01-10: Complex Conditional Statements

1 - Purpose

- create conditional statement that check multiple variables for multiple conditions
- using parenthesis to break up larger conditional statements
- create numerical ranges using conditional statements

1.1 - Student-Instructor meeting

This is the last lesson in Unit 1, which means it is time to schedule a 20-30 minute meeting with the instructor. Please email the instructor and provide some times that you are available.

2 - Questions about the material...

If you have any questions about the material in this lesson *feel free to email them to the instructor here*.

3 - Breaking up complex conditional statements

In the last lesson we dealt with conditional statement that checked multiple conditions on one variable using logical operators (**&&** and **||**)

```
if ( favAnimal == "Llama" || favAnimal == "Alpaca" )
if ( age > 30 && age < 40 )
```

And conditional statements that check a single condition on multiple variables using logical operators (**&&** and **||**):

```
if ( age > 20 && gender == "male" )
if ( favAnimal == "Llama" && favCheese == "Muenster" )
if ( weather == "rainy" || temperature < 40 )
```

Now we want to check multiple conditions on multiple variables. The conditional statement we will look at in this lesson are (in English):

1. ***if (age*** is between 20 and 50 and ***gender*** is male ***)***
2. ***if (favAnimal*** is Llama or Alpaca or Goat and ***favCheese*** is Muenster or Swiss ***)***
3. ***if (weather*** is rainy or windy or the ***temperature*** is less than 40 or greater than 90 ***)***

3.1 - Separating variables using parentheses

The strategy for creating a conditional statement with multiple conditions on multiple variables is to separate the variables using parentheses. Parentheses in programming have the same function as parentheses in

Algebra -- they are used determine the order-of-operations and they also help break larger problems (in this case, conditional statements) into multiple smaller problems.

So we take a statement like:

```
if( age is between 20 and 50 and gender is male )
```

and separate the variables using parenthesis so it becomes:

```
if ( ( age is between 20 and 50 ) and ( gender is male )  )
```

Since the age grouping and the gender grouping has to both be ***TRUE*** is order for the conditional statement to be ***TRUE***, we know the **&&** operator connects the groupings:

```
if ( ( age is between 20 and 50 ) && ( gender is male )  )
```

Gender is male is the same as saying gender == "male":

```
if ( ( age is between 20 and 50 ) && ( gender == "male" )  )
```

The conditional statement for age between 20 and 50 is: age greater than 20 and age less than 50 so:

```
if ( ( age > 20 && age < 50 ) && ( gender == "male" )  )
```

And there we have it. The important parts are the inner parentheses that break up the larger conditional statement into more manageable chunks.

3.2 - Parenthesis within parenthesis in script

The parentheses, like in Algebra, sets the order of operation so that the conditional statements inside the inner parentheses...

```
if (( age > 20 && age < 50 ) && ( gender == "male" ) )
```

