Note: Please submit your R program via Blackboard by the due date. The instruction for submitting the assignment is written on the course syllabus. Please read *AssignmentFormat.pdf* file for correct assignment format.

Grading

This assignments will be graded on correctness.

Format: 2 points

Please read *AssignmentFormat.pdf* file for correct assignment format.

Problem 1: 4 points

The purpose of this problem is detecting extreme values. Write a function, `extreme`, to detect extreme values. You can consider values greater than or less than 3 times standard deviation from the mean as extreme values.

- This function will take only one argument, which is a numeric vector.
- If extreme values were found, say 5, print "There are 5 extreme values found." If extreme values were not found, print "There are no extreme values."
- Generate 1000 numbers following standard normal distribution and use this function to test the extreme function.

Problem 2: 5 points

Write a function, `calCS`, to perform the following task:

- Calculate the area of a circle ($AC = \pi r^2$), the circumference of a circle ($CC = 2\pi r$), the volume of a sphere ($VS = \frac{4}{3}\pi r^3$), or the area of a sphere ($AS = 4\pi r^2$)
- The first argument is either "AC", "CC", "VS", or "AS" to determine which calculation needs to be performed
- The value that the first argument takes can contain either lower, upper, or mixed cases of letters (Use the `toupper` function)
- If the values of the argument are not "AC", "CC", "VS", or "AS", stop the function and write `` `your method is not supported' ``
- The second argument is the radius (`r`). Make sure to use `if ... else` statements for this problem

Problem 3: 4 points

For circles with radii 5, 10, 15, 20, and 25, write a loop to calculate the the area of the circle by using the function that you just wrote. In previous function, to calculate the area of a circle with radius = 5, we can simply write `calCS(`AC', 5)`; however, when you insert the function inside the loop, you need to explicitly use the print function. For example, `print (calCS(`AC', 5))`

Problem 4: 5 points

Based on the `painters` data set from the `MASS` library:

- Create a data set which contains observations with `Colour` >=17 and `School` equals "D".
- Create a data set that contains only `Da Udine` and `Barocci`.
- Create a data set which contains observations with `Colour` >=17 and `School` equals "D", but only with the `Composition` and `Drawing` variables.
- Create a categorical variable `Comp.cat` with three approximate equal levels based on `Composition`.

Problem 5: 5 points

For this problem, we will use the following data, which is in the wide form:

Program	s1	s2	s3	s4
CONT	85	85	86	85
RI	79	79	79	80
WI	84	85	84	83

- After creating this data set, transform it into the long form.
- Then transform the long form back to the wide form.

Problem 6: 5 points

Write a function, called `stackDataInList`. This function only takes one argument, `alist`, which is a list of `data.frame`. The number of elements in the list varies. In this function, you will stack the data frames on top of each other and return one single data frame. There are many solutions to solve this problem; however, please make sure to use a loop within the function to solve this problem. Without using a loop will not receive credits for this problem.

To test this function, you will use the `datList` object. The result from the `stackDataInList` should look like the ones below:

```
> load("datList.RData")
> datList

[[1]]
  ID     value
1  a 0.9101742
2  b 0.3841854
3  c 1.6821761

[[2]]
  ID      value
1  d -0.6357365
2  e -0.4616447
```

```
[[3]]
  ID      value
1  f  1.4322822
2  g -0.6506964
3  h -0.2073807
4  i -0.3928079

[[4]]
  ID      value
1  j -0.3199929
2  k -0.2791133
3  l  0.4941883
```

```
> stackDataInList(datList[1])

  ID     value
1  a 0.9101742
2  b 0.3841854
3  c 1.6821761
```

```
> stackDataInList(datList[c(1,3,4)])

   ID      value
1   a  0.9101742
2   b  0.3841854
3   c  1.6821761
4   f  1.4322822
5   g -0.6506964
6   h -0.2073807
7   i -0.3928079
8   j -0.3199929
9   k -0.2791133
10  l  0.4941883
```

```
> stackDataInList(datList)

   ID      value
1   a  0.9101742
2   b  0.3841854
3   c  1.6821761
4   d -0.6357365
5   e -0.4616447
6   f  1.4322822
7   g -0.6506964
8   h -0.2073807
9   i -0.3928079
10  j -0.3199929
11  k -0.2791133
12  l  0.4941883
```