CS9053 Introduction to Java

Midterm Coding Section

Professor Dean Christakos

February 27th, 2025

Due Date: March 6th, 2025 11:59PM

**Part I:**

You are going to take this class hierarchy structure and implement it.

**1. Abstract Base Class**

* **Lifeform (abstract)**
  + Common attributes: name, scientificName, lifespan
  + Abstract method: describe()
  + Concrete method: getDetails()

**2. Kingdom Level**

* **Animal (extends Lifeform)**
  + Adds isMobile, dietType
* **Plant (extends Lifeform)**
  + Adds photosynthesisEfficiency, growthRate

**3. Phylum Level**

* **Chordate (extends Animal)**
  + Adds vertebrate, nervousSystemComplexity
* **Arthropod (extends Animal)**
  + Adds exoskeleton, legCount

**4. Class Level**

* **Mammal (extends Chordate)**
  + Adds warmBlooded, gestationPeriod
* **Bird (extends Chordate)**
  + Adds wingSpan, canFly
* **Insect (extends Arthropod)**
  + Adds wingCount, metamorphosisStage

**5. Order Level**

* **Carnivore (extends Mammal)**
  + Adds huntingStrategy, biteForce
* **Herbivore (extends Mammal)**
  + Adds ruminationTime, herdBehavior
* **Raptor (extends Bird)**
  + Adds talonStrength, visionRange
* **Passerine (extends Bird)**
  + Adds songComplexity, beakType

**6. Family Level**

* **Felidae (extends Carnivore)**
  + Adds climbingAbility, roarVolume
* **Canidae (extends Carnivore)**
  + Adds packInstinct, olfactorySensitivity
* **Accipitridae (extends Raptor)**
  + Adds predationSuccessRate, wingLoading

**7. Genus Level**

* **Panthera (extends Felidae)**
  + Adds territorySize, maneSize
* **Canis (extends Canidae)**
  + Adds howlLoudness, endurance

**8. Species Level**

* **Lion (extends Panthera)**
  + Adds prideSize, maneColor
* **Wolf (extends Canis)**
  + Adds packLeadershipScore, howlFrequency

Assignment:

1. Create the inheritance hierarchy
2. Implement the constructors ensuring proper superclass constructor chaining. You can look up the “scientificName” for each in Wikipedia. They’re relatively simple.
3. Override methods appropriately, like describe(), which will have increasingly specific descriptions in the subclasses
4. Include abstract methods where applicable (for example, Lifeform.describe())
5. Create 5 objects of different classes.
6. Create an ArrayList<Lifeform> and put those 5 objects in it. Use a lambda expression to sort them by lifespan and then another lambda expression to sort them by name

**Part II:**

Create an LogEntry class with the fields date, severity, module, and message. It should implement toString() which has a format of your choice but should include all the field data.

Create an abstract LogProcessor class with the abstract method processLog(String inFile, String outFile). Create two concrete classes, WarnLogProcessor and ErrorLogProcessor.

The WarnLogProcessor reads the log file and ingests WARN messages, while the ErrorLogProcessor reads the log file and ingests ERROR messages.

I/O Exceptions should be caught

processLog() reads in the log data, stores each item in a LogEntry and then writes the log entry to an output file. When you call it, the output file should be specific to the class—one file for Warn messages, one file for Error messages.

You will read in the file Hadoop\_2k.log and output to two files, one for Warn entries and one for Error entries. WARN/ERROR/INFO/etc are indicators of severity. The date appears in the beginning of the entry. The module appears in square brackets after the severity indicator. The message comes after the module. Example:

2015-10-18 18:06:11,935 WARN [LeaseRenewer:msrabi@msra-sa-41:9000] org.apache.hadoop.hdfs.LeaseRenewer: Failed to renew lease for [DFSClient\_NONMAPREDUCE\_1537864556\_1] for 74 seconds. Will retry shortly ...

Hints:

Methods like String.split(), String.contains(), java.util.regex.Matcher, and java.util.regex.Pattern could be your friends for this. Look them up and see how they may be useful for this. Dates can be parsed with SimpleDateFormat is good for parsing strings into Date objects and serializing Date objects into formatted Strings.