Control Structures

Naomi Tague

January, 2022

Exchange functions

Review DataTypes.Rmd

How do you figure out the rarest fish in our simulated ocean

Two key "take homes"

- working with *factors*
- how to return multiple items from a function using *list*

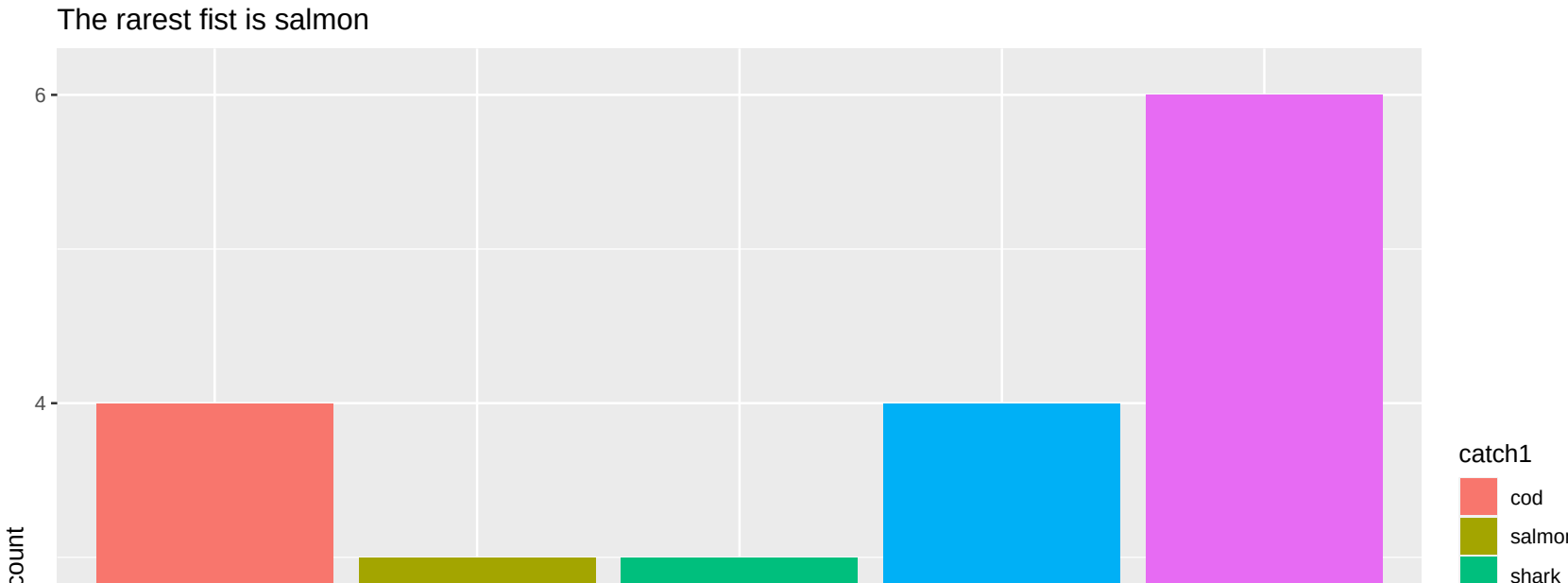
Answer

```
possible.fish = c("salmon","steelhead","shark","tuna","cod"
catch1 = base::sample(possible.fish, size=20, replace=T)
rarestfish = names(which.min(summary(as.factor(catch1))))
rarestfish
```

```
## [1] "salmon"
```

```
plottitle = sprintf("The rarest fist is %s", rarestfish)
ggplot(data.frame(catch1=catch1), aes(catch1, fill=catch1))
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pa
```



Flow Control (think of steering your program)

Another KEY concept is flow control In your function you do different things depending on a conditions

CLASSIC example is

IF then ELSE

If you have multiple conditions we can use case_when Here's a silly simple example of how it works

Simple example of flow control with if

```
mycortest  = function(x,y, thresh=0.8) {
  # compute correlation
  res = cor(x,y)
  classification = ifelse(res > thresh, "GOOD", "NotGood")
  return(classification)
}

a = runif(min=1, max=100, n=100)
b = runif(min=1, max=100, n=100)

mycortest(a,b)

## [1] "NotGood"

mycortest(a,a)

## [1] "GOOD"

# this doesn't work - why?
mycortest(a, 1)
```

Flow control with a simple if

```r
# Simple "IF*
# imagine we are trying to get a tuna  - lets "fish" by sa
possible.fish = c("salmon","steelhead","shark","tuna","cod"
catch1 = base::sample(possible.fish, size=1, replace=T)
catch1
```

```
## [1] "shark"
```

```r
ifelse(catch1 == "tuna", "success", "tryagain")
```

```
## [1] "tryagain"
```

```r
catch1 = "tuna"
ifelse(catch1 == "tuna", "success", "tryagain")
```

```
## [1] "success"
```

multiple alternatives

what if we have more than one category of fish - grade A, B, C
steelhead are A, tuna are B and everything else is C

R and other languages have ways to do this multiple alternatives
flow control in R an example is

case_when

case_when(

condition ~ response,

condition ~ response . . .)

```
fish = "steelhead"
```

```
case_when((fish =="steelhead")  ~ "A", (fish =="tuna") ~ "B
```

```
## [1] "A"
```

```
# apply to all of our ocean
# start by making a function
```

A more interesting example

Lets imagine that we are monitoring pollution in a lake, and we want to write a function that will let us know (flag) if risk associated with nutrient pollution are high, medium or low

From ecological studies, we know that Risk is high if water temperature is greater than a threshold for more than 5 days, and mean nutrient concentration is greater than a high threshold

Risk is medium if water temperature is greater than a threshold for more than 5 days and mean nutrient concentration is greater than a medium threshold

Inputs:
nutrient concentration for at least 5 days temperature for at least 5 days thresholds for temperature and nutrient (with default values)

Output:

Mean Nutrient Concentration Pollution Risk as “low”, “med” or “high”

What we've learned

- how to write a function (and add error checking)
- how to generate data
- how to repeat in code (different types of looping)
- how to make choices (flow control)

Assignment -on your own

Write a function that takes a vector of fish names and always returns three items

- the most common fish,
- the rarest fish
- the total number of fish

Create an Rmarkdown to demonstrate the use of your function with fish.txt - which is under Data on ESM_262_Examples

[Data on ESM_262_Examples]{https://github.com/naomitague/ESM_262_Examples/blob/main/Data/fish.txt}

Turn in on Gauchospace what your function returns when you run with fish.txt!

Challenge: What if we had multiple catches - how would you run your summary function for all of those catches - see below for an example to generate multiple catches You don't have to run this one in but we will go over in class