Part 2: Longitudinal Data Class and Methods

The purpose of this part is to create a new class for representing longitudinal data, which is data that is collected over time on a given subject/person. This data may be collected at multiple visits, in multiple locations. You will need to write a series of generics and methods for interacting with this kind of data.

The data for this part come from a small study on indoor air pollution on 10 subjects. Each subject was visited 3 times for data collection. Indoor air pollution was measured using a high-resolution monitor which records pollutant levels every 5 minutes and the monitor was placed in the home for about 1 week. In addition to measuring pollutant levels in the bedroom, a separate monitor was usually placed in another room in the house at roughly the same time.

Before doing this part you may want to review the section on object oriented programming (you can also read that section here).

The data are available as a CSV file and here:

data.zip

The variables in the dataset are

- id: the subject identification number
- visit: the visit number which can be 0, 1, or 2
- room: the room in which the monitor was placed
- value: the level of pollution in micrograms per cubic meter
- timepoint: the time point of the monitor value for a given visit/room

You will need to design a class called “LongitudinalData” that characterizes the structure of this longitudinal dataset. You will also need to design classes to represent the concept of a “subject”, a “visit”, and a “room”.

In addition you will need to implement the following functions

1. make_LD: a function that converts a data frame into a “LongitudinalData” object
2. subject: a generic function for extracting subject-specific information
3. visit: a generic function for extracting visit-specific information
4. room: a generic function for extracting room-specific information

For each generic/class combination you will need to implement a method, although not all combinations are necessary (see below). You will also need to write print and summary methods for some classes (again, see below).

To complete this Part, you can use either the S3 system, the S4 system, or the reference class system to implement the necessary functions. It is probably not wise to mix any of the systems together, but you should be able to compete the assignment using any of the three systems. The amount of work required should be the same when using any of the systems.

For this assessment, you will need to implement the necessary functions to be able to execute the code in the following script file:

oop_output.R

The output should appear similar to the output in the following file:

oop_output.txt

The output of your function does not need to match exactly, but it should convey the same information.

In order to submit this assignment, please prepare two files:

1. oop_code.R: an **R script file** that contains the code implementing your classes, methods, and generics for the longitudinal dataset.
2. oop_output.txt: a text file containing the output of running the above input code.