Exercises 10

Generics

1. **[EX] Practicing generic methods**
    a. Define 5 arrays. (of types int, double, String, Task and Point)
    b. Define a generic method to print the elements of an array.
    c. Overload the print method so that it can accept the lower and upper indexes to print a specified range of the array. If the range is out of bounds of the array, the method throws an exception.
    d. Overload the print method so that it prints Tasks as only two Tasks are printed in a single line.
    e. Define a generic method to return the middle element of an array.
    f. Define a generic method to return the maximum element of an array.
    g. Define a generic method to sort an array. (Use bubble sort algorithm for simplicity)
        i. You may need to swap some elements of the array, define a generic swap method.

2. **[EX] Generic Numeric class**
    a. Create a generic numeric function class that uses the bound type of numeric classes and has following methods:
    b. to return the reciprocal
    c. to return the fractional part of a number (for example, 9.76 should return 0.76)
    d. to check the absolute value of two types (for example, Double and Float) to see if they are equal.

3. **[PW] Generic Pair class**
    a. Make a generic Pair class (like Map.Entry). Make necessary changes to avoid raw types. Test it.

4. **[EX] Specific Element count**
    a. Create a generic method to count the number of elements in a collection that have a specific property (for example, even integers, palindrome, etc.).
    b. Write a test method for the specific example of
        i. even integers where you have a collection of integers and calling this generic method gives the number of even integers in the collection.
        ii. palindrome strings where you have a collection of strings and calling this method gives the number of palindrome strings in the collection.
        iii. Empty accounts where you have a collection of accounts and calling this generic method gives the number of accounts with zero balance in the collection.

5. **[PW] Stack enhanced**
    a. While learning OOP we had an exercise about creating Stack.
    b. Enhance that Stack class so that it can provide basic stack operations on any generic type.
        i. It might look alike the java.util.Stack
    c. If the Stack is full, attempt to push a new element should raise an exception.
    d. If the Stack is empty, attempting to peek or pop an element should raise an exception.