get evaluated before the conditional statements in the outer parentheses

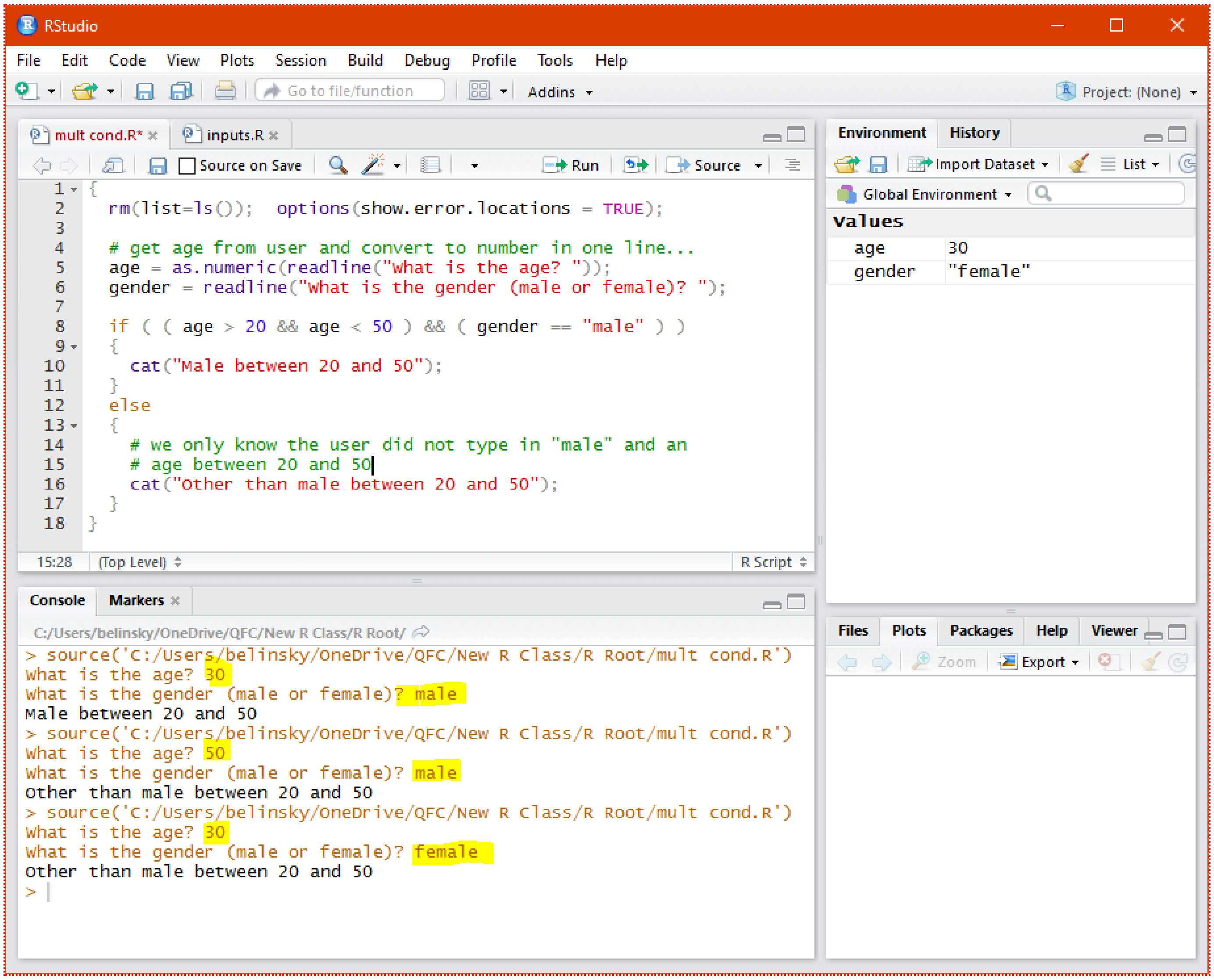
```
if ( ( age > 20 && age < 50 ) && ( gender == "male" ) )
```

Putting the conditional statement into a script:

```
{
  rm(list=ls()); options(show.error.locations = TRUE);

  # get age from user and convert to number in one line...
  age = as.numeric(readline("What is the age? "));
  gender = readline("What is the gender (male or female)? ");

  if ( ( age > 20 && age < 50 ) && ( gender == "male" ) )
  {
     cat("Male between 20 and 50");
  }
  else
  {
     # all we know is that the user did not type in "male" AND
     # an age between 20 and 50
     cat("Other than male between 20 and 50");
  }
}
```



```
{
  rm(list=ls());  options(show.error.locations = TRUE);

  # get age from user and convert to number in one line...
  age = as.numeric(readline("What is the age? "));
  gender = readline("What is the gender (male or female)? ");

  if ( ( age > 20 && age < 50 ) && ( gender == "male" ) )
  {
    cat("Male between 20 and 50");
  }
  else
  {
    # we only know the user did not type in "male" and an
    # age between 20 and 50
    cat("Other than male between 20 and 50");
  }
}
```

Values

age	30
gender	"female"

```
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is the age? 30
What is the gender (male or female)? male
Male between 20 and 50
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is the age? 50
What is the gender (male or female)? male
Other than male between 20 and 50
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is the age? 30
What is the gender (male or female)? female
Other than male between 20 and 50
> 
```

Fig 1: Checking multiple conditions on multiple variables

4 - Multiple conditions for multiple variables

The previous example had two conditions for the first variable (***age***), and one condition for the second variable (***gender***). The second example has multiple conditions for each variable (***favAnimal*** and ***favCheese***):

```
if ( favAnimal is Llama or Alpaca or Guanaco and favCheese is Muenster or Swiss )
```

Once again, the first step is to separate the variables (***favAnimal*** and ***favCheese***) using parentheses:

```
if ( ( favAnimal is Llama or Alpaca or Guanaco ) and
     ( favCheese is Muenster or Swiss ) )
```

The **&&** says that both conditionals inside the inner parentheses must be ***TRUE*** in order for the conditional statement to be ***TRUE:***

```
if ( ( favAnimal is Llama or Alpaca or Guanaco ) &&
     ( favCheese is Muenster or Swiss ) )
```

"Is" means is equal to, which is the (**==**) operator:

```
if ( ( favAnimal == "Llama" or "Alpaca" or "Guanaco " ) &&
     ( favCheese == "Muenster" or "Swiss" ) )
```

But we must be explicit and include the variable with every conditional operation:

```
if ( ( favAnimal == "Llama" or favAnimal == "Alpaca" or favAnimal == "Guanaco " )
       && ( favCheese == "Muenster" or favAnimal == "Swiss" ) )
```

