**Guided Learning Plan for ArrayList**

**Class\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Lesson 1 ——Introduction to ArrayList**

|  |  |  |  |
| --- | --- | --- | --- |
| **Topic** | **details** | | **Essential Knowledge** |
| What is ArrayList? What does Arraylist used for? | In the last unit, we learned about arrays to hold collections of related data. But arrays have **limitations**. The **size of an array is established** at the time of creation and cannot be changed. **What if you don’t know how big the collection of data will be?** What if you want to add and remove items from the collection and change the size of the collection while the program is running?  For example, if you wanted to represent a shopping list, you might add to the list throughout the week and remove things from the list while you are shopping. You probably would not know how many items will be on the list at the beginning of the week.  Luckily, Java has a class called **ArrayList** which is a **re-sizable array**.   * An ArrayList has an underlying array that **grows or shrinks** as needed. You can use ArrayList instead of arrays whenever you don’t know the size of the array you need or you know that you will add and remove items and may need to change the array’s size dynamically during run time. * An ArrayList is **mutable**, meaning it can change during runtime by **adding and removing** objects from it.   Quiz:  Which of the following is a reason to use an ArrayList instead of an array?  A. A list will always use less memory than an array. B. A list can store objects, but arrays can only store primitive types. C. A list has faster access to the last element than an array. D. A list resizes itself as necessary as items are added, but an array does not. | | |
| Compare with Array |  | | |
| Import package | The **ArrayList class** is in the java.util package.  Import statements have to be the first code in a Java source file. An import statement tells Java which class you mean when you use a short name (like ArrayList). It tells Java where to find the definition of that class.  import java.util.ArrayList; // import just the ArrayList class  import java.util.\*; // import everything in package including ArrayList | | Note  Don’t worry about adding import statements on the AP CS A exam. Any that you need will be provided for you. |
| Declaring and Creating ArrayLists | 1.To **declare** a ArrayList use:  **ArrayList<Type> name**  Change the Type to be whatever type of objects you want to store in the ArrayList, for example String as shown in the code below.  (If you leave off the <Type> it will default to Object.)  2.To actually **create** a ArrayList use  **new ArrayList<Type>()**.  (If you leave off the <Type> it will default to Object.) | | Note   1. ArrayLists can **only hold** **objects** like String and the wrapper classes Integer and Double. They cannot hold primitive types like int, double, etc. 2. You don’t have to specify the generic type <Type>, since it will default to Object, but it is good practice to specify it to restrict what to allow in your ArrayList. 3. Using a type ArrayList<Type> is preferred over just using ArrayList because it allows the compiler to find errors that would otherwise be missed until run-time. 4. Declaring a ArrayList **doesn’t** actually create a ArrayList. It only creates a variable that can refer to a ArrayList. 5. **If you leave off the <Type> it will default to Object.** |
| **Syntax:**  // ArrayList<Type> name = new ArrayList<Type>();  // An ArrayList of Strings:  ArrayList<String> shoppingList = new ArrayList<String>(); | | |
| Exercise | Quiz:  Which of the following is the correct way to create an ArrayList of integers?  A. ArrayList[int] numbers = new ArrayList(); B. ArrayList<String> numbers = new ArrayList(); C. ArrayList<int> numbers = new ArrayList<int>(); D. ArrayList<Integer> numbers = new ArrayList<Integer>(); | | Note:  All primitive types must be wrapped in objects before they are added to an ArrayList. |
| 1、Write the declaration and creation for an ArrayList of integer named numList:  2、Write the declaration and creation for an ArrayList of double named numList:  3、Write the declaration and creation for an ArrayList of boolean named boolList:  4、Write the declaration and creation for an ArrayList of Student named StudentList:(Student is a self-defined class)  5、Write the declaration and creation for an ArrayList named years that can be used to store elements both of type Integer and of type String. | | |
| How to initialize the ArrayList?  How to add values to ArrayList? | ArrayList<String> shoppingList = new ArrayList<String>()  shoppingList .add(“carrots”);  shoppingList.add("bread");  System.out.println(shoppingList);  //result:[carrots, bread] | Since ArrayList is classes, you can use its add method to add values, described in detail in the next lesson. | |
| Extensions | 1.Although it is not on the AP exam, you can convert arrays to ArrayLists using its constructor with an argument Arrays.asList(arrayname) like the following.  2.Note that ArrayLists have a toString() method that is automatically called to print the list in a nice format.  import java.util.\*;  public class ArrayListFromArray  {  public static void main(String[] args)  {  String[] names = {"Dakota", "Madison", "Brooklyn"};  ArrayList<String> namesList = new ArrayList<String>(Arrays.asList(names));  System.out.println(namesList);  }  }  // results: [Dakota, Madison, Brooklyn] | | |
| Summary  (ESSENTIAL KNOWLEDGE  ) | 1. An ArrayList object is mutable and contains object references. 2. The ArrayList constructor ArrayList() constructs an empty list. 3. Java allows the generic type ArrayList<E>, where the generic type E specifies the type of the elements. 4. When ArrayList<E> is specified, the types of the reference parameters and return type when using the methods are type E. 5. ArrayList<E> is preferred over ArrayList because it allows the compiler to find errors that would otherwise be found at run-time. | | |
| Quiz | 1. Consider the following statement, which is intended to create an ArrayList named theater\_club to store elements of type Student. Assume that the Student class has been properly defined and includes a no-parameter constructor.   ArrayList<Student> theater\_club = new /\* missing code \*/;  Which choice can replace /\* missing code \*/ so that the statement compiles without error?  A Student()  B Student ArrayList()  C ArrayList(Student)  D ArrayList<Student>()  E ArrayList<theater\_club>()   1. Consider the following statement, which is intended to create an ArrayList named values that can be used to store Integer elements.   /\* missing code \*/ = new ArrayList<>();  Which of the following can be used to replace /\* missing code \*/ so that the statement compiles without error?  I ArrayList values  II ArrayList<int> values  III ArrayList<Integer> values  A I only  B II only  C III only  D I and III only  E II and III only   1. Consider the following statement, which is intended to create an ArrayList named years that can be used to store elements both of type Integer and of type String.   /\* missing code \*/ = new ArrayList();  Which of the following can be used to replace /\* missing code \*/ so that the statement compiles without error?  A ArrayList years  B ArrayList years()  C ArrayList years[]  D ArrayList<Integer> years  E ArrayList years | | |