Instructions:

- Your R code shall be written in a single script file.
- Name the script file `lab1_[your_id].R`. For example, `lab1_5123700044.R`.
- Separate your answers/code into sections according to `task_id` and `part_id`. For example,

```
# Task 1 part (a) ------------------------------------------------------------
tmp.df = data.frame( x = rnorm(10), y = rbinom(10, size = 1, prob = 0.3))

# Task 1 part (b) ------------------------------------------------------------
## A percentile is a measure used in statistics indicating the value below
## which a given percentage of observations in a group of observations fall.
## For example, the 20th percentile is the value (or score) below which
## 20% of the observations may be found.

# Task 1 part (c) ------------------------------------------------------------
my.func = function(x){t.test(x[[1]]~x[[2]])}

my.func(tmp.df)

.
.
.

# Task 2 part (a) ------------------------------------------------------------
cut(tmp.df$x, breaks = seq(-3, 3, length.out = 5))

rm(list = ls())
```

Task 1 (5 points)
This task is about R basics and being able to use the R environment productively.

(a) (1 point) Describe the difference between a single `?` and a double `??`.

```
> ?round
> ??regression
```

(b) (1 point) By default, R stores everything in double precision and prints 7 significant digits of numerical values, but you can ask R to print more by explicitly calling the `print` function. Create a double precision numerical variable called `x`, whose value is 1/7, then display 15 significant digits of `x`.

(c) (1 point) Study the output of the following R command. When should we use a coplot?

```
> demo(graphics)
```

(d) (1 point) What do the following R commands illustrate?

```
> .1 == 0.1
[1] TRUE
> .1 + .2 == .3
[1] FALSE
```

(e) (1 point) Write simple R expressions to generate a vector `y.vec` containing

$$1, -1, 2, -1, 3, -1, \ldots, 100, -1$$

Task 2 (14 points)
This task is about subsetting in R.

```
> z = sample(c(sample(-100:100, 27), rep(NA, 3)))
>
> tmp = runif(1, min = 3, max = 4)
>
> m.mat = matrix(c(1:4, tmp, 6L:9L), nrow = 3)
> colnames(m.mat) = c("A", "B", "A")
>
> m.df = data.frame(A = 1:3, B = c(4, tmp, 6L), A = 7L:9L)
```

(a) What expressions would you extract the following subsets from `z`?

i. (1 point) The first value of `z`.

ii. (1 point) The second through fifth values of `z`.

iii. (1 point) All values of `z` except for the last two. (Don't rely on `z` having any particular fixed length.)

iv. (1 point) The 2nd, 4th, 6th, etc. values of `z`

v. (1 point) All the positive values in `z`.

vi. (1 point) All the non-`NA` values in `z`.

vii. (1 point) Every third value of `z`, starting with the second.

(b) (1 point) What do the following R commands illustrate?

```
> z.named = setNames(z, state.name[1:length(z)])
> z.named[c("Michigan")]
```

(c) What expressions would you extract the following subsets from `m.mat`?

i. (1 point) The first two rows of `m.mat`.

ii. (1 point) All elements that are bigger than $\log_2(10)$ in the second column of `m.mat`.

(d) (1 point) What does the following R command illustrate?

```
> m.mat[1:9]
```

(e) What expressions would you extract the following subsets from `m.df`?

i. (1 point) The first two rows of `m.df`.

ii. (1 point) All elements that are bigger than $\log_2(10)$ in the second column of `m.df`.

(f) (1 point) What do the following R commands and their outputs illustrate?

```
> m.mat[1, 1] == m.df[1, 1]
> m.mat[,"B"] == m.df[,"B"]
> m.mat$B
> m.df$B
>
> colnames(m.mat); colnames(m.df);
>
> m.mat[, 3] == m.df[, 3]
> is.integer(m.mat[, 3])
> is.integer(m.df[, 3])
```

Task 3 (4 points)

