Semester 1 Examinations 2021 / 2022

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)	Dr. John Woodward
Internal Examiner(s)	Professor Michael Madden *Prof. Jim Duggan
Instructions:	Answer any 3 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	7 (including cover page)
Department(s)	School of Computer Science

Requirements		
	Release in Exam Venue	Yes [X] No [ ]
	MCQ Answersheet	Yes [ ] No [X]
	Handout	None
	Statistical/ Log Tables	None
	Cambridge Tables	None
	Graph Paper	None
	Log Graph Paper	None
	Other Materials	None
	Graphic material in colour	Yes [X] No [ ]

1. (a) Consider the following code snippet:

```
library(ggplot2)
library(dplyr)

sum <- function(x)base::sum(x[-1])
```

When the code is loaded into R, draw a diagram of the environments in the search path.

Explain what the call to sum(1:5) will return and why it will return that value.

[5]

(b) Visualise the following code, and show the result of the call to f1(). Explain the mechanism by which the value is calculated.

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

y <- f1(3)(2)
```

[8]

(c) Implement a closure that acts as a stock counter (SKU)

Create a counter as follows, the initial number of stock items is 10.

```
st = SKU(10)
```

Function Name	**Details**
increment(n)	Increases the SKU amount by n
decrement(n)	Decreases the SKU amount by n
get_stock()	Returns the current number of stock items.

Visualise the resulting closure state after a call to increment (10)

[12]

2. (a) The following are two tables (t1, t2).

```
> t1
# A tibble: 5 x 2
  StudentID Name
      <int> <chr>
1         1 AA
2         2 BB
3         3 CC
4         4 DD
5         5 EE
```

```
> t2
# A tibble: 5 x 3
  StudentID Subject Grade
      <int> <chr>   <int>
1         1 CX101      63
2         3 CX101      91
3         1 CX103      77
4         3 CX101      87
5         3 CX102      83
```

Show what **dplyr** functions can be used to create the following results, and explain how the process works.

```
> r1
# A tibble: 2 x 2
  StudentID Name
      <int> <chr>
1         1 AA
2         3 CC
```

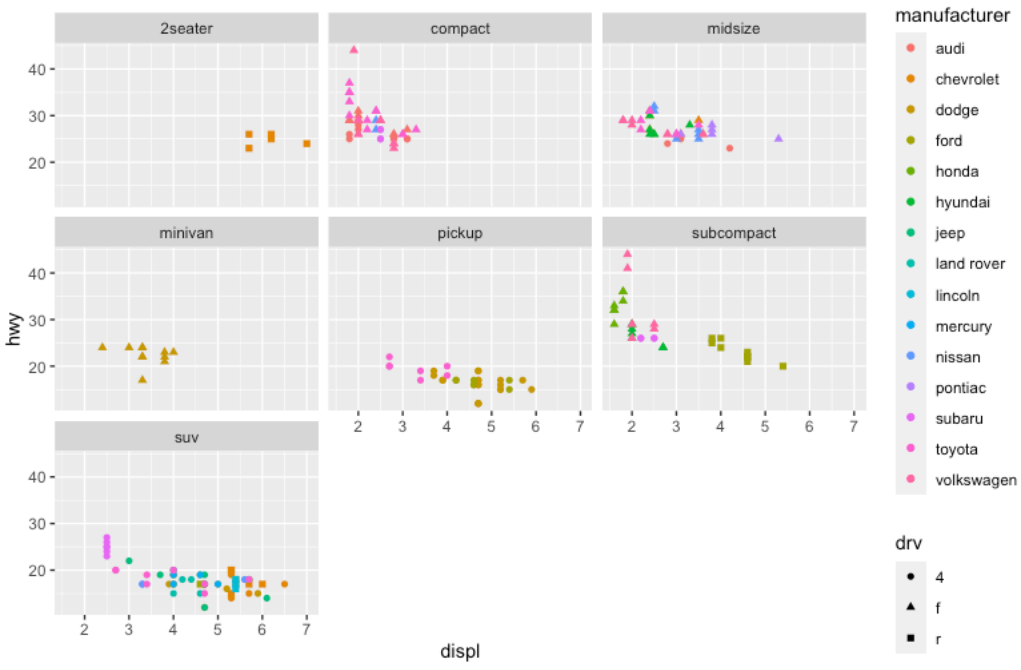
```
> r2
# A tibble: 3 x 2
  StudentID Name
      <int> <chr>
1         2 BB
2         4 DD
3         5 EE
```

[4]

(b) Summarise the main idea behind exploratory data analysis.

Based on the mpg data set, show the code that generates the following plot. The variables used include displ, hwy, manufacturer, class and drv.

[6]



(c) Perform the following analysis, based on the tibbles observations and stations (aimsir17) – first 3 observations for each is shown.

