Dhan’s Answers:

# Week 2 Quiz

Passed

9/10 questions correct

Quiz passed!

[Continue Course](https://www.coursera.org/learn/r-programming/supplement/amLgW/programming-assignment-1-instructions-air-pollution)

[Back to Week 2](https://www.coursera.org/learn/r-programming/home/week/2)

Correct

1.

Suppose I define the following function in R

cube <- function(x, n) {

x^3

}

What is the result of running

cube(3)

in R after defining this function?



The number 27 is returned

**Well done!**

Because 'n' is not evaluated, it is not needed even though it is a formal argument.



An error is returned because 'n' is not specified in the call to 'cube'



The users is prompted to specify the value of 'n'.



A warning is given with no value returned.

Correct

2.

The following code will produce a warning in R.

x <- 1:10

if(x > 5) {

x <- 0

}

Why?



You cannot set 'x' to be 0 because 'x' is a vector and 0 is a scalar.



'x' is a vector of length 10 and 'if' can only test a single logical statement.

**Well done!**



There are no elements in 'x' that are greater than 5



The expression uses curly braces.



The syntax of this R expression is incorrect.

Correct

3.

Consider the following function

f <- function(x) {

g <- function(y) {

y + z

}

z <- 4

x + g(x)

}

If I then run in R

z <- 10

f(3)

What value is returned?



7



10

**Well done!**



4



16

Correct

4.

Consider the following expression:

x <- 5

y <- if(x < 3) {

NA

} else {

10

}

What is the value of 'y' after evaluating this expression?



NA



3



5



10

**Well done!**

Correct

5.

Consider the following R function

h <- function(x, y = NULL, d = 3L) {

z <- cbind(x, d)

if(!is.null(y))

z <- z + y

else

z <- z + f

g <- x + y / z

if(d == 3L)

return(g)

g <- g + 10

g

}

Which symbol in the above function is a free variable?



f

**Well done!**



z



d



L



g

Correct

6.

What is an environment in R?



a list whose elements are all functions



a special type of function



a collection of symbol/value pairs

**Well done!**



an R package that only contains data

Correct

7.

The R language uses what type of scoping rule for resolving free variables?



lexical scoping

**Well done!**



compilation scoping



global scoping



dynamic scoping

Incorrect

8.

How are free variables in R functions resolved?



The values of free variables are searched for in the environment in which the function was defined



The values of free variables are searched for in the working directory



The values of free variables are searched for in the global environment



The values of free variables are searched for in the environment in which the function was called

**Sorry, that's incorrect.**

Correct

9.

What is one of the consequences of the scoping rules used in R?



R objects cannot be larger than 100 MB



All objects must be stored in memory

**Well done!**



Functions cannot be nested



All objects can be stored on the disk

Correct

10.

In R, what is the parent frame?



It is always the global environment



It is the environment in which a function was called

**Well done!**



It is the environment in which a function was defined



It is the package search list

<https://github.com/cynthia0611/Coursera-R-Programming/blob/master/Week-2/Quiz%202.md>

**Question 1**

Suppose I define the following function in R

cube <- function(x, n) {

x^3

}

What is the result of running

cube(3)

in R after defining this function? Your Answer Score Explanation An error is returned because 'n' is not specified in the call to 'cube'  
**The number 27 is returned Correct 1.00 Because 'n' is not evaluated, it is not needed even though it is a formal argument.** The users is prompted to specify the value of 'n'.  
A warning is given with no value returned.

**Question 2**

The following code will produce a warning in R.

x <- 1:10

if(x > 5) {

x <- 0

}

Why?

The expression uses curly braces.  
You cannot set 'x' to be 0 because 'x' is a vector and 0 is a scalar.  
**'x' is a vector of length 10 and 'if' can only test a single logical statement. Correct 1.00**  
There are no elements in 'x' that are greater than 5  
The syntax of this R expression is incorrect.

