Pittawat Taveekitworachai 61130500220

4a

1.

```
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
Yes! Extract Num.word
Yes! Extract Num.word
Yes! Extract Num.word
Finalist: two
Finalist: three
["two", "three"]
```

2.

```
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
Yes! Extract Num.word
Yes! Extract Num.word
Yes! Extract Num.word
Finalist: three
Finalist: ten
["three", "ten"]
```

3.

```
1st char = t?
1st char = t?
1st char = t?
Yes! Extract Num.word
Finalist: two
1st char = t?
Yes! Extract Num.word
Finalist: three
[two, three]
```

AND

1st char = t?

```
1st char = t?
1st char = t?
Yes! Extract Num.word
1st char = t?
Yes! Extract Num.word
Finalist: three
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
1st char = t?
Yes! Extract Num.word
Finalist: ten
[three, ten]
```

4.

The expected result and the actual result are different. This is because of the fact that the stream working involve the **pipelining** and **short circuiting**. The stream takes input and does not work one stage with every input and move to next stage. But rather one input will go through the whole pipeline. This can be seen from the first `limit` example. `zero`, `one` does not match the filter, so it only reach the `1st char = t?` stage. Unlike the first two, the `two` pass through the filter and pass through the `Yes! Extract`

`Num.word` in the map and `Finalist: two` in the second map. The same is also true for the `three` case. Some methods in the stream pipeline also short-circuited. For example, the `limit(2)` also count and see if it reach the desired number. Once it is, it short circuited and done. This proves to be useful for the case of big data, so we don't have to involve and work with the whole dataset. For the second `skip` example. We found that it prevent the first one the reach the stage behind the `skip`. This example also iterate until reach the end of the list, since we want all the rest except for the first one.