:

- (Moderate) Store your list of products in a text file and then include an option to add to the product list, which then outputs to the product file.
- (Buff) Do a push up every time you get an exception or error while running your code



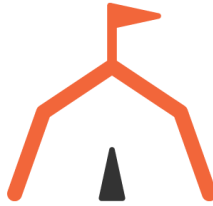
## LIBRARY TERMINAL

Write a console program which allows a user to search a library catalog and reserve books.

- Your solution must include some kind of a book class with a title, author, status, and due date if checked out.
  - Status should be On Shelf or Checked Out (or other statuses you can imagine).
- 12 items minimum; All stored in a list.
- Allow the user to:
  - Display the entire list of books. Format it nicely.
  - Search for a book by author.
  - Search for a book by title keyword.
  - Select a book from the list to check out.
    - If it's already checked out, let them know.
    - If not, check it out to them and set the due date to 2 weeks from today.
  - Return a book. (You can decide how that looks/what questions it asks.)

Optional enhancements:

- (Moderate) When the user quits, save the current library book list (including due dates and statuses) to the text file so the next time the program runs, it remembers.
- (Julius Caesar) Burn down the library of Alexandria and set human Civilization back by a few hundred years.



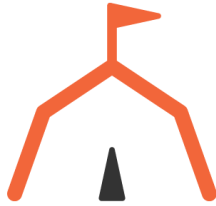
## FITNESS CENTER

Write a console application for a fitness center to help manage members and membership options. At a minimum, this program should include:

- A class to hold basic details about Members (this class should eventually have at least 2 child classes) and hold the following details at a minimum:
  - id, name
  - an abstract method **void CheckIn(Club club)**
- A minimum of two child classes that represent a Single Club Member and Multi-Club Members (these members can visit various locations using the same membership). The classes should have the following:
  - Single Club Members: a variable that assigns them to a club. The CheckIn method throws an exception if it's not their club.
  - Multi-Club Members: a variable that stores their membership points. The CheckIn method adds to their membership points.
- A Club class that holds basic details about each fitness club, including at minimum:
  - name, address
- Allow users to:
  - Add members (both kinds), remove members or display member information.
  - Check a particular member in at a particular club. (Call the **CheckIn** method). Display a friendly error message if there is an exception. Don't let it crash the program.
  - Select a member and generate a bill of fees. Include membership points for Multi-Club Members.
- A main class which takes input from the user:
  - Asks a user if they want to select a club
  - Added members should be given the option to select from at least 4 fitness center locations or have the option to be a multi-club member.

Optional enhancements:

- (Easy/Medium) Allow new members to receive discounts if they sign up during certain time periods, explore the DateTime library for help with date and time.
- (Medium) Store clubs and members in text files.
- (Hard) Out Pizza the hut



## PRESENTATIONS GUIDELINES

At the end of the project, we will present our project build to both staff and students. This doesn't need to be anything fancy, just a quick run through of your code.

Here's the rules for presentations each group should:

- Take 10 minutes maximum to present (including time for questions)
- Spend most of your time showing the project running
- Show off the most important objects and methods in the code (maybe one per team member)
- Don't go line by line, quick few sentence summaries are more than enough detail
- Share the presentation between group members roughly equally, everyone should speak
- Focus on the back-end, meaning transforming and manipulating data. We care about how the pizza is made, not how it looks. (EG: we check out a book update the due date in the book, and then update it in our File)
- Talk about the challenges you faced and how you worked through them as a group (Employers looove hearing about this exact kinda stuff)