**Question 3**

Consider the following function

f <- function(x) {

g <- function(y) {

y + z

}

z <- 4

x + g(x)

}

If I then run in R

z <- 10

f(3)

What value is returned?

**10 Correct 1.00**  
7  
4  
16

**Question 4**

Consider the following expression:

x <- 5

y <- if(x < 3) {

NA

} else {

10

}

What is the value of 'y' after evaluating this expression?

NA  
**10 Correct 1.00**  
5  
3

**Question 5**

Consider the following R function

h <- function(x, y = NULL, d = 3L) {

z <- cbind(x, d)

if(!is.null(y))

z <- z + y

else

z <- z + f

g <- x + y / z

if(d == 3L)

return(g)

g <- g + 10

g

}

Which symbol in the above function is a free variable?

**f**  
z  
d  
L  
g

**Question 6**

What is an environment in R?

a list whose elements are all functions  
an R package that only contains data  
a special type of function  
**a collection of symbol/value pairs Correct 1.00**

**Question 7**

The R language uses what type of scoping rule for resolving free variables?

dynamic scoping  
global scoping  
compilation scoping  
**lexical scoping Correct 1.00**

**Question 8**

How are free variables in R functions resolved?

The values of free variables are searched for in the working directory  
The values of free variables are searched for in the environment in which the function was called  
The values of free variables are searched for in the global environment  
**The values of free variables are searched for in the environment in which the function was defined Correct 1.00**

**Question 9**

What is one of the consequences of the scoping rules used in R?

All objects can be stored on the disk  
**All objects must be stored in memory Correct 1.00**  
R objects cannot be larger than 100 MB  
Functions cannot be nested

**Question 10**

In R, what is the parent frame?

It is the package search list  
It is the environment in which a function was defined Inorrect 0.00  
**It is the environment in which a function was called**  
It is always the global environment

<https://github.com/Xiaodan/Coursera-R-Programming/blob/master/week2/week2.R>

62 lines (59 sloc)  1.09 KB

|  |  |
| --- | --- |
|  | # Quiz 2 |
|  | # Problem 1. |
|  | cube <- function(x, n){ |
|  | x^3 |
|  | } |
|  | cube(3) # 27 |
|  | # Problem 2. |
|  | x <- 1:10 |
|  | if(x > 5) { |
|  | x <- 0 |
|  | } # warning |
|  | # Problem 3. |
|  | f <- function(x) { |
|  | g <- function(y) { |
|  | y + z |
|  | } |
|  | z <- 4 |
|  | x + g(x) |
|  | } |
|  | z <- 10 |
|  | f(3) # 10 |
|  | # Problem 4. |
|  | x <- 5 |
|  | y <- if(x < 3) { |
|  | NA |
|  | } else { |
|  | 10 |
|  | } |
|  | y # 10 |
|  | # Problem 5. |
|  | h <- function(x, y = NULL, d = 3L) { |
|  | z <- cbind(x, d) |
|  | if(!is.null(y)) |
|  | z <- z + y |
|  | else |
|  | z <- z + f |
|  | g <- x + y / z |
|  | if(d == 3L) |
|  | return(g) |
|  | g <- g + 10 |
|  | g |
|  | } |
|  | # free variable: f |
|  | f <- function(x, y) { |
|  | x^2 + y / z |
|  | } # free variable: z |
|  | # Problem 6. |
|  | # environment: a collection of symbol/value pair. |
|  | # Problem 7. |
|  | # type of scoping rule for resolving free variables: Lexical scoping |
|  | # Problem 8. |
|  | # free variables in R functions are resolved by: |
|  | # The values of free variables are searched for in the environment |
|  | # in which the function was defined. |
|  | # Problem 9. |
|  | # one of the consequences of the scoping rules used in R: |
|  | # All objects must be stored in memory. |
|  | # Problem 10. |
|  | # parent theme: it is the environment in which a function was called. |
|  |  |