|  |  |  |
| --- | --- | --- |
| **APCS Exposure Java** | **Exercises 10.01-05** | **Date:** |
| **Name:** | | **Period:** |

1. How many values can be stored by a simple data type?

2. What is a *data structure*?

3. Any data type that can store more than one value is a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

4. What is another word for an *array*?

5. *Arrays* first became popular with what programming language?

6. What is an *array*?

7. A one-dimensional array is frequently also called a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

8. A two-dimensional array is frequently also called a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

9. If you need to store several pieces of data for many people, and you create one array for their names, another array for their addresses, another for their birthdates, another for their social security numbers, etc. What kind of arrays have you created?

10. *Records* first became popular with what programming language?

11. What is a *record*?

12. What is a *file*?

13. What does the *file* data structure allow?

14. What is a stack?

15. What does *LIFO* stand for?

16. What is the *Improved Data Structure Definition*?

17. What is the *Improved Array Definition*?

18. What do an *array* and a *stack* have in common?

19. Refer to the previous question.

If they have this in common, why are they considered different data structures?

20. What is meant by a *contiguous* block of memory?

21. Write the Java code to declare an **int** array called *ages*?

22. Refer to the previous question. Write the Java code that will allocate memory for 30 *ages*?

23. Java arrays indicate individual elements with an \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ inside two brackets.

24. What data type will the array index always be?

25. What is the first array index?

26. Write the Java code that will both declare and allocate memory for an array of 40 **String**s called *names* in one single programming statement.

27. If the individual elements of an **int** array are not initialized, what is stored in them by default?

28. Write the Java code that will both declare and allocate memory for an initialized array of **double** values called **bob**. The initialized values will be **1.1**, **2.2**, **3.3**, **4.4**, and **5.5**.

29. Refer to the previous question. In this situation, does the size of the array need to be specified?

30. What does **length** do when used with a *Java Static Array*?

31. Refer to the previous question. Is **length** a method?

32. Can the **length** field be changed?

33. What is the difference between a *static array* and a *dynamic array*?

34. When generating random integers, what is the purpose of the *seed*?

35. Look at program ***Java1008.java***. How would you change the code if you were using **Math.random**?

36. Look at program ***Java1009.java***. This program displays random sentences. How is that possible?

37. Whether you have an array of integers, an array of real numbers, or an array of strings, what data type will the *index* always be?

38. What is the difference between the *old* **for** loop and the *new enhanced* **for** loop?

39. Does the *new enhanced* **for** loop only work with **String** objects?

40. Does the *new enhanced* **for** loop only work with *primitive* data types?

41. Look at programs ***Java1011.java*** and ***Java1012.java***. How was the second program changed?

Why does this still work?

42. The *enhanced* **for** loop is called the \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ loop.

43. Does the *enhanced* **for** loop replace the older **for** loop? Explain why.