Finally, we replace the "or" with the (**||**) operator and get the conditional statement:

```
if ( ( favAnimal == "Llama" || favAnimal == "Alpaca" || favAnimal == "Guanaco" )
        && ( favCheese == "Muenster" || favCheese == "Swiss" ) )
```

We might want to break up this statement into multiple lines -- just be careful keeping the parentheses in order. The outer conditional statement (parenthesis and logical operator) is highlighted:

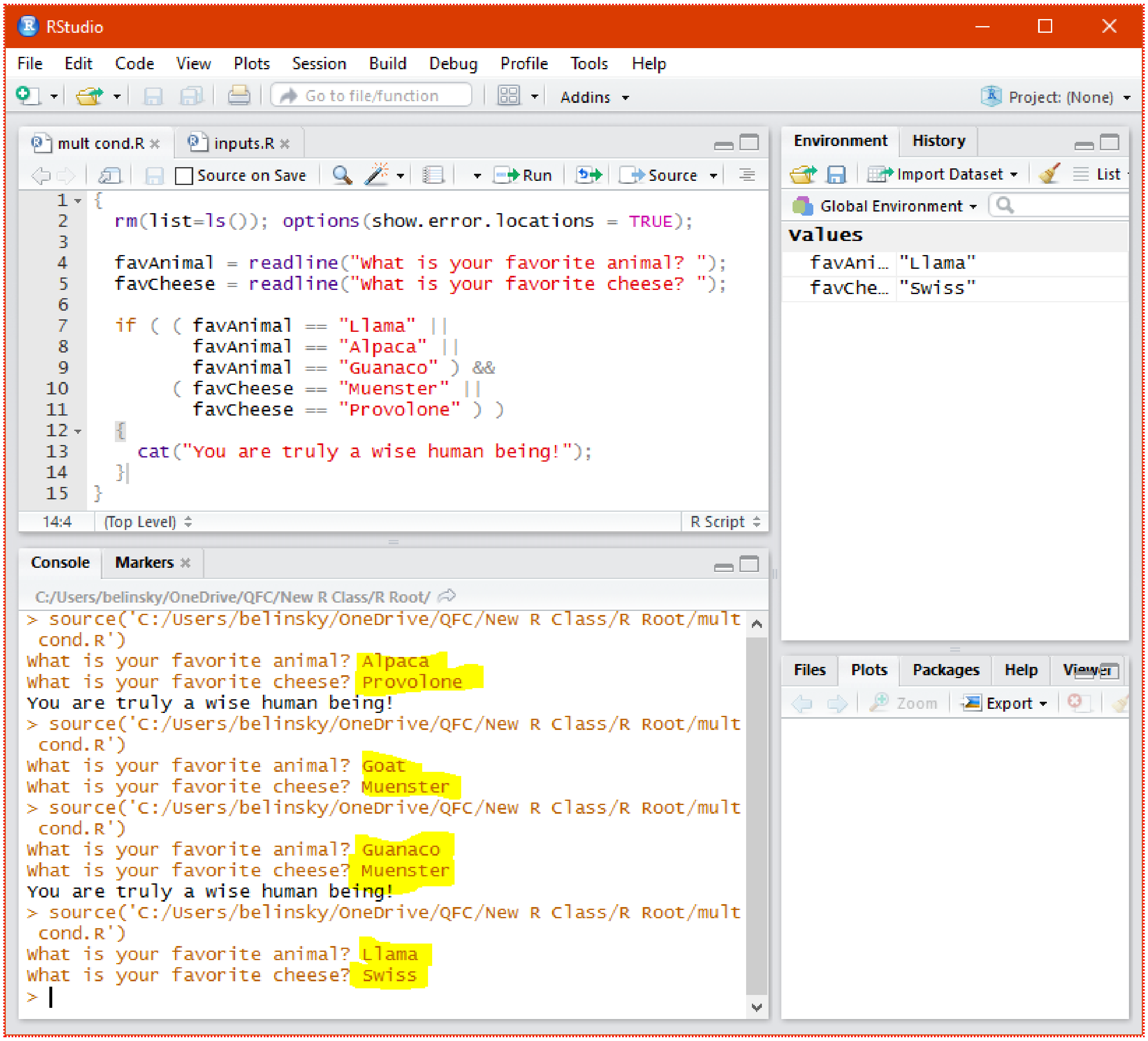
```
if ( ( favAnimal == "Llama" ||
       favAnimal == "Alpaca" ||
       favAnimal == "Guanaco" ) &&
     ( favCheese == "Muenster" ||
       favCheese == "Swiss" ) )
```

