Lab #1: One-dimensional Arrays

The main aim of the lab is to solve some common problems related to one-dimensional arrays.

Task 1: For a given class **MyArray.java** is as follows:

```
public class MyArray {
	private int[] array;
	public MyArray(int[] array) {
		this.array = array;
	}

	//...

}
```

Task 1.1: Implement some basic methods in class **MyArray.java** as follows:

```
	// Count the number of odd numbers in the array
	public int countOddNumbers() {
		// TODO
		return 0;
	}

	// get the index of the second even number in the array
	// return -1 (if not)
	public int indexOfSecondEvenNumber() {
		// TODO
		return -1;
	}

	// Method mirror that outputs the contents of an array in a reverse
	//order like a mirror
	//Example: input [1, 2, 3] ==> output: [1, 2, 3, 3, 2, 1]

	public  int[]  mirror() {
		// TODO
		return null;
	}
```

Task 1.2: Implement some advanced methods in class MyArray.java as follows:

```
	// GET the element which its occurence is the most in the array.
	// Input: int[] array = {12, 10, 9, 45, 2, 10, 10, 45}
	// Output: 10
	public int getMode() {
		// TODO
		return 0;
	}

	// GET unique elements in the array.
	// Input: int[] array = {12, 10, 9, 45, 2, 10, 10, 45}
	// Output: 12, 9, 2
	public int[] getUniqueElements() {
		// TODO
		return null;
```

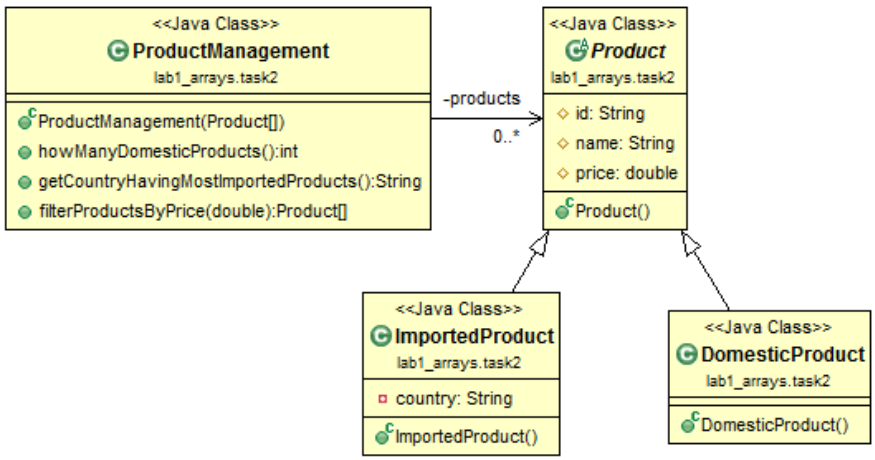
```
	}

	// GET distinct elements in the array.
	// Input: int[] array = {12, 10, 9, 45, 2, 10, 10, 45}
	// Output: 12, 10, 9, 45, 2
	public int[] getDistinctElements() {
		// TODO
		return null;
	}
```

Remember to test the implemented methods.

==========

Task 2: For a given class diagram as follows:



Implements some basic methods in ProductManagement.java

(Remember implementing OOP code)

```
public class ProductManagement {
	private Product[] products;

	public ProductManagement(Product[] products) {
		this.products = products;
	}

	// How many domestic products?
	public int howManyDomesticProducts() {
		// TODO
		return 0;
	}

	// Get the country name which most products are imported from
	public String getCountryHavingMostImportedProducts() {
		// TODO
		return null;
	}

	// Filter products having prices higher than a given price
	public Product[] filterProductsByPrice(double price) {
		// TODO
		return null;
	}
```

```
}
```

==========

Task 3 (Advanced): For a given class named MyCaesar using for encrypting and decrypting texts as follows:

```
public class MyCaesar {
	public static final char[] ALPHABET = { 'A', 'B', 'C', 'D', 'E', 'F',
'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O',
			'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z' };
	private int n;// shift steps (right shift)

	public MyCaesar(int shiftSteps) {
		this.n = shiftSteps;
	}
//....

}
```

Task 3.1: Implement the encrypt and decrypt methods for a charater and a text in the MyCaesar class. Write tests to check the correctness of the implemented methods.

```
// Encrypt a character according to the given shif steps.
// Encrypt: En(x) = (x + n) mod 26. x represents the index of c in the
ALPHABET
	// array
	public char encryptChar(char c) {
		// TO DO
		return 0;
	}

	// Encrypt a text using the above function for encrypting a charater.
	public String encrypt(String input) {
		// TO DO
		return "";
	}

	// Decrypt a character according to the given shif steps.
	// Decrypt: Dn(x) = (x - n) mod 26. x represents the index of c in
the ALPHABET array
	public char decryptChar(char c) {
		// TO DO
		return 0;
	}

	// Decrypt a encrypted text using the above function for decrypting a
charater.
	public String decrypt(String input) {
		// TO DO
		return "";
	}
```

Task 4 (advanced task): Expanding the problem in **Task 3** so that the program can encrypt and decrypt a given text including **numbers** and **charaters**.

Task 5 (advanced task): Expanding the problem in **Task 4** so that the program can encrypt and decrypt a given text including **numbers** and **charaters** where the text is entered from console by users.

Task 6 (advanced task): Expanding the problem in **Task 4** so that the program can encrypt and decrypt the text content in a text file using the supported methods for reading and writing text file.

```
// Encrypt a encrypted the text content in the srcfile and save it into
desFile.
	public void encrypt(String srcFile, String desFile) {
		// TO DO
	}

	// Decrypt a encrypted the text content in the srcfile and save it
into desFile.
	public void decrypt(String srcFile, String desFile) {
		// TO DO
	}
```