NUI Galway
OÉ Gaillimh

Semester 1 Examinations 2019 / 2020

Exam Code(s)	1CSD1, 1CSD2
Exam(s)	M.Sc. in Computer Science (Data Analytics)
Module Code(s)	CT5102
Module(s)	Programming for Data Analytics
Paper No.	I
External Examiner(s)	Professor P. L. Lanzi
Internal Examiner(s)	Professor Michael Madden *Dr. Jim Duggan

Instructions: Answer any 3 questions. All questions carry equal marks.

Duration	2hrs
No. of Pages	7 (including cover page)
Department(s)	School of Computer Science

Requirements

1. (a) Consider the following code snippet:

```
library(dplyr)
library(ggplot2)
library(tidyr)
```

When the code is loaded into R, draw a diagram of the environments in the search path, and include globalenv(), baseenv() and emptyenv(). Explain each environments position in the hierarchy.

[4]

(b) What does the following code return? Explain the mechanism by which the value is calculated.

```
f1 <- function (x){
  function (y){
    y^x
  }
}

f1(2)(4)
```

[4]

(c) What is the value of x when the function f1(100) is called. Explain your answer.

```
x <- 100

f1 <- function(x1){
  f2 <- function(x2){
    x <<- x1 + x2
  }
  f2(10)
}

Y <- f1(100)
```

[4]

(d) Implement a closure that maintains a counter. The state variable starts with a value of 0. The functions are:

Function Name	**Details**
increment(n)	Increments the counter by n, default =1
decrement(n)	Decrements the counter by n, default =1
reset(n)	Resets the counter to n, default value =0
get_state()	Returns the current counter value

[13]

2. (a) Describe each of the following dplyr functions: **select(), mutate(),** and **semi_join().**

[5]

(b) Consider the following tibble **st** (students) and **res** (results).

st

```
> st
# A tibble: 4 x 2
     ID Name
  <dbl> <chr>
1    10 J. Murphy
2    20 A. Sanchez
3    30 F. Mueller
4    40 P. Rock
```

res

```
> res
# A tibble: 5 x 4
     ID Module Title       Result
  <dbl> <chr>  <chr>        <dbl>
1    10 CX101  Programming     70
2    10 CX102  Databases       67
3    20 CX102  Databases       86
4    40 CX102  Databases       58
5    50 CX102  Databases       71
```

Identify the relevant **dplyr** calls that generate the following tables.

(1) Joining, by = "ID"

```
# A tibble: 6 x 5
     ID Name       Module Title       Result
  <dbl> <chr>      <chr>  <chr>        <dbl>
1    10 J. Murphy  CX101  Programming     70
2    10 J. Murphy  CX102  Databases       67
3    20 A. Sanchez CX102  Databases       86
4    30 F. Mueller NA     NA              NA
5    40 P. Rock    CX102  Databases       58
6    50 NA         CX102  Databases       71
```

(2) Joining, by = "ID"

```
# A tibble: 1 x 2
     ID Name
  <dbl> <chr>
1    30 F. Mueller
```

(3) Joining, by = "ID"

```
# A tibble: 4 x 5
     ID Name       Module Title       Result
  <dbl> <chr>      <chr>  <chr>        <dbl>
1    10 J. Murphy  CX101  Programming     70
2    10 J. Murphy  CX102  Databases       67
3    20 A. Sanchez CX102  Databases       86
4    40 P. Rock    CX102  Databases       58
```

(4) Joining, by = "ID"

```
# A tibble: 1 x 4
     ID Module Title     Result
  <dbl> <chr>  <chr>      <dbl>
1    50 CX102  Databases     71
```

[10]

(c) Consider the following tibble (called table4b) which shows population data for a number of countries.

```
> table4b
# A tibble: 3 x 3
  country          `1999`      `2000`
* <chr>             <int>       <int>
1 Afghanistan   19987071   20595360
2 Brazil       172006362  174504898
3 China       1272915272 1280428583
```

Convert this to tidy data format, using **gather(),** with the following result:

```
> td
# A tibble: 6 x 3
  country     Year  Population
  <chr>       <chr>      <int>
1 Afghanistan 1999    19987071
2 Brazil      1999   172006362
3 China       1999  1272915272
4 Afghanistan 2000    20595360
5 Brazil      2000   174504898
6 China       2000  1280428583
```