Lets add a message if the person gets both the cheese and the animal right:

```
{
  rm(list=ls()); options(show.error.locations = TRUE);

  favAnimal = readline("What is your favorite animal? ");
  favCheese = readline("What is your favorite cheese? ");

  if ( ( favAnimal == "Llama" ||
         favAnimal == "Alpaca" ||
         favAnimal == "Guanaco" ) &&
       ( favCheese == "Muenster" ||
         favCheese == "Provolone" ) )
  {
     cat("You are truly a wise human being!");
  }
}
```



```
{
  rm(list=ls()); options(show.error.locations = TRUE);

  favAnimal = readline("What is your favorite animal? ");
  favCheese = readline("What is your favorite cheese? ");

  if ( ( favAnimal == "Llama" ||
         favAnimal == "Alpaca" ||
         favAnimal == "Guanaco" ) &&
       ( favCheese == "Muenster" ||
         favCheese == "Provolone" ) )
  {
    cat("You are truly a wise human being!");
  }
}
```

14:4 (Top Level) R Script

Values

favAni…	"Llama"
favChe…	"Swiss"

```
C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult
 cond.R')
What is your favorite animal? Alpaca
What is your favorite cheese? Provolone
You are truly a wise human being!
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult
 cond.R')
What is your favorite animal? Goat
What is your favorite cheese? Muenster
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult
 cond.R')
What is your favorite animal? Guanaco
What is your favorite cheese? Muenster
You are truly a wise human being!
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult
 cond.R')
What is your favorite animal? Llama
What is your favorite cheese? Swiss
> 
```

*Fig 2: Checking multiple **favCheese** and **favAnimal** conditions*

5 - If-Else-If structures and multiple variables

Perhaps we also want to check if the person got *either the cheese or the animal right*. We can extend the ***if-else*** structure to check for the *cheese-only* and *animal-only* conditions by using an ***if-else-if*** structure:

```
if (animal and cheese are correct)   # 1st test

else if (animal is correct)          # 2nd test

else if (cheese is correct)          # 3rd test

else  // the 3 tests all returned FALSE

```

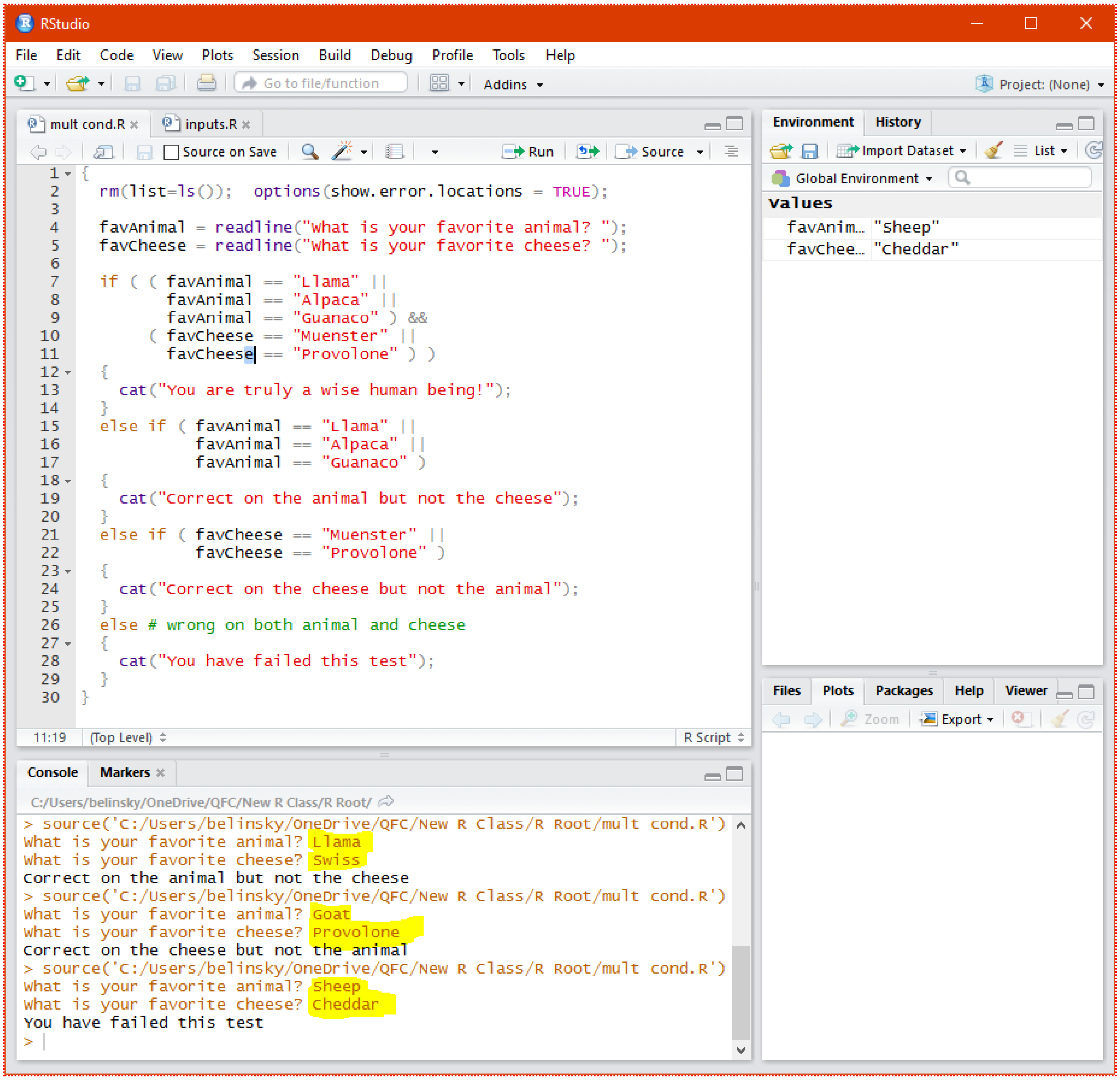
Remember that an ***if-else-if*** executes in order and exits as soon as one of the conditional statements is ***TRUE***. So:

