ASSIGNMENT 2 REPORT

Ideas:

- I tried to write the most efficient code and tried to not repeat myself. For that, I find most efficient gathering needed variables in base class (Personnel) and modifying these in subclass's constructor as needed.

- As a result of reducing code length, I was able to use only one function to calculate the salaries of jobs except part-time employee and security. (These classes override getSalary() method of Personnel class)

- I used a "Parser" class to read/parse input files, I choose to separate my code to classes.

Algorithm:

- getPersonnels() function of Parser reads the personnel.txt input file, looping it line by line, gets required information. Also, this function is responsible for creation of objects (Security, Chief, Worker…). This function benefits from polymorphism as it treats every sub class of Personnel as Personnel, by assigning them to Personnel object and adding them together to Personnel ArrayList.

- parseWorkingHours() is the other function of Parser class, which reads second input file monitoring.txt. It compares registration numbers of people, and if a match is found it creates an ArrayList which holds working hours information and stores it in Personnel object.

- For salary calculation, a lot of jobs used same function getSalary(). This was possible because the calculation of salaries could be mostly reduced to a function with several parameters.

- For example, if we think a chief, he gets 125TL per day. If we calculate the money gained per month (125 * 5 * 4), we get 2500, so we can say he has a baseSalary like an academician or an officer. Also they have the same default working hours, 40. So calculation is really same.

- Although similar, it is not the same for security personnel and part-time employee's because these jobs have a minimum hour of working, so they need a different algorithm to calculate the salary as they don't get money for a week if they don't exceed the minimum hours. To achieve this kind of behavior, these two classes override the default getSalary() method of Personnel class.

- Once calculation is done, output is returned by getInfo() method of Personnel class, which is directly written to the output file. I do this in a loop to write all necessary outputs.

- So, to summarize, after getting information from input files, I believe it is easy to calculate the results because no complex algorithm is needed. There are some OOP concept to use/ not use, and there are a lot of ways to implement this. I find that my implementation is both easy to read and understand.

Class Structure and UML Diagram: