CSC 121H1 S 2018 Quiz 1 (Version A)
February 12, 2018
Duration — 35 minutes
Aids allowed: none

UTORid: ⌊_⌊_⌊_⌊_⌊_⌊_⌊_⌊_⌋

Last Name: ____________________ **First Name:** ____________________

Lecture Section: L0101 (MWF12)
Instructor: Mark Kazakevich

*Do **not** turn this page until you have received the signal to start.*
(Please fill out the identification section above, **write your name on the back of the test**, and read the instructions below.)
Good Luck!

This quiz is double-sided, and consists of 4 questions and a list of built-in R function descriptions. *When you receive the signal to start, please make sure that your copy is complete.*

- Read each question **carefully**.
- Comments are not required except where indicated, although they may help us mark your answers.
- No error checking is required: assume all user input and all argument values are valid.
- If you use any space for rough work, indicate clearly what you want marked.
- Do not remove any pages from the quiz booklet.
- You may use a pencil; however, work written in pencil will not be considered for remarking.
- In some cases, part marks will be awarded for partial answers.

# 1:	______/ 8
# 2:	______/ 4
# 3:	______/ 5
# 4:	______/ 5
TOTAL:	______/22

Question 1. [8 marks]

Beside each set of commands below, write what R will output after those commands are typed into the R console. For each set of commands, only write the output of the **last command**. Note that the ">" shown at the beginnings of each line is the R command prompt, not something typed.
Write your answer after the [1].

Code	Output
`> 20 / 5 + 3 * 2`	[1]

`> z <- c(1.3, 4, 6.7)`
`> typeof(z)`
	[1]

`> k <- c("d", "e", "f")`
`> k[3] <- "csc"`
`> k`
	[1]

`> v <- c(2, 4, 6, 8, 10)`
`> v <- c(v[c(2:4)], 33)`
`> v`
	[1]

`> a <- 4 * 2`
`> t <- 3 < a`
`> t`
	[1]

`> as.character(4 + 5)`	[1]

`> g <- c(8, 3)`
`> g %% 2 == 0`
	[1]

`> d <- c(1, 2, 3)`
`> d[-1]`
`> d`
	[1]

Question 2. [4 marks]

Consider the following function, MysteryFunc.

```
MysteryFunc <- function(a, b) {

  p <- 0
  v <- numeric(0)
  while (p < 3) {
    v <- c(v, a, b)
    p <- p + 1
  }

  return(v)

}
```

Part A: What is the R console output of the following function calls? Put the result in the box after each function call, directly after the [1].
(*Hint:* Step through the code with the arguments and figure out what is happening to the variables as the function runs. Use the blank pages at the back for rough work if you want more room.)

```
> MysteryFunc(1, 2)
[1]
```

```
> MysteryFunc(3, 7)
[1]
```

Part B: Based on the above function calls and output, write a proper **docstring** in the box after the function header below that explains what the function MysteryFunc does. You do **not** need to write the # symbols at the beginning of each line.

```
MysteryFunc <- function(a, b) {
```

Write your docstring here

```
  .
  # rest of function
  .
 }
```

Question 3. [5 marks]

The follwing are some test cases for the function `SeeingDouble`

```
> SeeingDouble(c("hi", "hello"))
[1] "hihi"  "hellohello"
> SeeingDouble(c("a", "b", "c"))
[1] "aa" "bb" "cc"
```

Complete the function body of the function `SeeingDouble` according to its docstring below. Remember that you can put two strings together using the `paste` function, which is also described on the back of the quiz if you need a reminder. Do not use any functions we haven't talked about in class.

You **must use a loop** of some sort to do the work for this function.

```
SeeingDouble <- function(v) {
  # Returns a copy of string vector 'v', except the string at every index
  # has been doubled.
  #
  # Precondition: length(v) > 0











}
```

Question 4. [5 marks]

Below is a table that gives the amount of time it takes to prepare an order of cupcakes, depending on the amount of cupcakes ordered.

number of cupcakes	preparation time
less than 5 cupcakes	40 minutes
5 to 7 cupcakes	60 minutes
over 7 cupcakes	90 minutes

If the customer wants icing on their cupcakes, an **extra** 15 minutes is added.

The follwing are some test cases for the function `PreparationTime`.

```
> PreparationTime(3, FALSE)
[1] 40
> PreparationTime(6, TRUE)
[1] 75
> PreparationTime(10, FALSE)
[1] 90
```

On the **next page**, complete the function body of the function `PreparationTime` according to its docstring. Do not use any functions we haven't talked about in class.

Continued on next page...

```
PreparationTime <- function(numCakes, wantIcing) {
  # Returns the time it takes to prepare 'numCakes' number of cupcakes.
  # 'wantIcing' is a logical value that indicates whether or not
  # the customer wants icing.
  #
  # Precondition: numCakes > 0
















}
```

[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]

R built-in function descriptions

```
paste(s1, s2, sep = s3)
   Returns a new string that is s1 followed by s2, with s3 as the separation character.
   s3 is a space character: " " by default if sep is not specified.
as.double(x)
   Returns a numeric representation of 'x' that is of type 'double'
as.integer(x)
   Returns a numeric representation of 'x' that is of type 'integer'
as.character(x)
   Returns a string representation of 'x' that is of type 'character'.
is.double(x)
   Returns TRUE if and only if x is of type 'double'.
is.integer(x)
   Returns TRUE if and only if x is of type 'integer'.
is.character(x)
   Returns TRUE if and only if x is of type 'character'.
typeof(x)
   Returns a string that identifies the type of the value of x.
```

End of Quiz