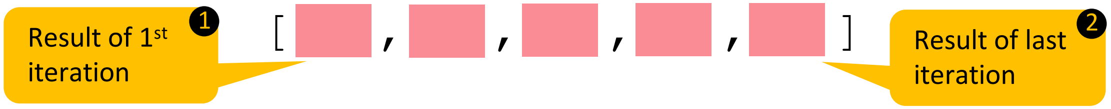
Advanced looping techniques

Store results of a loop in a vector

- Instead of "printing" the results of every loop iteration we store it in a vector:



1 Result of 1st iteration

2 Result of last iteration

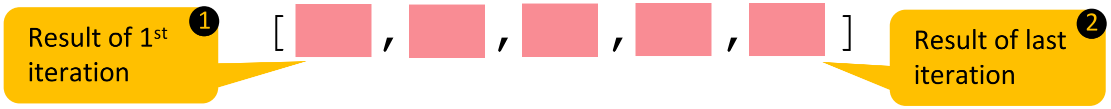
- Let's find out the numbers of characters of each city and store the result in a vector:

```
cities <- c("New York", "Paris",
            "London", "Tokyo", "Cape Town")

numChars <- c()

for(i in 1:length(cities)){
     numChars[i] <- nchar(cities[i])
}
numChars

>>>[8, 5, 6, 5, 9]
```

3 `nchar()` counts the number of characters

Store results of a loop in a vector

- Instead of "printing" the results of every loop iteration we store it in a vector:



- Let's find out the numbers of characters of each city and store the result in a vector:

```
cities <- c("New York", "Paris",
            "London", "Tokyo", "Cape Town")

numChars <- c()

for(i in 1:length(cities)){
    numChars[i] <- nchar(cities[i])
}
numChars

>>>[8, 5, 6, 5, 9]
```

3 `nchar()` counts the number of characters

Substitute loops with the `sapply()` function

- Easy and fast to avoid loops and write less code.
- `sapply()` allows to apply a function over a vector

```
cities <- c("New York", "Paris",
            "London", "Tokyo", "Cape Town")

sapply(cities, nchar)
New York     Paris    London     Tokyo Cape Town
      8         5         6         5         9

sapply(cities, nchar,  USE.NAMES=FALSE)
[1] 8 5 6 5 9
```

Remove names

Important: The result of the `sapply()` function will always have the same type!

What is vectorization?

- Usually one would loop through a vector (or use `sapply()`) to apply a function on every value of the vector.

- R supports vectorization of operations:

$$\begin{aligned}1+1&=2\\2+2&=4\\3+3&=6\end{aligned}$$

Loop version 1

$$\begin{bmatrix}1\\2\\3\end{bmatrix}+\begin{bmatrix}1\\2\\3\end{bmatrix}=\begin{bmatrix}2\\4\\6\end{bmatrix}$$

Vectorized version 2

- In R, vectorized code is always faster than loops and you need to write less code.

Substitute loops with vectorization

In R, most function are "vectorized": they not just work on a single value, but on a whole vector at the same time.

```
cities <- c("New York", "Paris",
            "London", "Tokyo", "Cape Town")

nchar(cities)
[1] 8 5 6 5 9
```

When does using `for` loops make sense over relying on vectorization?

- If you are using functions that do not work with vectors.
- In loops where each iteration is dependent on the results of the previous iterations.