I Will the following code snippet compile? If not, why?

1.
```
public static void print(List<? extends Number> list) {
   for (Number n : list)
      System.out.print(n + " ");
}
```

2.
```
class MyClass<T> {
   public T getInstance() {
      T instance = new T();
      return instance;
   }}
```

3.
```
class MyClass<T> {
   public static T getFirst(T[] arr) {
      if(arr.length > 0)
         return arr[0];
      return null;
   }}
```

4.
```
interface I1<T> { ... }
class MyClass implements I1<T> { ... }
```

5.
```
class C1{}
class C2 extends C1 {}
class C3 extends C2 {}
public class exercise {
   public static void main(String args[]) {
      List<C3> list = new ArrayList<>();
      m1(list);
   }
   static void m1(List<? super C2> list){}
}
```

6.
```
interface I1{}
class C1{}
public class exercise {
   public static void main(String args[]) {}
   public static <T extends C1 & I1> void m2(){}
}
```

7.
```
<T super Number> void m3(T e){}
```

II Exercises

1. Define any class with overloaded generic methods
2. Write a generic method reverseArray that reverses the order of elements in an array.
3. Implement a generic method that takes an array of any type of number as argument and returns the sum of the elements of the array.
4. Given a non-generic class named C1 that has a member variable stack which was instantiated as: Stack<Number> stack = new Stack<>();
   a) Add a generic method to class C1 that receives a list as argument and pushes all elements of the list to stack. Use a wildcard type.
   b) Add a generic method to class C1 that receives a list as argument and adds all elements of stack to the list.
5. Design a generic class called SortedSet that defines a set which stores its elements in ascending order.