- If both the animal and cheese are correct (*1st test*), the scripts exits the ***if-else-if*** structure and does not do the other tests.
- If the ***animal-and-cheese*** test fails, the script will do the *2nd test* (***animal-only***).
- If the ***animal-only*** test fails, the script will do the *3rd test* (***cheese-only***).
- If the ***cheese-only*** test fails then we know both animal and cheese are wrong and the ***else*** statement is executed

```
{
  rm(list=ls()); options(show.error.locations = TRUE);

  favAnimal = readline("What is your favorite animal? ");
  favCheese = readline("What is your favorite cheese? ");

  if ( ( favAnimal == "Llama" ||
         favAnimal == "Alpaca" ||
         favAnimal == "Guanaco" ) &&
       ( favCheese == "Muenster" ||
         favCheese == "Provolone" ) )  # 1st test (animal and cheese)
  {
     cat("You are truly a wise human being!");
  }
  else if ( favAnimal == "Llama" ||
            favAnimal == "Alpaca" ||
            favAnimal == "Guanaco" )   # 2nd test (animal only)
  {
     cat("Correct on the animal but not the cheese");
  }
  else if ( favCheese == "Muenster" ||
            favCheese == "Provolone" ) # 3rd test (cheese only)
  {
     cat("Correct on the cheese but not the animal");
  }
  else # all 3 tests returned FALSE (so, neither animal nor cheese)
  {
     cat("You have failed this test");
  }
}
```



```
{
  rm(list=ls());  options(show.error.locations = TRUE);

  favAnimal = readline("What is your favorite animal? ");
  favCheese = readline("What is your favorite cheese? ");

  if ( ( favAnimal == "Llama" ||
         favAnimal == "Alpaca" ||
         favAnimal == "Guanaco" ) &&
       ( favCheese == "Muenster" ||
         favCheese == "Provolone" ) )
  {
    cat("You are truly a wise human being!");
  }
  else if ( favAnimal == "Llama" ||
            favAnimal == "Alpaca" ||
            favAnimal == "Guanaco" )
  {
    cat("Correct on the animal but not the cheese");
  }
  else if ( favCheese == "Muenster" ||
            favCheese == "Provolone" )
  {
    cat("Correct on the cheese but not the animal");
  }
  else # wrong on both animal and cheese
  {
    cat("You have failed this test");
  }
}
```

Values

favAnim...	"Sheep"
favChee...	"Cheddar"

```
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is your favorite animal? Llama
What is your favorite cheese? Swiss
Correct on the animal but not the cheese
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is your favorite animal? Goat
What is your favorite cheese? Provolone
Correct on the cheese but not the animal
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is your favorite animal? Sheep
What is your favorite cheese? Cheddar
You have failed this test
> |
```

*Fig 3: **If-else-if** structure used to check for multiple **favAnimal/favCheese** possibilities*

6 - Multiple conditions and ranges

Our third example connects outer conditional operations using an ***or (||)*** instead of an ***and (&&):***

```
if ( weather is rainy or windy or the
    temperature is less than 40 or greater than 90 )
```

First, let's separate the variables ***weather*** and ***temperature*** using parentheses:

```
if ( ( weather is rainy or windy ) or
_____( the temperature is less than 40 or greater than 90 ) )
```

