Homework and Project Code Style Guide
of
STAT 5353 Probability & Statistics for Data Science & Bioinfo.

Good style is important because while your code only has one author, it will usually have multiple readers, such as your team members and graders. When you know you will be working with multiple people on the same code, it's a good idea to agree on a common style up-front.

To ensure all students to write code in a consistent way, a student who will not follow the guide in most of his/her homework assignment or project report will the entire attitude credit.

1 Notation and Naming

1.1 File names

File names should end in .r and be meaningful.

```
# Good
explore_diamonds.r
qiwei_li_hw_1.r

# Bad
foo.r
my_homework.R
```

1.2 Identifiers

Variable and function names should be lowercase. Use `_` to separate words within a name. Generally, variable names should be nouns and function names should be verbs. Strive for concise but meaningful names (this is not easy!)

```
# Good
day_one
day_1

# Bad
```

```
first_day_of_the_month
DayOne
dayone
djm1
```

2 Syntax

2.1 Spacing

Place spaces around all operators (`<-`, `+`, `-`, `*`, `/`, `==`, etc.). Do not place a space before a comma, but always place one after a comma.

```
# Good
average <- mean(feet / 12 + inches, na.rm = T)

# Bad
average<-mean(feet/12+inches,na.rm=T)
```

Place a space before left parentheses, except in a function call.

```
# Good
if (debug)
for (i in 1:100)
plot(x, y)

# Bad
if(debug)
for(i in 1:100)
plot (x, y)
```

Do not place spaces around code in parentheses or square brackets. (Except if there's a trailing comma: always place a space after a comma, just like in ordinary English.) Place a space before left parentheses, except in a function call.

```
# Good
if (debug)
diamonds[5, ]

# Bad
if ( debug )  # No spaces around debug
x[1,]  # Needs a space after the comma
x[1 ,]  # Space goes after, not before
```

2.2 Curly braces

An opening curly brace should never go on its own line and should always be followed by a new line; a closing curly brace should always go on its own line, unless followed by else. Always indent the code inside the curly braces.

```
# Good
if (y < 0 && debug) {
  message("Y is negative")
}

if (y == 0) {
  log(x)
} else {
  y ^ x
}

# Bad
if (y < 0 && debug)
message("Y is negative")

if (y == 0) {
log(x)
}
else {
y ^ x
}
```

2.3 Indentation

When indenting your code, use two spaces. Never use tabs or mix tabs and spaces.

2.4 Line length

Keep your lines less than 80 characters. This is the amount that will fit comfortably on a printed page at a reasonable size. If you find you are running out of room, this is probably an indication that you should encapsulate some of the work in a separate function.

2.5 Assignment

Use `<-`, not `=`, for assignment.

```
# Good
x <- 5
```

```
# Bad
x = 5
```

3 Organization

3.1 Commenting guidelines

Comment your code. Entire commented lines should begin with # and one space. Comments should explain both why and what.