**44-542 Object-Oriented Programming**

**Lab04: Arrays and ArrayLists**

**Objective:** Covers the usage of **Arrays, ArrayLists and String Formatting.**

**NOTE:**

* Do not hard code any values.
* Check the sample output to know how the results need to be printed.
* Read every instruction carefully and follow them strictly.
* Do not change the name of the attributes, and methods given below.

**Overview:**

1. You need to import the project provided to you.
2. Create classes **Card**, **CardArray**, **CardArrayList**
3. **Card** class will have information about types of cards and number of cards.
4. **CardsArray** Class creates an array of Cards objects.
5. **CardsArrayList** creates an **ArrayList** of cards objects.
6. **CardDriver** will be provided to you partially.

**Steps for Project Creation:**

1. Download the Project from course website, unzip the file ***Lastname*\_Lab04ArraysAndArrayList.zip.**
2. Open the project from NetBeans and rename the project to ***Lastname*\_Lab04ArraysAndArrayList** where **Lastname** is your last name.
3. The driver class is already created in cards package.
4. All the other classes you must create must also be placed in the same package, named **cards**.
5. Create three new Java classes and name them as **Card, CardsArrayList, and CardsArray respectively**.

**Card Class**

This class contains the below attributes, shown in the table. All attributes are private.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Attribute Description** |
| **number** | **int** | Number of Playing cards |
| **Type** | **String** | Type of the card like spade, diamond, hearts & clubs |

1. Constructor: There will be one constructor with parameters, with the following prototype. Constructor with two parameters.

**public Card(int number, String type)**

1. Write getter & setter methods for the above mentioned two attributes.
2. **toString():** It must return the number on card. See the Sample output for formatting

**CardsArrayList Class**

This class has the following private attributes:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Attribute Description** |
| **clubs** | **ArrayList<Card>** | Array list of type card |
| **diamonds** | **ArrayList<Card>** | Array list of type card |
| **hearts** | **ArrayList<Card>** | Array list of type card |
| **spades** | **ArrayList<Card>** | Array list of type card |

1. Constructor**:** There will be one no-argument constructor, with the below prototype used to initialize all the ArrayLists declared.

**public CardsArrayList()**

1. Methods: The description of methods is given below. All the methods are public. Do not add any public methods beyond those shown here.
   1. Create getter methods for the attributes declared.
   2. **addCardsToList(Cards c):**  Returns a **String** value. This method will take the cards object as a parameter. Using this object, you can find the type of card and use it in a switch statement. Based on the type of card like “clubs”, “diamonds”, “hearts” or “spade”, add the cards to their respective ArrayList. Return the message as respective **“card added successfully”** for every successful addition. Check the sample output for details.
   3. **removeCardsFromList(int position, ArrayList<Card> c):** This method returns a **String.** It takes in the **position** at which the element needs to be removed. It also takes in the ArrayList of cards as input parameters from which card needs to be removed. The goal of this method is to remove elements from array list at a specified position. If the **position** passed is less than zero or if the **position** is greater or equal to the size of the ArrayList passed then it must return a message as “**ArrayList size underflow, card cannot be removed**”. Else this method must remove the element from the array list and return a message as **“Card is removed successfully!”**
   4. **toString():** This method returns **String.** Using enhanced for loop for each type of card, iterate through list of all the cards like hearts, diamonds, spades and clubs. Print the list of element present in all the array list of cards (Separate enhanced for loop for each card type). See the Sample output.

**CardsArray Class**

This class remains same as Array list class. However, the implementation must be performed based on the definitions of Array.

This class has the following private attributes:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Attribute Description** |
| **clubs** | **Card[]** | Array of type card |
| **diamonds** | **Card[]** | Array of type card |
| **hearts** | **Card[]** | Array of type card |
| **spades** | **Card[]** | Array of type card |
| **DECK\_SIZE** | **Int** | It is constant with value 5 |
| **clubscount** | **Int** | These attributes store the count of the cards |
| **diamondscount** | **Int** | These attributes store the count of the cards |
| **heartscount** | **Int** | These attributes store the count of the cards |
| **spadescount** | **Int** | These attributes store the count of the cards |

1. Constructor**:** There will be one no-argument constructor, with the below prototype used to create an empty array of all Cards with size set to **DECK\_SIZE**, and initialize **clubsCount, diamondscount, heartscount, spadescount** to 0.

**public CardsArray()**

1. Methods: The description of methods is given below. All the methods are public. Do not add any public methods beyond those shown here.
2. Create getter methods for **clubscount, diamondscount, heartscount, spadescount** variables.
3. **addCardsToArray(Cards c):** This method will take the cards object as a parameter. Using this object, you can find the type of card. Based on the type of card like “clubs”, “diamonds”, “hearts” or “spade”, add the card to the respective array at the end. However, you will need to check if the card count is not crossing the **DECK\_SIZE.** Moreover, increment the count by one on each successful addition. If **DECK\_SIZE** is less than or equal to the card count then remove the card at 0th position from the respective array and add the new card at the end of array.

**HINT:** Call **removeCardsFromArray(int position, String cardType)** method to remove card and call **addCardsToArray(Cards c)** to add the new card .

1. **removeCardsFromArray(int position, String cardType):** This method takes **position** and **cardType** as parameters. Use if-else statement to remove elements from “clubs”, “diamonds”, “spades”, and “hearts” type arrays at specified positon based on the card type. If the **position** is less than the card count or **position** is equal or greater than zero then it must remove the card from the specified position and other elements are shifted to fill the empty space. Also decrease the count by one on each successful removal.
2. **toString():** This method returns **String.** Using enhanced for loop for each type of card, iterate through all the cards like hearts, diamonds, spades and clubs. Print the list of element present in the array of cards. See the Sample output.

The driver reads from a file named **inputFile.txt**, which is provided for you. The driver must also print the output to a file named **outputFileText.txt**. This file must be created by you in the appropriate location. A sample output, using this input file, is at the end of this document.

**Note:**

* The output should be shown both on the console and saved to a file. Use **PrintWriter** to save the console output to a file.

**Sample Output:**

|  |
| --- |
| Array Data:  Hearts Array:[ 2 6 3 4 7 ]  Diamonds Array:[ 6 8 11 5 ]  Spades Array:[ 10 11 2 7 9 ]  Clubs Array:[ 5 9 13 4 11 ]  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ArrayList Data:  Hearts ArrayList:[ 5 2 6 3 4 7 ]  Diamonds ArrayList:[ 6 8 11 5 ]  Spades ArrayList:[ 13 10 11 2 7 9 ]  Clubs ArrayList:[ 6 12 5 9 13 4 11 ]  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Remove at position 3 from hearts arraylist  Remove at position 2 from spades arraylist  Remove at position 1 from clubs arraylist  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ArrayList Data after removing element:  Hearts ArrayList:[ 5 2 6 4 7 ]  Diamonds ArrayList:[ 6 8 11 5 ]  Spades ArrayList:[ 13 10 11 7 9 ]  Clubs ArrayList:[ 6 12 5 13 4 11 ] |

**Submit you solution by following the steps below:**

* Save your files in NetBeans.
* Zip your entire Project. (It should be called **Lastname\_Lab04ArraysAndArrayList**.**zip** where Last name is your last name.)
* Submit the Zip file to the Lab04ArrayAndArrayList dropbox.
* Download the Zip file you have submitted.
* Look in the Zip file and verify the class files, Javadoc in the Zip folder are updated. If not resave your project in NetBeans and resubmit.
* Make sure output file contains appropriate output.