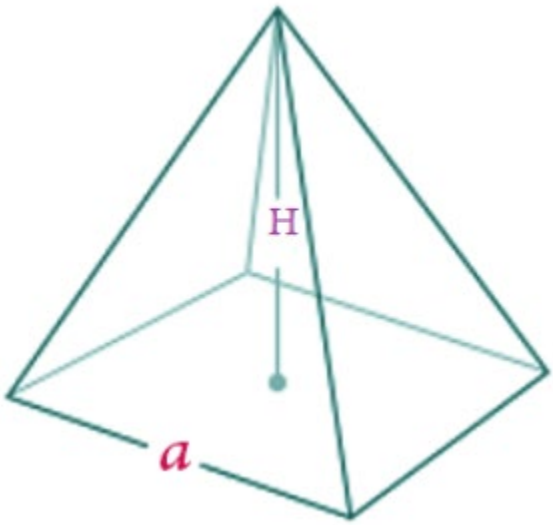
Introduction to Java
CS9053 Section I
Thursday 6:00 PM – 8:30 PM
Prof. Dean Christakos
Feb 8th, 2024
Due: Feb 15th, 2024 11:59 PM

Part I – Creating Objects

1. Square pyramid: In the lecture you have seen the creation of a circle. Here you are going to create a square pyramid. A square pyramid has a square base. It has base, *a*, and a height, *H*.



The Volume of a Pyramid is given by $\frac{1}{3}a^2H$
l is known at the "slant" and is given by $\sqrt{h^2 + r^2}$
The surface area of a square pyramid is given by $a^2 + 2a\sqrt{\frac{a^2}{4} + H^2}$

You will create a class SquarePyramid using the following UML:

SquarePyramid
-nextId: int -id: int -side: double -height: double
+ SquarePyramid() + SquarePyramid(side: double, height: double) +getSide(): double +setSide(side: double): void +getHeight(): double +setHeight(height: double): void +getVolume(): double +getSurfaceArea():double +getID(): int

In standard UML parlance, "+" indicates that a field or method is public and "-" indicates that a field or method is private. An underlined field or method indicates it is static.

Every time you create a new SquarePyramid instance, it should have a new sequential id, based on the value of `nextId`, which should be incremented every time you create a new SquarePyramid instance.

2. Objects and Arrays of Objects

Your objective is to develop the **`Hero`** and **`Party`** class and to use their methods. The classes are described below to guide you.

Hero

The **`Hero`** class contains data related to a hero in a video game. Implement the class according to the following **UML** diagram:

Hero

Data fields	
-name : String	Name of the Hero
-role: String	Role of the Hero
-level: int	Contains the level of the Hero
-experience: int	Contains the experience of the Hero
-MAX_LEVEL: int	Constant static int containing the max level of 100.
-ROLES: String[]	The possible roles: “Warrior”, “Priest”, “Wizard”, “Thief” (ie, a static, pre-defined array of Strings)
Methods	
+Hero (name: String)	Set the name to given value, set role to `"Unassigned"`, set level to 1, and experience to 0.
+setRole(role: String)	Set the role of the Hero. If the role is not one of the allowed roles, print `"Invalid role"` and set the role to `"Unassigned"`.
+getters for each non-static field	Get the name, role, level, or experience.
+expToLevelUp() :int	Return the amount of experience necessary to advance to the next level. The formula is $level^2$. For example, at level 6, this should return $6^2 = 36$. This is a helper method for the gainExperience method.
+gainExperience(experience: int)	Increase the experience of the Hero. If the experience reaches or passes the maximum for the Hero’s current level, increase the level by 1 by “consuming” the necessary experience. For example, at level 4 with 20 experience, the Hero should level up to 5 with 4 experience remaining. When a Hero levels up, print “<name> is now level <level>!” The max level is 100. Note: If enough experience is gained at once, the Hero will level up more than once. For example, a level 3 hero receiving 75 experience will level up to 6 with (75 – 9 – 16 – 25) = 25 experience remaining.
+toString() :String	Return a String containing the information about the hero:

	Template: "<name> the <role> is level <level> with <experience> experience." Example: "Hagrid the Wizard is level 55 with 120 experience."

Party

The **`Party`** class contains an array of **`Hero`**'s that are in the party.
Implement the class according to the following UML diagram:

Party

Data fields	
-heroes: Hero[3]	Array of Hero's in the current party.
Methods	
+Party()	Initialize the Hero array to hold 3 Hero's. Initialize each element to null.
+addHero(hero: Hero, index: int)	Add a Hero to the current party at the given index. If there is an existing Hero at the index, replace it. If the hero is already in the party, print "`<name> is already in the party.`"
+removeHero(index: int)	Remove the Hero from the given index (to "remove" from the array, set the array element to null and shift any other elements to the left, so that the only nulls are at the end of the array).
+getHero(index: int) :Hero	Return the Hero of the desired index.
+gainExperience(experience: int)	Prints "`The party gained <experience> experience.`" Increase the experience of all the Hero's in the Party. Hint: Use each Hero's gainExperience method.

+toString() :String	Return a String containing the information about the hero.
	Template:
	"`Party:`
	`<name> the <role> is level <level> with <experience> experience.`
	`<name> the <role> is level <level> with <experience> experience.`
	`<name> the <role> is level <level> with <experience> experience.`
	"
	Example:
	"`Party:`
	`Thor the Warrior is level 10 with 4 experience.`

	`Groot the Thief is level 20 with 19 experience.`
	`Doctor Strange the Wizard is level 5 with 1 experience."`
	
	Note: Only print assigned party members (Do not print anything if any heroes elements are null).
	Hint: Use each Hero's toString method.

After implementing this, create three heroes with names and roles of your choice, add them to the party, and give the party 100 experience points. Then print out the party by calling the `toString()` method for the party and passing it to `System.out.println()`