```
> observations
# A tibble: 219,000 x 12
   station  year month   day  hour date                 rain
   <chr>   <dbl> <dbl> <int> <int> <dttm>              <dbl>
 1 ATHENRY  2017     1     1     0 2017-01-01 00:00:00   0
 2 ATHENRY  2017     1     1     1 2017-01-01 01:00:00   0
 3 ATHENRY  2017     1     1     2 2017-01-01 02:00:00   0
# … with 218,990 more rows, and 5 more variables:
#   temp <dbl>, rhum <dbl>, msl <dbl>, wdsp <dbl>,
#   wddir <dbl>

> stations
# A tibble: 25 x 5
   station        county    height latitude longitude
   <chr>          <chr>      <dbl>    <dbl>     <dbl>
 1 ATHENRY        Galway        40     53.3     -8.79
 2 BALLYHAISE     Cavan         78     54.1     -7.31
 3 BELMULLET      Mayo           9     54.2    -10.0
```

(a) Calculate the total monthly rainfall for each station

```
> arrange(month_r,month,station)
# A tibble: 300 x 3
   station        month TotalRain
   <chr>          <dbl>     <dbl>
 1 ATHENRY            1      47.4
 2 BALLYHAISE         1      32.8
 3 BELMULLET          1      60
```

(b) Add the county information to each monthly summary.

```
> month_c
# A tibble: 300 x 4
   station        month TotalRain county
   <chr>          <dbl>     <dbl> <chr>
 1 ATHENRY            1      47.4 Galway
 2 BALLYHAISE         1      32.8 Cavan
 3 BELMULLET          1      60   Mayo
 4 CASEMENT           1      27.6 Dublin
```

(c) Show the county averages (by month) is descending order of rainfall.

```
> county_avr
# A tibble: 180 x 3
   month county  MeanRain
   <dbl> <chr>      <dbl>
 1     9 Kerry       204.
 2    12 Kerry       199.
 3     9 Sligo       174.
 4     1 Kerry       169.
 5     8 Donegal     168.
 6    10 Kerry       162.
```

[15]

3. (a) Describe the key differences between the S3 class system and message-passing OO systems such as Java and C++. Show an example of this using the generic function summary() from Base R. Show how you could extend the function summary() so that it generates the following output from a linear model result (output is based on a call to the Base R function coef())

```
mod <- lm(hwy~displ,data=mpg)
# Add extra step here
> class(mod)
[1] "my_lm" "lm"

> summary(mod)
Summary Coefficients
(Intercept)       displ
  35.697651   -3.530589
```

[10]

(b) Consider the following R output (the tibble stations is from aimsir17).

```
stat_test <- stations

class(stat_test)
[1] "tbl_df"     "tbl"        "data.frame"
>
> head(stat_test)
# A tibble: 6 x 5
  station      county height latitude longitude
  <chr>        <chr>   <dbl>    <dbl>     <dbl>
1 ATHENRY      Galway     40     53.3     -8.79
2 BALLYHAISE   Cavan      78     54.1     -7.31
3 BELMULLET    Mayo        9     54.2    -10.0
4 CASEMENT     Dublin     91     53.3     -6.44
5 CLAREMORRIS  Mayo        68     53.7     -8.99
6 CORK AIRPORT Cork      155     51.8     -8.49
```

Using your knowledge of S3, and in particular, how new classes can inherit from existing classes, modify stat_test so that it has the following structure.

```
> class(stat_test)
[1] "my_s"       "tbl_df"     "tbl"        "data.frame"
```

Then, create versions of head() and tail() that return the first three and last three observations in the modified stations tibble. For example, the function call to head() should return:

```
> head(stat_test)
# A tibble: 3 x 5
  station    county height latitude longitude
  <chr>      <chr>   <dbl>    <dbl>     <dbl>
1 ATHENRY    Galway     40     53.3     -8.79
2 BALLYHAISE Cavan      78     54.1     -7.31
3 BELMULLET  Mayo        9     54.2    -10.0
```

[15]

4. (a) Describe the following functions from the package purrr.

- **`map(.x, .f)`**
- **`map_dbl(.x, .f)`**

[4]

(b) Describe the two ways of defining an anonymous function using purrr.

Show how the tilde-dot shorthand notation can be used to generate values for the equation $\mathbf{y = 3x^2 - 10x + 100}$, assuming an input range of [-100, +100], in steps of 0.1

[4]

(c) Explain the benefit of the **`group_split()`** function and why it can be used with purrr functions. Show how the **`group_split()`** function could be used, along with a relevant **`map_*`** function, to generate the following summary from the mpg tibble.

```
   NObs CarClass   AvrHwy AvrCty
  <int> <chr>       <dbl>  <dbl>
1     5 2seater      24.8   15.4
2    47 compact      28.3   20.1
3    41 midsize      27.3   18.8
4    11 minivan      22.4   15.8
5    33 pickup       16.9   13
6    35 subcompact   28.1   20.4
7    62 suv          18.1   13.5
```

[7]

(d) Describe the advantage of *nesting* a tibble, and demonstrate how the following two tibbles can be created, with the RSquared (r.squared) for the model hwy~displ shown in the second tibble.

```
# A tibble: 6 x 2
# Groups:   manufacturer [6]
  manufacturer data
  <chr>        <list>
1 audi         <tibble [18 × 10]>
2 chevrolet    <tibble [19 × 10]>
3 dodge        <tibble [37 × 10]>

# A tibble: 6 x 3
# Groups:   manufacturer [6]
  manufacturer data                RSquared
  <chr>        <list>                 <dbl>
1 audi         <tibble [18 × 10]>  0.486
2 chevrolet    <tibble [19 × 10]>  0.300
3 dodge        <tibble [37 × 10]>  0.486
```

[10]