|  |  |  |
| --- | --- | --- |
|  | OOP Exercise  Video Games |  |

This exercise is open-book, open-note, and open-internet; however, you must not collaborate outside your group. You and your partner(s) are to develop a basic Object-Oriented Model for your selected topic. You will have **30 minutes** to complete your model. Following this time, each group will present (2-3 minutes) their model to the class, highlighting key design choices providing insight into your thought process. An example to help guide you has been provided on the reverse of this page.

**Your topic is Video Games**



Image Credit [Evan Amos](https://commons.wikimedia.org/wiki/User:Evan-Amos)

Some helpful hints:

1. Consider what your **base class** or root of your diagram will be. This is the most **abstract** piece of your diagram. In this case, you might consider simply “CGame” or “CVideoGame” (the initial “C” helps the programmer distinguish this as a Class name).
2. Consider which qualities are common among all different types of the base class. List those qualities underneath your base class. This is a type of **encapsulation**. In this case, you might choose qualities like whether or not the game is single-player, online or LAN, etc.
3. Consider what will **inherit** from that base class and design a subclass for each distinction. In this case, you might wish to design a subclass for Role-Playing Games, Military Games, Strategy Games, etc. You might also follow a different line and choose to align your model with First Person games, Third Person games, etc. It is your choice!
4. Now consider which qualities are important to include in each of your subclasses. If you begin to notice something that is common among all subclasses, where do you think that quality might belong?
5. Don’t stop here! Consider other properties of your topic and either add further divisions or add further independent classes! In this case, you might add a class for massive multiple online role-playing games and a subclass for different games of that genre. You can also create some classes that other classes might **have**! For example, all games **have** players…

**Example: Coast Guard Employees**

* The base class in this case is a Coast Guard Employee.
  + Each Coast Guard Employee must have a pay, a current location, a characteristic to determine whether they are retired or not, and a counter for the number of years they’ve served in the USCG.
* Following that, a distinction can be made between Service Members and Civilian Employees.
  + A Service Member has unique properties like a characterization of whether they are active duty or not. A Civilian will have a job description unique from what a Service Member might have.
* Underneath Service Members, we may further divide Enlisted Members and Officers. Under civilians, we can further divide contractors and federal employees.
  + We ought to then list their properties as well.