This task is about creating and manipulating a data frame in R.

```
> gradebook.df # 40 students
```

```
   gindex grade         desc  fail gender proj
1       3     C Satisfactory FALSE Female   18
2       3     C Satisfactory FALSE Female   18
3       4     D         Poor FALSE Female   18
4       1     A    Excellent FALSE Female   18
5       1     A    Excellent FALSE Female   18
6       2     B         Good FALSE Female   18
7       2     B         Good FALSE Female   17
8       2     B         Good FALSE Female   17
9       2     B         Good FALSE Female   17
10      1     A    Excellent FALSE Female   17
11      2     B         Good FALSE Female   16
12      3     C Satisfactory FALSE Female   16
13      1     A    Excellent FALSE Female   16
14      2     B         Good FALSE Female   15
15      5     F   Inadequate  TRUE Female   15
16      1     A    Excellent FALSE Female   15
17      2     B         Good FALSE Female   15
18      1     A    Excellent FALSE Female   15
19      4     D         Poor FALSE Female   15
20      1     A    Excellent FALSE Female   15
21      1     A    Excellent FALSE   Male   18
22      1     A    Excellent FALSE   Male   18
23      3     C Satisfactory FALSE   Male   18
24      3     C Satisfactory FALSE   Male   18
25      1     A    Excellent FALSE   Male   18
26      3     C Satisfactory FALSE   Male   17
27      2     B         Good FALSE   Male   17
28      2     B         Good FALSE   Male   17
29      1     A    Excellent FALSE   Male   17
30      2     B         Good FALSE   Male   16
31      3     C Satisfactory FALSE   Male   16
32      2     B         Good FALSE   Male   16
33      1     A    Excellent FALSE   Male   16
34      1     A    Excellent FALSE   Male   16
35      2     B         Good FALSE   Male   15
36      3     C Satisfactory FALSE   Male   15
37      3     C Satisfactory FALSE   Male   15
38      2     B         Good FALSE   Male   15
39      3     C Satisfactory FALSE   Male   15
40      2     B         Good FALSE   Male   15
```

```
> sapply(gradebook.df, class)
```

```
   gindex      grade       desc       fail     gender       proj
"integer"   "factor"   "factor"  "logical"   "factor"  "integer"
```

(a) (1 point) Create the data frame `gradebook.df`. [Hint: It doesn't involve a lot of typing.]

(b) (1 point) Create a data frame that contains the number of students for each grade.

(c) (1 point) Create a data frame that contains the mean `proj` for each grade.

(d) (1 point) Create a random sample of size 10 as a data frame out of those 40 students.

Task 4 (3 points)

This task is about 4 functions for every statistical distribution function. For example, the normal distribution has `pnorm`, `qnorm`, `dnorm`, and `rnorm`. The first 3 are for computing cumulative probabilities, quantiles and density values, respectively, and the last one is for generating random numbers.

(a) (1 point) Use `rnorm` to generate a random sample of size 100 from $N(4, 2^2)$, and then use `hist` to plot a histogram (for frequencies).

(b) (1 point) One can also use `hist` to plot a histogram representing an density estimate. Do this with the above sample, and superimpose it with the true density curve.

(c) (1 point) Generate a sample mixed with 300 random values drawn from $N(0, 1)$ and 700 ones from $N(4, 2^2)$ by using `rnorm` *only once*.

Task 5 (3 points)

This task is about plotting using different colours.

(a) (1 point) Create a graph of the density of the chi-square distribution. Write your solution as an R function `chisqdens.plot` that depends on a parameter `nu` (degrees of freedom) so that it is easy to try different values of `nu`. The lower end of the plot should always be `x=0`, and upper value should be set to `qchisq(0.999, nu)` by default. Use your function to create a graph for the density of χ_2^2.

(b) (1 point) Modify your function `chisqdens.plot` so that it can take a vector `nu.vec` and create a single graph with multiple densities on it, one for each element of `nu.vec` using different colours and line types. Use this version of your function to create a graph that shows the chi-square densities with degrees of freedom 2, 4, 8 and 16.

(c) (1 point) Modify your function `chisqdens.plot` again so that the areas under the different density curve are filled using different colours generated with the `hsv` function using an alpha value of 0.25.

Task 6 (4 points)

This task is about investigating the unknown distribution that generated real data.

(a) (1 point) Study the following

```
> data("faithful") # load built in data set
> ?faithful
```

What information do the variables `eruptions` and `waiting` contain?

(b) (1 point) Produce a histogram and a density plot of the waiting variable. First use the default bandwidth and then try to find a better value.

(c) (1 point) Produce a normal QQ plot of the waiting variable. What does the plot show?

(d) (1 point) Produce a plot of `waiting` against `eruptions` and add a smooth curve to the plot using `lowess`. Can you interpret what the plot is saying?

Task 7 (2 points)

This task is about trellis plots. To produce these plots you will first need to run the following

```
> library(lattice)
```

The R data set `ethanol` contains data on tests of a single cylinder engine to investigate how the amount of nitrous oxides (NOx) produced by the engine depend on how the engine is tuned. Use `xyplot` to investigate the effect of `C` and `E` on `Nox` graphically.