CSC 120 (R Section)— Lab Exercise 11

This is a non-credit exercise, which you do not hand in.
You may work on your own or together with another student, as you please.

This is the last lab. You might want to work on any exercises from last week's or earlier labs that you didn't get to then. I've also included some more exercises here for general programming practice.

Swap years born and died entered incorrectly.

Suppose we have a data frame containing data on people, that has columns `born` and `died` that are supposed to be the years that each person was born and died. However, it seems that for some people, these were mistakenly swapped. Write a function `swap_born_died` that swaps these dates for all people for which the year they are said to have been born is later than the year they are said to have died.

For example:

```
> d1 <- data.frame (list(sex=c("M","F","F","M","M"),
+                        born=c(1897,1941,1902,1910,1923),
+                        died=c(1977,1902,1988,1931,1888))
+ )
> d1
  sex born died
1   M 1897 1977
2   F 1941 1902
3   F 1902 1988
4   M 1910 1931
5   M 1923 1888
> swap_born_died(d1)
  sex born died
1   M 1897 1977
2   F 1902 1941
3   F 1902 1988
4   M 1910 1931
5   M 1888 1923
```

You can try solving this using a loop, and without a loop (using vector operations).

Once you have this version working, you can try modifying your function so that it doesn't try to swap `born` and `died` if either of these is NA. You'll need to come up with a data frame with some NA values to test it on.

Create a matrix from two string vectors.

Write a function called `string_matrix_from_string_vectors`, which takes two vectors of strings as arguments, and returns a matrix of strings with number of rows equal to the length of the first argument and number of columns equal to the length of the second argument. The element in row i and column j should be a string of two characters — the first character of element i of the first argument and the first character of element j of the second argument. The row names of the matrix should be set to the first argument and the column names to the second argument.

For example:

```
> string_matrix_from_string_vectors(c("apple","orange","peach"),c("joe","mary"))
       joe  mary
apple  "aj" "am"
orange "oj" "om"
peach  "pj" "pm"
```

Recall that you can use `paste` to put strings together and `substring` to extract part of a string. Use `help` for the details if you don't remember them.

Find the length of the longest run in a vector.

Write a function `find_longest_run` that takes one argument that is a vector, and returns the length of the longest "run" in this vector. A "run" is a sequence of consecutive values that are all the same. You can assume that none of the elements of the vector are NA.

For example:

```
> longest_run(c(5,1,1,3,2,2,2,7))
[1] 3
> longest_run(c(5,1,3,3,2,7))
[1] 2
> longest_run(c(8,3,1,3,8))
[1] 1
> longest_run(1:1000)
[1] 1
> longest_run(7)
[1] 1
```