Worksheet #13:
Parallelism in Java Streams, Parallel Prefix Sums

Name: _____   Netid: _____

1. What output will the following Java Streams code print?

2. Which stream operation in this example could benefit from a parallel prefix sum implementation, and why?
    (Assume a larger array when answering this question, so that overheads of parallelism are not an issue.)

```
1.    Arrays
2.      .asList("a1", "a2", "b1", "c2", "c1")
3.      .parallelStream()
4.      .filter(s -> s.startsWith("c"))
5.      .sorted()
6.      .map(String::toUpperCase)
7.      .forEach(System.out::println);
```