Java Streams

Demonstrative Examples

map

```
public static List<String> transform7(List<String> collection) {
              List<String> coll = new ArrayList<>();
              for (String element : collection) {
                  coll.add(element.toUpperCase());
              }
              return coll;
          }
 public static List<String> transform8(List<String> collection) {
               return collection.stream() // Convert collection to Stream
                       .map(String::toUpperCase) // Convert each element to upper case
                       .collect(Collectors.toList()); // Collect results to a new list
          }
```

filter

```
public static List<String> transform7(List<String> collection) {
              List<String> newCollection = new ArrayList<>();
              for (String element : collection) {
                  if (element.length() < 4) {
                      newCollection.add(element);
                  }
              }
public static List<String> transform(List<String> collection) {
              return collection.stream() // Convert collection to Stream
                      .filter(value -> value.length() < 4) // Filter elements with length smaller than 4 characters
                      .collect(Collectors.toList()); // Collect results to a new list
          }
```

flatMap

```
public static List<String> transform7(List<List<String>> collection) {
              List<String> newCollection = new ArrayList<>();
              for (List<String> subCollection : collection) {
                  for (String value : subCollection) {
                      newCollection.add(value);
                  }
              }
public static List<String> transform(List<List<String>> collection) {
              return collection.stream() // Convert collection to Stream
                      .flatMap(value -> value.stream()) // Replace list with stream
                      .collect(Collectors.toList()); // Collect results to a new list
          }
```

max

```
public static Person getOldestPerson7(List<Person> people) {
        Person oldestPerson = new Person("", 0);
        for (Person person : people) {
            if (person.getAge() > oldestPerson.getAge()) {
                oldestPerson = person;
            }
        }
public static Person getOldestPerson(List<Person> people) {
        return people.stream() // Convert collection to Stream
                .max(Comparator.comparing(Person::getAge)) // Compares people ages
                .get(); // Gets stream result
    }
```

sum and reduce

```
public static int calculate7(List<Integer> numbers) {
        int total = 0;
        for (int number : numbers) {
            total += number;
        }
        return total;
    }
public static int calculate(List<Integer> people) {
        return people.stream() // Convert collection to Stream
                .reduce(0, (total, number) -> total + number); // Sum elements with 0 as starting value
    }
```

filter and map

```
public static Set<String> getKidNames7(List<Person> people) {
            Set<String> kids = new HashSet<>();
            for (Person person : people) {
                if (person.getAge() < 18) {
                    kids.add(person.getName());
                }
            }
public static Set<String> getKidNames(List<Person> people) {
            return people.stream()
                    .filter(person -> person.getAge() < 18) // Filter kids (under age of 18)
                    .map(Person::getName) // Map Person elements to names
                    .collect(Collectors.toSet()); // Collect values to a Set
    }
```