# Project Planning

1. Pick a data source
   1. This can be your own or one from the given list
   2. If you decide to create your own, make that first.
2. List out the requirements
   1. If there are any “either or” requirements pick the one that you will do
3. Underneath each requirement, describe how your application will meet those requirements
   1. Don’t be too specific here.
   2. For example: one of the requirements is “You must create a class”. Just put down what the name of that class will be.
4. Once you are sure that your project will meet all the requirements, start breaking down how the project will be implemented. This is where you will describe what your class will look like. What functions an object will have. What format the data will persist in (ie, csv file or database). What data will be displayed on the screen. Think through all the possibilities you can; you will miss some though, so don’t sweat it too much.
5. Once you have done that, break it down even further and describe each individual function.
6. Go write the program! Focus on individual tasks; don’t worry about the big picture at this point. Focus on implementing the things you defined in step 4 using the information you created in step 5.

Note: You will not be able to plan everything out perfectly. Once you start programming and you noticed you missed something, or something won’t work, that’s okay. The purpose of this is to get you going and help you break the project out into manageable chunks.

# Example

1. I will use data from some site that has average professor salaries averaged out for the past 10 years by state

* Find a dataset of your choosing that your application will read from.
* You must use Object Oriented Programming principles and data structures to model your data in the following manner:
  + You must create a Class that models all or, more likely, a subset of your data
  + You must create at least two instances of that class, each representing a separate piece of data.
  + You must populate the variables of those objects from your dataset
* Your application must persist (write) data to a file, database, or other external location.
* Your application must allow the user to view this data in some manner.
* Your code has comments
* It must include a README file located at the top level directory of your project providing the following:
  + A brief description of your project’s purpose
  + Instructions on any special requirements to run your project.
* Your project code is on your GitHub account in its own repository



* Find a dataset of your choosing that your application will read from.

Some website

* You must use Object Oriented Programming principles and data structures to model your data in the following manner:
  + You must create a Class that models all or, more likely, a subset of your data

Salary class

* + You must create at least two instances of that class, each representing a separate piece of data.

They both will be in my file reader where I map the file to the object and put a bunch into a list

* + You must populate the variables of those objects from your dataset

Same place as above

* Your application must persist (write) data to a file, database, or other external location.

I will write to a new csv file in my writer class

* Your application must allow the user to view this data in some manner.

I will display a list of the data in a table in the console window

* Your code has comments
* It must include a README file located at the top level directory of your project providing the following:
  + A brief description of your project’s purpose

{{Write your description}}

* + Instructions on any special requirements to run your project.

{{ Any extra steps you will need to make your file run. This could be where to store your file etc }}

* Your project code is on your GitHub account in its own repository

1. I will have 4 classes. A class that stores all the data called Salaries, reader class called SalaryReader, a writer class called SalaryWriter, and a Program class.
   1. Salaries
      1. This will just have a bunch of properties that will store the data from the salary csv
   2. SalaryReader
      1. This class will read from the main data source
      2. This class will read the saved data from last session
   3. SalaryWriter
      1. This will write data to a file to save the users session when they exit
   4. The Program class will have everything for running the console app.
      1. Reading user input
      2. Outputting messages to users
      3. Sending data to the writer and reader classes
2. I will have 4 classes. A class that stores all the data called Salaries, reader class called SalaryReader, a writer class called SalaryWriter, and a Program class.
   1. Salaries
      1. This will just have a bunch of properties that will store the data from the salary csv
         1. Id: int
         2. State: string
         3. AvergeSalary: decimal
   2. SalaryReader
      1. This class will read from the main data source
         1. Takes a string which will be the file path
         2. Returns a list of salary objects
         3. Flow
            1. This will find the file from the path using a StreamReader
            2. I will read a line from the stream and map it to an object
            3. I will add that object to a list
            4. I will do this until the whole file is read
      2. This class will read the saved data from last session
         1. This will take nothing and return a list of Salary objects
         2. There is a save file that will be checked upon app start
         3. Find the file and open it using a streamReader.
         4. Map line to the salary object
         5. Put it in the list
         6. Do this until the whole file is read
   3. SalaryWriter
      1. This will write data to a file to save the users session when they exit
         1. This will take a list of salary objects and return nothing
         2. This will create a save file and open it using a StreamWriter
         3. This will loop through the list and write the a line per object
   4. The Program class will have everything for running the console app.
      1. Reading user input
         1. Ask the user for input. I will have three legit options
            1. Exit
            2. Average salaries for the country Salaries
            3. Average salaries by state
      2. Outputting messages to users
      3. Sending data to the writer and reader classes
         1. At the start, I will read data from the save file. If no file exists, I will get a clean one from the main source.
         2. At the end I will write all my data in memory to a save file