Question 1 – 5

I put all errors here. Total 11 errors. You can match Error # with those in ConceptualUnderstanding class in final-exam-solution project, and line # in the same class in final-exam-problem project.

Error 1 – Line 4: import edu.nyu.cs.finale.Concept

**missing ';' at end of the statement**

**Solution:** import edu.nyu.cs.finale.Concept;

Error 2 – Line 9: private String accumulatedWisdom[] = new String[];

**missing size of array**

**Solution: It can fix by adding size as = new String[3];**

**or with values directly as = new String[] {"A","B","C"};**

Error 3 – Line 14: this.concepts = {

**"this" is invalid in static method. It can only use in instance method**

**it can only be used in instance method**

Error 4 – Line 14: this.concepts = {

**missing new operator before Concept[]**

**Solution 3-4: concepts = new Concept[]** {

Error 5 – Line 21: if (this.comprehend(details)) {

**"this" is invalid in static method**

**Solution: if (comprehend(details)) {**

Error 6 – Line 32-34: catch (GeneralizedExistentialException e) {

System.out.println(“We must carry on.”);

}

**comprehend method() does not throw GeneralizedExistentialException. So, catch statement is invalid.**

**Solution: remove the catch statement.**

Error 7 – Line 39: public static comprehend(String details) {

**missing return boolean**

Error 8 – Line 39: public static comprehend(String details) {

**missing throws LackOfConceptualClarityException because it throws out by the method when if condition meet**

**Solution 7-8: public static boolean comprehend(String details) throws LackOfConceptualClarityException {**

Error 9 – Line 40: if (details.indexOf(“monism”) > 0 || details.length > 500) {

**details is a String which has no length field. Should use length() method.**

**Solution: if (details.indexOf("monism") > 0 || details.length() > 500) {**

Error 10 – Line 42: return false;

**Statements after throw statement are not reachable.**

**Solution: remove it.**

Error11 – Line 45-46: this.accumulatedWisdom[this.wisdomCounter] = details;

this.wisdomCounter++;

**“this” is invalid in static method. Remove them.**

**Solution: accumulatedWisdom[wisdomCounter] = details;**

**wisdomCounter++;**

**Note: Because comprehend is a static method, accumulatedWisdom[] and wisdomCounter must be static variables.**

**So their definition must be changed to static. See line 14 and 15 in solution class.**

Two warnings

Warning 1 – Line 26: continue;

It does not cause any problem but doing nothing. Because process will always continue even without this statement.

Warning 2 – Line 29-31: catch (LackOfConceptualClarityException e) {

System.out.println(“Sorry, we’re lost.”);

}

It depends on which package LackOfConceptualClarityException is in. If it is in the same package as ConceptualUnderstanding class, it is OK. If it is not, “import ….. LackOfConceptualClarityException” statement must be added.

Question 6: Explain which part of the code exhibits polymorphism, and how so

**Code*: concepts = new Concept[] {***

***new Being(),***

***new Nothingness()***

***};***

**Shows Being and Nothingness are Concept. So they can be handled as Concept, which is polymorphism for.**

Question 7: Can you say for sure whether the Concept class has any child classes? Explain why or why not?

**Yes. They Being and Nothingness. The code in Question 6 shows Being and Nothingness are Concept and Concept is a class, so they must be derived from Concept.**

Question 8: Dose this code suggest that the Concept class’s getSummary() method may throw a LackOfClarityException? Explain why or why not.

**getSummary() does not throw LackOfClarityException. If it throws, then the code will have complier error. Because there is not try block to catch LackOfClarityException around getSummary() method call.**

Question 9: Explain abstraction and indicate where it is exhibited in this code, if at all.

**Abstraction is one of the 3 pillars of Object Oriented Programming(OOP). It literally means to perceive an entity in a system or context from a particular perspective. We take out unnecessary details and only focus on aspects that are necessary to that context or system under consideration.**

**In this code, Concept is an abstraction from Being and Nothingness, which has a common feature summary, modeled by getSummary() method.**

Question 10: Explain how ArrayList differ from arrays and how they can be used to improve certain parts of this code.

**ArrayList has no predetermined size limit but array has. Replace String[] accumulatedWisdom with List<String> will allow code to handle unpredetermined calls to comprehend() method.**

Question 11:

**See Point class in edu.nyu.cs.finale.drawLine package in final-exam-solution project.**

Question 12: According to how object-oriented programing ideology is implemented in Java, should the two instance variables in the Point class be declared as public, protected, or private? Justify your answer.

**I am not sure about it. I see the similar question somewhere else. You better ask or discuss it with your class mates. If professor talked about it. Use his answer.**

Question 13:

**See Line class in edu.nyu.cs.finale.drawLine package in final-exam-solution project.**

Question 14:

**See RecursiveDrawLine class in edu.nyu.cs.finale.drawLine package in final-exam-solution project.**

Question 15: How many Line objects total will b instantiated when the code created in the previous question executes.

**1+2+4 = 7**

Question 16: Each element in a two-dimensional array must have the same number of sub-elements

b. False

Question 17: a class can implements more than one interface

1. True

Question 18 Constructors in a class definition.. (select all that apply)

1. Must accept no parameters
2. Must have no return value
3. Must be overloaded at least once.
4. Must not be declared as public or private

Question 19: The keywork “super” is a reference to a class’s … (select all that apply)

1. No-args constructor
2. Parent class
3. Instantiated object
4. Package
5. Access modifier (this refer to public, protected, private)

Question 20: To adhere to the concept of abstraction, where possible, data fields in a Java program should be made…

1. Public
2. Static
3. Private
4. Void
5. -None of the above

Same as question 12. I am not sure. Definite not b. d. and e. lecture note <https://knowledge.kitchen/Inheritance_and_other_more_advanced_object-oriented_programming_concepts> sounds like private.

Question 21:

1. “Killer”

Question 22:

1. “Cuddles”

**Note: Question 21 and 22 test your understanding about how Java pass data through function. In Question 21, *cuddles* is pass as Object to method *doggify*(Cat c, String name). In Question 22, *cuddles* is pass as String to method *doggify*(String s, String name). In both questions, *doggify* changes the first input parameter, which is *cuddles* defined outside of it (in main method), using the second input parameter. Then ask you what happens to the outside variable *cuddles* in main function. The trick is *doggify* can change the outside variable *cuddles* when it passes in as Object as in question 21. But *doggify* cannot change the outside variable *cuddles* when it passes in as String as in Question 22. Simply say, function cannot modify outside variable if it is a prime type or String, but function can modify outside variable if it is Object, include array and List. String is very special. It is not prime type. But it behaves as prime when passing to function as input parameter.**

See “How Java create, compare and pass data through function” for more information.

Question 23: The == operator performs a comparison …

1. By “reference”
2. By value

See “How Java create, compare and pass data through function” for more information.

Question 24: The special keyword, this, when used within a static method of a class refers to

1. The current class.
2. The current object.
3. The parent object.
4. -All of the above.
5. -None of the above

Question 25:

See Exam class in edu.nyu.cs.finale.exam package in final-exam-solution project

Question 26:

See ComputerSicenceCourse and Student classes in edu.nyu.cs.finale.course package in final-exam-solution project