Guidelines

•No calculators or any other electronic devices are allowed in this exam.

Student ID:				Name:	
Section:				Instructor:	
1	2	3.1	3.2	Total	

Question 1 . . . . 30 points

1. Trace the execution of the following expression: **8 6 4 - / 6 4 3 % 2 + - > 8 7 2 1 - + ≠ &&**.
   Show the content of the data structure(s) **after** parsing each operation.

-	/	%	+	-
>	-	+	≠	&&

2. Trace the execution of the following expression: **((4 + 2) / (7 % 4)) + 2 - (5 * 2 +(6 / 3)) +2**.

   Draw the content of the data structure(s) after parsing each operation.

+	/	%	+	-
*	+	/	+	$

3. Consider the following code:

```java
public static void f1(int n) {
  n++;
  f3(n);
  f2(n);
  f3(n); }
public static void f2(int n) {
  f3(n);
  n++;
  f3(n); }
public static void f3(int n) {
  System.out.println(n); }
public static void main(String[] args) {
  int n = 3;
  f2(n);
  n++;
  f1(n); }
```

For each snapshot of the call stack below, indicate if it is valid for this code.

f3				f2	
f2	f3	f2	f3	f3	f3
f1	f1	f3	f2	f2	f1
main	main	f1	main	f1	f2
				main	main

Ⓐ Yes Ⓑ No   Ⓐ Yes Ⓑ No   Ⓐ Yes Ⓑ No   Ⓐ Yes Ⓑ No   Ⓐ Yes Ⓑ No   Ⓐ Yes Ⓑ No

Question 2 .... 15 points

We want to write an algorithm that reads a set of elements, and computes and store their frequencies. At the end, the algorithm prints each element with its frequency in the order of first appearance of the element.

Example 1. *If the input is "A, B, A, D, D, C, B", then the output will be:* A 2, B 2, D 2, C 1.

We want to design a data structure to store the elements according to their order of first appearance and keeps track of their frequencies.

1. Write a specification for a data structure that can be used in this algorithm. Your specification must be as simple as possible. Use the same format seen in class (name of the oepration, preconditions and results).

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

2. Translate this specification into a Java generic interface.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

Question 3 . . . . 55 points

We want to write an algorithm that takes as input a text, finds all the words that appear in it and prints them in random order without repetition.

Example 2. *If the input text is* "dogs hate cats and cats also hate dogs", *a possible output is:* hate, cats, dogs, and, also. *Notice that there are no repetitions.*

To achieve this, we will use a data structure that supports the following interface:

```
public interface RandomSet<T> {
  // Returns the number of elements in the set.
  int size();
  // Returns true if the set is full.
  boolean full();
  // Adds e to the set if it does not yet exist and returns true. If e exists already, the
      method returns false.
  boolean add(T e);
  // Removes a randomly selected element and returns it.
  T remRandom();
}
```

1. Write the class `ArrayRandomSet` which implements the interface `RandomSet` using array storage.

   Remark 1. *To get a random number in Java use the following:*

   ```
   import java.util.Random;
   ...
   Random rnd = new Random();
   int n = 5;
   int k = rnd.nextInt(n); // returns a random number between 0 and n-1
   ```

   . . . .

   . . . .

   . . . .

   . . . .

   . . . .

   . . . .

   . . . .

   . . . .

   . . . .

   . . . .

   . . . .

   . . . .

   . . . .

   . . . .

2. Write the method **public static <T> boolean** `find(RandomSet<T> s, T e)`, which returns **true** if e exists in s, false otherwise.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....