Next, generate the following summary table.

```
# A tibble: 2 x 5
  Year      MaxPop CountryMax   MinPop CountryMin
  <chr>      <int> <chr>         <int> <chr>
1 1999  1272915272 China      19987071 Afghanistan
2 2000  1280428583 China      20595360 Afghanistan
```

Finally, produce the following tibble which shows the growth percentages for the three countries. Note, any row with an NA value must be filtered out.

```
# A tibble: 3 x 6
# Groups:   country [3]
  country     Year  Population   Previous   Delta PChange
  <chr>       <chr>      <int>      <int>   <int>   <dbl>
1 Afghanistan 2000    20595360   19987071  608289   3.04
2 Brazil      2000   174504898  172006362 2498536   1.45
3 China       2000  1280428583 1272915272 7513311   0.590
```

[10]

(3) (a) What is meant by *generic function object oriented systems*, and explain the following output. Show how to represent this information on a diagram.

```
> methods("mean")
[1] mean.Date        mean.default   mean.difftime  mean.POSIXct
mean.POSIXlt
```

[5]

(b) Write the equivalent of this code into a function called **my_class()**, making use of the **structure()** function.

```
x <- list(a=1:3,b=4:7)
class(x) <- "my_class"
```

[5]

(c) Consider the following function call to **lm()** – using the old faithful data set - and explain the results of the function call **attributes(mod).**

```
> mod <- lm(eruptions ~ waiting, data=faithful)
> attributes(mod)
$names
 [1] "coefficients"  "residuals"     "effects"
 [4] "rank"          "fitted.values" "assign"
 [7] "qr"            "df.residual"   "xlevels"
[10] "call"          "terms"         "model"
$class
[1] "lm"
```

[5]

(d) The function call **`coef(mod)`** returns the following information.

```
> coef(mod)
(Intercept)     waiting
-1.87401599  0.07562795
```

The function is defined as follows in R. Comment on what kind of function this is.

```
> coef
function (object, ...)
UseMethod("coef")
```

[5]

(e) Using one line of code, change the class of mod to "my_lm" (it should also inherit from lm), and write a new class function that will behave as follows.

```
> coef(mod)
Welcome to coef() for the class my_lm
Here is the output from coef() for the class lm
(Intercept)     waiting
-1.87401599  0.07562795
```

[5]

(4) (a) Given the following data frame, show (and explain) the results (including the type of the result) of the following commands.

```
> df
      country year  cases population
1 Afghanistan 1999    745   19987071
2 Afghanistan 2000   2666   20595360
3      Brazil 1999  37737  172006362
4      Brazil 2000  80488  174504898
5       China 1999 212258 1272915272
6       China 2000 213766 1280428583
```

- `df[c(T,F),3:4]`
- `df[df$country=="Brazil",c("population")]`
- `df[df$country=="China",c("population"),drop=F]`

[5]

(b) Given the following vector m:

```
> m
[1] 1 2 3 4 5 6
```

Show how this vector can be transformed to the following, using the attr() function, and the relevant matrix naming functions.

```
> m
      Col 1 Col 2 Col 3
Row 1     1     3     5
Row 2     2     4     6
```

[5]

(c) Visualise the following list.

```
l <- list(a=1:2,b="Hello",c=list(d=3:4,e=c(T,F)))
```

Draw a representation of the results returned by the following commands, and explain each solution.

```
l[1:2]

l[[2]]

l[[3]]

l[[3]][1:2]

l[[3]][[1]][2]
```

[5]

(d) Write a function *my_table(v)* that takes in a vector of whole numbers, and returns a frequency count for each element. For example:

```
> my_table(c(1,1,2,3,4,4,5,5))

1 2 3 4 5
2 1 1 2 2
```

[5]

(e) Given the following data frame df, use **lapply()** to transform the data frame so that all values above 40 are set to NA.

```
> df
   C1 C2 C3 C4 C5
1  23  6 25 16 11
2  35 13 15 25  2
3  42 12 28 11 42
4  38 17 23 47 48
5  19 20 12 10 18
6  11 36 49 40 18
7  26  8 18 15  6
8  24 34 42 31 11
9  48 20 19 39 26
10  9  8  3  3  7
```

```
> df_t
   C1 C2 C3 C4 C5
1  23  6 25 16 11
2  35 13 15 25  2
3  NA 12 28 11 NA
4  38 17 23 NA NA
5  19 20 12 10 18
6  11 36 NA 40 18
7  26  8 18 15  6
8  24 34 NA 31 11
9  NA 20 19 39 26
10  9  8  3  3  7
```

[5]