Loopy for Loops and Silly for Samples - Hands-on!

YOUR NAME

2023-10-11

Part 1: Loops

`for` loops

First, we are going to be focusing on "for loops". A for loop applies a command to each value provided then stops. In other words, the `for` loop runs for a preset number of times.

Basic example:

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```

- You don't actually have to use i for the value there but most people do. `i` is presumably short for "index", which makes sense. But you could just as well call it `banana`.

Ex.1) run the same command above replacing `i` with `banana`.

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```

So this loop printed each value for `i` (or `banana`). How would you write code that printed the values 6-10?

```
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
```

It's often useful to define a placeholder variable before running your loop. For example, this loop calculates the mean of a given data set.

```
## [1] 11
```

You can also write loops with character vectors. Here's a character vector: `days<-c("Mon", "Tues", "Wednes", "Thurs", "Fri", "Satur", "Sun")`

Ex.2) Write a loop that adds day to each day of the week so that the output of `days` gives you "Monday", "Tuesday", etc, one in each line.

Hint: if you add the argument sep = " " to paste you don't get a space in between)

```
## [1] "Monday"
## [1] "Tuesday"
## [1] "Wednesday"
## [1] "Thursday"
## [1] "Friday"
## [1] "Saturday"
## [1] "Sunday"
```

`while` loops

Ex.3) A while loop repeats code until a condition is met. It's really important to include the condition because the loop might continue forever if you don't.

Here's an example of a loop that prints x then multiplies it by two, as long as x is under 16.

```
## [1] 2
## [1] 4
## [1] 8
```

Now write a while loop that prints the value, then doubles it, starting at 1 and stopping at 100.

```
## [1] 1
## [1] 2
## [1] 4
## [1] 8
## [1] 16
## [1] 32
## [1] 64
```

Just for fun try running that loop without with while(x). This will run forever. Make sure you remember to add the condition!

Say you have a number and you want to find the smallest divisor that isn't 1. You could use a while loop to do this.

```
## [1] 7
```

You can do a lot more with loops once you learn about functions. But alas, that's for next lab. For now that's probably enough for loops.

Recommended: This guide someone wrote on the internet is really good if you want to also read it. https://intro2r.com/loops.html

---

Part 2: Sampling

There is a basic function in R for taking samples very appropriately named sample. You need three arguments for it to work (there is a 4th also, probability). The first is the data set, the second is the number of samples, and the third is with or without replacement.

Ex.4) Let's with some basic sampling. Imagine you have a coin, equal chances of heads and tails. You want to flip it 10 times and see how many heads and how many tails. First make a vector called coin, with heads and tails. then sample it 10 times, with replacement

```
##  [1] "tails" "tails" "tails" "heads" "tails" "tails" "tails" "heads" "tails"
## [10] "tails"
```

Now lets try and do this same thing with a for loop. Make an empty vector called all_tosses and use it to generate 10 coin tosses. Hint:Because you are looping over the sample you are only sampling one each time

Now I'm being sneaky. I challenge you to flip a coin but its weighted. It lands on heads 60% of the time. Use the same loop and add the probabilities to the sample function. Hint: make them decimals in a vector.

Now let's see what happens if we slip both coins even more times. Use your original loop with the fair coin and toss is 100 times. Then take the weighted coin, change the variable names to make them different (add the word weighted or rigged) and toss that 100 times. Then count how many heads and how many tails came from each coin. Make sure to make an empty vector for each loop. hint: you can use the table function to get a count of heads and tails

```
## all_tosses
## Heads Tails
##    52    48
```

```
## all_tosses_weighted
## Heads Tails
##    55    45
```

This is why you shouldn't play games with other peoples strange coins

Ex.5) Read in the `human_genes.csv` file and name it `human_genes`. Play around with it using your strong exploratory data analysis skills. Get a good feel for the data set.

Overall goal of this exercise is to compare sample means at different n to the population mean.

Its big right? So let's take some samples.

NOTE: Your output will look different to the pdf because your samples will be different values.