Question 4

Use the following class to answer the following parts. Assume all code written is in a separate class.

```
public class ComparableCat implements Comparable {
    private String name;
    private int age;

    public ComparableCat(String name, int age) { this.name = name; this.age  age; }

    public int getAge() { return age; }
    public String getName() { return name; }

    @Override
    public int compareTo(Object o) {
        if (o instanceof ComparableCat c) return age - c.age;
        throw new illegalArgumentException(
            "Can't compare ComparableCat with" + o.getClass()
        );
    }
}
```

Part 1

Complete the function below which returns the number of cats in the given list with a name that begins with the given character. You must use only `List.stream()` and stream operations in your solution. You can convert strings to character arrays with the instance method `String.toCharArray()`. Assume that each object in the list has a name with at least 1 character.

```
public static int countNameStartsWith(List<ComparableCat> cats, char c) {





}
```

Part 2

Complete the function below to sort a list of cats by descending age. Use an *explicit* lambda function in your solution.

```
public static void sortDescending(List<ComparableCat> cats) {
}
```

Part 3

Complete the function below to sort a list of cats by ascending age. Use a method reference (key extractor) in your solution.

```
public static void sortAscending(List<ComparableCat> cats) {




}
```

Part 4

Complete the function below to sort a list of cats by ascending age. Use the fact that the `ComparableCat` class implements `Comparable`.

```
public static void sortAscendingComparable(List<ComparableCat> cats) {




}
```