Now, we are saying if either the ***weather*** or ***temperature*** inner conditional statement is ***TRUE*** then the whole conditional statement is ***TRUE*** -- this is an (**||**) operation:

```
if ( ( weather is rainy or windy ) ||
     ( the temperature is less than 40 or greater than 90 ) )
```

Let's be explicit with the conditional statements in each inner parentheses:

```
if ( ( weather is rainy or weather is windy ) ||
     ( the temperature is less than 40 or temperature is greater than 90 ) )
```

And replace "is" with the is equal to (==) operator, "less than" with (**<**), and "greater than" with (**>**)

```
if ( ( weather == "rainy" or weather == "windy" ) ||
     ( the temperature < 40 or temperature > 90 ) )
```

And, finally replace the ***or*** connection the ***weather*** and ***temperature*** conditions with (||)

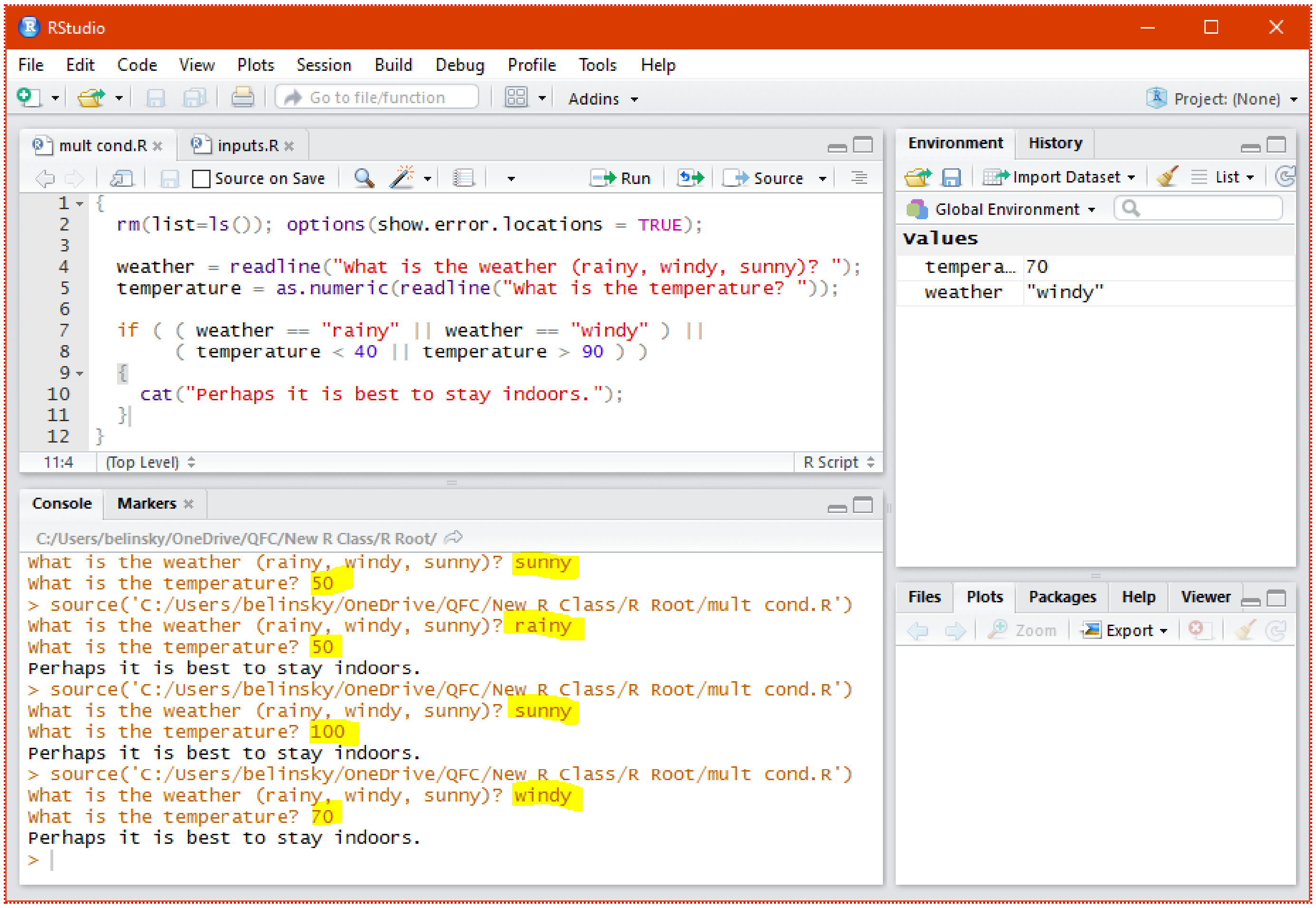
```
if ( ( weather == "rainy" || weather == "windy" ) ||
     ( the temperature < 40 || temperature > 90 ) )
```

```
{
  rm(list=ls()); options(show.error.locations = TRUE);

  weather = readline("What is the weather (rainy, windy, sunny)? ");
  temperature = as.numeric(readline("What is the temperature? "));

  if ( ( weather == "rainy" || weather == "windy" ) ||
       ( temperature < 40 || temperature > 90 ) )
  {
     cat("Perhaps it is best to stay indoors.");
  }
}
```

