AAI/CPE/EE 551 WS/WS1

Course Project

Suraj Bose and Nicole Giardino

sbose5@stevens.edu, ngiardin@stevens.edu

10/16/2025

"I pledge to adhere to the Stevens Graduate Student Code of Academic Integrity"

Signed,

Suraj Bose

Nicole Giardino

Problem Statement:

A dataset (csv) is provided which consists of 7,200 records with detailed hourly (00:00 to 23:00) observations related to building energy consumption as temperature, humidity, building type, and occupancy changes. These observations are tied to 20 buildings with differing IDs, B001-B020, and the study was conducted for two full weeks. The team sets out to develop a python program capable of predicting the energy consumption of buildings under different conditions.

Part 1:

Our program will have two meaningful and well-defined classes to help achieve our goal of energy performance prediction. The first class will be a "building" class, under which we will house each individual building's data over time. This would include the full temperature, humidity, energy consumption, building type, and occupancy over the span of the 14 days that data was collected. This would be repeated over each building to help separate them into groups which we can then use to compare specific parameters. Specifically, the constructor would be making the definition of the building class. Then the attributes would be the building title, building type, and all the datasets assigned to the class. The methods would be the operations performed on the attributes, such as any calculations as required to achieve our goal. Finally, the instance object would be what we would call to get the outputs from our class.

We intend on making the relationship between our classes a composition as it would better suit our goal. Our second class would be developing the model that would rely on the building class to be produced. This class would read the building classes for each building and then determine the relationship between the data. Specifically, this class would produce the method used to predict the energy consumption based on the parameters input by the user.

The first function we would define would read through the csv file and reproduce it in the python workspace. This function would be housed within the development of the building class. Our second function would be used for the energy prediction model within the second class. These two functions would likely be reliant on a series of loops and conditional statements to iterate through the buildings and evaluate the performance and refinement of the predictive model. For example, we believe that we will have an iterative for loop to read the csv dataset while producing the building class. Similarly, there should be a while loop within the second class which will continue refinement of the predictive model until a certain accuracy metric is achieved. For the accuracy metric, the team is considering using RMSE through the numpy library. Matplotlib can be used to plot the datasets of various buildings and produce visuals for the user to better understand the results that they are seeing from the output of our program.

The team plans on handling exceptions as we work on our program and determine the most critical locations to ensure the program is robust and will not fail. An example of this would be producing error messages if the data file is not of the correct format or there are issues with the alignment of data in the csv file. This can be done with except statements. Using pytest, the team intends on creating two standalone python scripts which will check to ensure that a dataset

is properly imported, and a second one which will ensure that the dataset that is imported is going to be assigned correctly to building classes.

For ease of usage by a user who did not develop the code, my teammate and I will provide a README file which summarizes the usage of the program and a short description of how the datasets are managed and filtered. Additionally, the entire program will be well-commented and have a docstring to ensure continuity of work both between our team and to make the process clear for the user.

Part 2:

The four chosen components to use for Part Two of this project are as follows: 2. Comprehension for at least one data type, for example, list comprehension, 3. At least one built-in library/module, 4. At least two types of mutable objects, two types of immutable objects, and 8. __str__. The intended uses of these components can be seen described below.

First, comprehension, and specifically, list comprehension will be used to filter, sort, and/or process the data found within the Excel file. This feature will allow the team to organize and transform information to make the data more readable for the user, as well as more useful for analyzing trend data. For example, list comprehension can be used to filter data records where energy exceeds a specified threshold. It can also be used to convert the original temperature values from Celsius to Fahrenheit, which quickly generates a new list of values in units that are more understandable to the team.

Second, built-in libraries or modules can be used. With Pandas, the team's Excel/CSV files are put into a structured table. This allows the team to filter, sort, calculate averages, and even create new columns. Using libraries such as this will be imperative to understanding, adding to, and sorting through the data given.

Additionally, both mutable and immutable data types will be utilized in this code. Immutable objects, such as strings and tuples, will be used for fixed data from the Excel file, such as building types/IDs or timestamps, which remain constant throughout the program's execution. Mutable objects, such as lists or dictionaries, will be used to store and organize data records such as building energy usage, temperature, humidity, and occupancy, as these values can change and be updated during the analysis.

Finally, the __str__ method will be used in the classes of the program to provide the user with readable summaries of the objects. For instance, printing a Building object will be able to display all of the information for a specified building at a user-selected time and building ID number in a concise format. This will make it easier for users to read the desired information about a specified building, without being overwhelmed by the full list of records or having to manually inspect all of the data records themselves.

Building Energy Usage Dataset:
https://www.kaggle.com/datasets/programmer3/building-energy-usage-dataset?resource=download

Requirements:
Suraj
Nicole
Both

Meaningful, Well-defined class with constructors, attributes, methods, and instance objects	Built original class with __init__ to initialize attributes, an instance method to update data, and a __str__ to return data as a string	Complete
Meaningful, Well-defined class with constructors, attributes, methods, and instance objects- Relate to the class above with either inheritance or composition	Created the class - buildingCalc. Contains constructors, attributes.	Complete
Meaningful & Well-Defined Function to Solve the Problem	Function under buildingCalc to calculate estimates	Complete
Meaningful & Well-Defined Function to Solve the Problem	Matplotlib to visualize data: Created 4 plots per run that update based on users inputs	Complete
2 Advanced Libraries are used (matplotlib, PyTorch, NumPy, Pandas)	Pandas and matplotlib are used	Complete
2 Approaches to Capture Exceptions	ValueError if invalid building selected inside the while loop ValueError if no data is available after applying the filters.	Complete
2 Meaningful tests with pyTest	Created three tests: 1 checks if the correct timeframe is assigned based on timeofday data. The other two check if the data columns of temp, humidity, and energy usage are all numbers (one passes and one fails for two separate cases to show functionality)	Complete

Meaningful data I/O -Reading from a file or database	Data is loaded to df using Pandas	Complete
For Loop	For loop to group dataset by building ID	Complete
While Loop	while loops were created to let the user choose what building stats they want to see; this includes a ValueError Exception if an invalid building is input (Created the user interface and logic for the while loops)	Complete
If Statement	Nested if statements within the while loop	Complete
Docstring & meaningful comments for each class/function	Adding comments as we work	Complete
README	Wrote the README	Complete
One of the special functions: enumerate, map, zip, filter, lambda, or reduce	Updated the old filter function (boolean indexing) to use filter() and lambda instead	Complete
Comprehension for at least one data type, for example, list comprehension	Used within BuildingCalc to process temperature data and put it in Fahrenheit	Complete
one built-in library/module	Filter, lambda, len(), and print() are examples of built in modules	Complete
two types of mutable objects, two types of immutable objects	Pandas Dataframe & buildings dictionary are mutable. Strings & Tuples are immutable. Added tuple for TIMEOFDAY	Complete
one operator overloading	Added in operator overloading with len() inside the Building Class. This allows us to call self instead of self.df	Complete
One generator function or generator expression	Generator function row_check in buildingcalc class	Complete
__name__	Used for the main script code	Complete
__str__	Used for making the data printed more readable and understandable to the user	Complete