

Answer the following questions using the AP Computer Science A Quick Reference (possibly found at http://apcentral.collegeboard.com/apc/public/repository/ap_comp_sci_a_quick_reference.pdf but also linked from our Java class home page)

- 1) What is `Integer.MIN_VALUE` according to the comments?
- 2) In the following code, roughly how many loop iterations would occur with this loop?

```
for (int i = 0; i > Integer.MIN_VALUE; i++)  
{  
    // stuff happens here  
}
```

- 3) List the 3 methods in the `String` class.
- 4) Explain the 3 possible values that the `compareTo` method returns.
- 5) List 3 methods of the `ArrayList` class (or the `List` interface).
- 6) What does the `Math.random()` method return?
- 7) Explain why the `add` method is listed twice under the `List` interface.
- 8) What is the return type of the `toString` method?

Answer the following questions using the AP Computer Science A Quick Reference (possibly found at http://apcentral.collegeboard.com/apc/public/repository/ap_comp_sci_a_quick_reference.pdf but also linked from our Java class home page)

- 9) If you have an object's location, which **Location** method would you use to determine the direction that another location is in?
- 10) If you have an object's location, which **Location** method would you use to get the location next to you in a specific direction?
- 11) What is the actual numerical value of **HALF_LEFT**?
- 12) What does the **Location.equals()** method do?
- 13) What method would you use to determine how many rows a grid has?
- 14) What method returns an ArrayList with all the occupied locations in a grid?
- 15) Which **Grid** method returns an ArrayList of all the empty locations which are adjacent to a particular location?
- 16) Which **Grid** method gives you the object (actor, bug, etc.) that is sitting at the input location?
- 17) Look at the **Actor** class constructor. When a new actor is created, what is his color and direction?
- 18) What method do you use to set an **Actor's** direction?
- 19) Which class does the **Flower** class extend?
- 20) When a new **Bug** is created (and no color is assigned), what color is he?
- 21) Does a **Bug** have his own **act** method or does he inherit it from **Actor**?
- 22) How does a **Bug** get hold of and name the **Grid** object that he's in? (Write the line of Java code that does this from the Bug class.)
- 23) How does a **Bug** get hold of and name the **Location** object that he's sitting in? (Write the line of Java code that does this from the Bug class.)
- 24) When the **Bug** moves, how does he leave a flower behind him? (Write the 2 lines of Java code that do this.)
- 25) How does a **Bug** check to see if the space in front of him has a flower in it? (Write the line of Java code that does this.)
- 26) Which spaces does a **Bug** check to see if he "canMove"?

- 27) What class does **Critter** extend?
- 28) List the 5 methods that are called in the **act** method in **Critter**.
- 29) What does the **Critter** do to the actors around him?
- 30) Find a line of Java code in **Critter** where a random number is computed.
- 31) What is the postcondition of **getActors** in **Critter**? (Look in the comments above it.)
- 32) What 2 methods does **ChameleonCritter** override?
- 33) Who is **ChameleonCritter**'s "grandparent" (its parent's parent)?
- 34) How does a **Critter** determine if there is a **Rock** or another **Critter** in a location he's going to "process"? (Write the line of Java code.)
- 35) What happens if there is something that is not a **Rock** or a **Critter** in that location?
- 36) Write the line of Java code that makes the **ChameleonCritter** change his color.