In the previous examples, both inner conditional operations had to be ***TRUE*** for the whole conditional statements to be ***TRUE***.
In this example if *either* inner conditional operation (***weather*** or ***temperature***) evaluates to ***TRUE*** then the whole conditional statement is ***TRUE***.



```
{
  rm(list=ls()); options(show.error.locations = TRUE);

  weather = readline("What is the weather (rainy, windy, sunny)? ");
  temperature = as.numeric(readline("What is the temperature? "));

  if ( ( weather == "rainy" || weather == "windy" ) ||
       ( temperature < 40 || temperature > 90 ) )
  {
    cat("Perhaps it is best to stay indoors.");
  }
}
```

```
Values
tempera… 70
weather  "windy"
```

```
C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/
What is the weather (rainy, windy, sunny)? sunny
What is the temperature? 50
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is the weather (rainy, windy, sunny)? rainy
What is the temperature? 50
Perhaps it is best to stay indoors.
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is the weather (rainy, windy, sunny)? sunny
What is the temperature? 100
Perhaps it is best to stay indoors.
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is the weather (rainy, windy, sunny)? windy
What is the temperature? 70
Perhaps it is best to stay indoors.
> |
```

*Fig 4: Checking the **weather** and **temperature** range*

6.1 - Interpreting a FALSE condition

If the conditional statement in the previous example (*Fig.4*) evaluates to ***TRUE***, we know that at least one of these conditions is ***TRUE***:

1. Weather is rainy
2. Weather is windy
3. Temperature is less than 40
4. Temperature is greater than 90

What do we know if the conditional statement evaluates to ***FALSE***? Well there are two things:

1. The weather is not rainy nor windy.
   So, *the weather must be sunny* (assuming the user only had those three choices).
2. The temperature is not less than 40 nor greater than 90.
   So *the temperature must be between 40 and 90*.

Let's create an ***if-else-if*** structure that takes advantage of the fact that we know the *weather must be sunny* if the conditional statement evaluates to ***FALSE***.

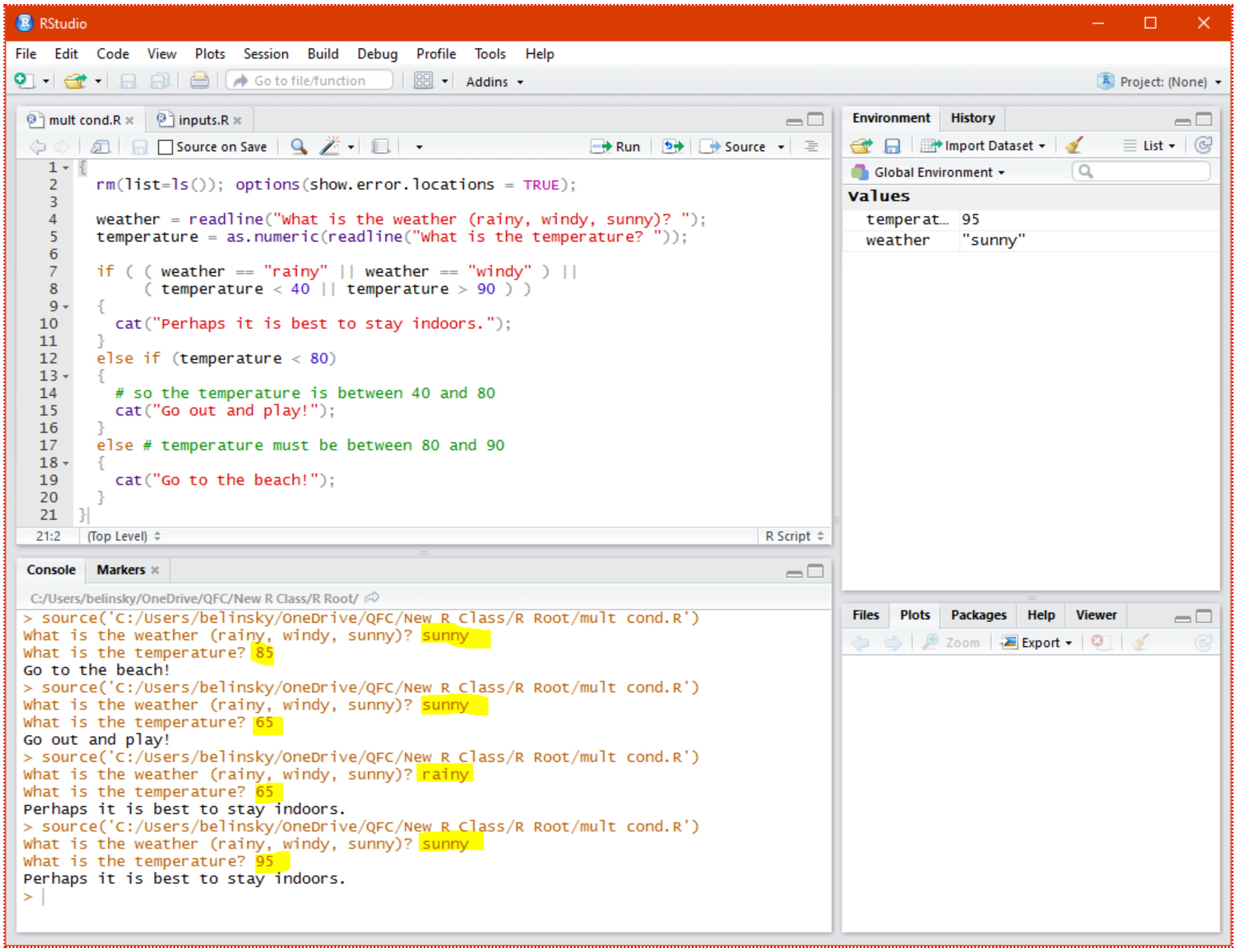
Using an ***if-else-if*** structure, we will create a second test that checks if temperatures are less than 80. If this is ***TRUE*** then the *weather must be sunny* and the *temperature must be between 40 and 80*.

