|  |  |  |
| --- | --- | --- |
| **APCS Exposure Java** | **Exercises 11.01-06** | **Date:** |
| **Name:** | | **Period:** |

1. List 2 differences between a *Java static array* and the *dynamic* **ArrayList**.

2. What Java class is used for a *dynamic array*?

3. With an **ArrayList** object the quantity of elements can be altered on the fly during program execution, which is officially known as \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ .

4. Where does the **add** method, *add* data in an **ArrayList**?

5. Look at program ***Java1101.java***. What piece of information is not necessary to create an **ArrayList** object that was necessary to create a *Java static array*?

6. The contents of an **ArrayList** can be display by just placing the **ArrayList** indentifier in a **System.out.println** statement (as is shown in program ***Java1307.java***). What happens if you try this with a *Java static array*?

7. What stores the number of elements in a *Java static array*?

8. What method returns the number of elements in an **ArrayList**?

9. What happens if you attempt to access an array element at a location that does not exist?

10. What **ArrayList** method lets you access individual elements in the **ArrayList**?

11. What is the first index in an **ArrayList**?

12. What method is used to alter an item in an **ArrayList**?

13. Refer to the previous question. Explain the 2 parameters of this method.

14. What method will delete an item from an **ArrayList**?

15. Refer to the previous question. What does this method’s one parameter signify?

16. Which **ArrayList** method is *overloaded*?

17. What is the difference between the *one-parameter* **add** methodand the *two-parameter* **add** method?

18. The *two-parameter* **add** method seems to have the same parameters as the **set** method.

What is the difference between these 2 methods?

19. Since **ArrayList** has all of these useful methods, why do we bother with *Java static arrays*?

20. The **Integer** class is a \_\_\_\_\_\_\_\_\_\_\_\_\_ class, which stores **int** values in an object.

21. *Java static arrays* can store both primitive data types and objects. What about **ArrayList**?

22. Can **Integer** objects be *added*?

23. What are the *wrapper* classes for **double** and **boolean**?

24. What does an object store?

25. Look at programs ***Java1108.java*** and ***Java1109.java***. The first program does not compile.

What was added to the second program that allows it to compile?

26. Refer to the previous question. Program ***Java1110.java*** has an even better solution to the problem.

What is it called?

27. Look at program ***Java1112.java***. How does this program violate the very definition of an array?

28. Look at program ***Java1113.java***. What does this program use that prevents the violation of the previous question?

29. Convert this **for** loop to a **for..each** loop. Assume **schools** is an **ArrayList** of **String**.

**for (int x = 0; x < schools.size; x++)**

**System.out.println(schools.get(x));**

30. Look at program ***Java1115.java***. Explain how this program created a 2D **ArrayList**.

31. Can a nested **for..each** loop be used to traverse a 2D **ArrayList**?

32. Why do we still use the original **for** loops if we have the new **for..each** loops?