If the first two tests fail, the weather must be sunny and the *temperature must be between 80 and 90*.

```r
{
  rm(list=ls()); options(show.error.locations = TRUE);

  weather = readline("What is the weather (rainy, windy, sunny)? ");
  temperature = as.numeric(readline("What is the temperature? "));

  if ( ( weather == "rainy" || weather == "windy" ) ||
       ( temperature < 40 || temperature > 90 ) )      # first test
  {
     cat("Perhaps it is best to stay indoors.");
  }
  else if (temperature < 80)                           # second test
  {
    # we know the temperature is between 40 and 90 and it is sunny
    cat("Go out and play!");
  }
  else # temperature must be between 80 and 90, weather must be sunny
  {
    cat("Go to the beach!");
  }
}
```


```r
{
  rm(list=ls());  options(show.error.locations = TRUE);

  weather = readline("what is the weather (rainy, windy, sunny)? ");
  temperature = as.numeric(readline("what is the temperature? "));

  if ( ( weather == "rainy" || weather == "windy" ) ||
       ( temperature < 40 || temperature > 90 ) )
  {
    cat("Perhaps it is best to stay indoors.");
  }
  else if (temperature < 80)
  {
    # so the temperature is between 40 and 80
    cat("Go out and play!");
  }
  else # temperature must be between 80 and 90
  {
    cat("Go to the beach!");
  }
}
```

```
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is the weather (rainy, windy, sunny)? sunny
What is the temperature? 85
Go to the beach!
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is the weather (rainy, windy, sunny)? sunny
What is the temperature? 65
Go out and play!
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is the weather (rainy, windy, sunny)? rainy
What is the temperature? 65
Perhaps it is best to stay indoors.
> source('C:/Users/belinsky/OneDrive/QFC/New R Class/R Root/mult cond.R')
What is the weather (rainy, windy, sunny)? sunny
What is the temperature? 95
Perhaps it is best to stay indoors.
>
```

```
Values
  temperat…   95
  weather     "sunny"
```

*Fig 5: Using **if-else-if** to check for multiple **weather/temperature** conditions*

7 - Application

If you have any questions regarding this application, feel free to email them to the instructor here.
You can attach files to the email above or send the whole Root Folder as a zipped file. Instructions for zipping the Root Folder are here.

A) Have a user enter values for:
   1) The age of a fish
   2) The location that the fish was caught (north or south)

B) Categorize the fish based on location and age:
   1) All fish caught in Northport between the ages of 3 and 5 are in category "I"
   2) All fish caught in Southport between the ages of 2 and 6 are in category "II"
   3) All fish too old to be in categories "A" or "B" are in category "III"
   4) All fish too young to be in categories "A" or "B" are in category "IV"

C) Create an error condition for the previous script that gives an "invalid port" message if the port is not North or South. This fish should not be categorized.

*Save you script file as **app1-10.r** in the **scripts** folder of your RStudio Project for the class.*