Psuedocode for Person.java

* Global Variables:
  + name to hold the name
  + address to hold the address
  + phoneNumber to hold the phone number
  + emailAddress to hold the email address
* Person (String, String, String, String) Constructor:
  + Set the name to what was passed
  + Set the address to what was passed
  + Set the phoneNumber to what was passed
  + Set the email address to what was passed
* getName() method:
  + return the name
* setName(String) method:
  + Set name to what was passed
* getAddress() method
  + Return the address
* setAddress(String) method
  + Set address to what was passed
* getPhoneNumber() method
  + Return phoneNumber
* setPhoneNumber(String) method
  + Set phoneNumber to what was passed
* getEmailAddress() method
  + Return emailAddress
* setEmailAddress(String) method
  + Set emailAddress to what was passed

Pseudocode for Student.java (Subclass of Person)

* Global variables:
  + STATUS to hold the class status
* Student(String, String, String, String, String) Constructor:
  + Call superclass Person, and pass the name, address, phoneNumber, and emailAddress to it
  + Set the status to what was passed
* getStatus() method
  + Return STATUS
* toString() method
  + Return all attributes, and super attributes about the Student.

Pseudocode for Employee.java (Subclass of Person)

* Global Variables:
  + officeNumber to hold the office number
  + salary to hold the salary
  + dateHired to get the date hired of the Employee
* Employee(String, String, String, String, int, int, MyDate) Constructor
  + Call superclass Person and pass the name, address, phone number, and email address passed
  + Set officeNumber to what was passed
  + Set the salary to what was passed
  + Set the dateHired to what was passed
* getOfficeNumber() method
  + Return officeNumber
* setOfficeNumber(int) method
  + Set the officeNumber to what was passed
* getSalary() method
  + Return the salary
* setSalary(int) method
  + Set the salary to what was passed
* getDateHired() method
  + Return a reference to the MyDate object
* setDateHired(MyDate) Method
  + Set dateHired to the reference passed.

Pseudocode for Faculty (Subclass of Employee)

* Global variables:
  + officeHours to hold the officeHours
  + rank to hold the rank of the employee
* Faculty(String, String, String, String, int, int, MyDate, String, String) Constructor
  + Call super class and pass the name, address, phone number, email address, office number, salary, and date hired that was passed
  + Set the officeHours to what was passed
  + Set the rank to what was passed
* getOfficeHours() method
  + Return officeHours
* setOfficeHours(String) method
  + Set officeHours to what was passed
* getRank() method
  + Return the rank
* setRank(String) method
  + Set the rank to what was passed
* toString() method
  + Return attributes from the super class and Faculty class

Pseudocode for Staff.java

* Global variables
  + title to hold the Staff title
* Staff(String, String, String, String, int, int, MyDate, String) Constructor
  + Call the Superclass and pass the name, address, phone number, email address, office number, salary, and date hired that was passed
  + Set the title to what was passed
* getTitle() method
  + Return the title
* setTitle(String) method
  + Set the title that was passed
* toString() method
  + Return the attributed of the Superclass and Staff

Pseudocode for MyDate.java

* Global variables
  + date to hold the date
* MyDate(String) Constructor
  + Set the date to what was passed
* getDate() method
  + Return the date

Pseudocode for Driver.java

* Variables
  + choice to hold the choices from the end user
  + name to hold the name
  + address to hold the address
  + emailAddress to hold the email address
  + phoneNumber to hold the phone number
  + officeNumber to hold the office number
  + salary to hold the salary
  + dateHired to hold the date hired
  + officeHours to hold the office hours
  + rank to hold the class rank
  + title to hold the title of the Staff
  + status to hold the class status
* Ask the user to select 1 to create a Student or 2 to create an Employee
* Record the choice in choice
* If the end user selects 1:
  + Ask for the student’s name and record
  + Ask for the address and record
  + Ask for the phone number and record
  + Ask for the email Address and record
  + Ask for the class status and record
  + Create a new Student object with the information you got
  + Print the Student object using the toString method
* If the end user selects 2:
  + Ask for the Employee’s name and record
  + Ask for the address and record
  + Ask for the phone number and record
  + Ask for the email address and record
  + Ask the user to select 1 to create a Faculty member or 2 for a Staff member
  + Get the choice
  + If the user selects 1:
    - Ask for the office Number and record
    - Ask for the salary and record
    - Ask for the date hired and record
    - Ask for the office hours and record
    - Ask for the rank and record
    - Create a new Faculty object with all the information recorded
    - Print the faculty object calling the toString method
  + If the user selects 2:
    - Ask for the office Number and record
    - Ask for the salary and record
    - Ask for the date hired and record
    - Ask for the title and rcord
    - Create a Staff object with all the information recorded
    - Print the Staff object calling the toString method.

Test Plan

Create test cases to make sure everything is outputting correctly and not having problems with messing information around

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Cases** | **Input** | **Expected Result** | **Actual Result** | **Did it pass?** |
| Case 1 | Student or Employee: Student  Name: Michael Amaya  Address: 123 Main St  Phone Number: (301) 555-5555  Email Address: [mike@someemail.com](mailto:mike@someemail.com)  Class Status: Sophomore | Student: Michael Amaya  Status: Sophomore  Address: 123 Main St  Phone Number: (301) 555-5555  Email Address: mike@someemail.com | Student: Michael Amaya  Status: Sophomore  Address: 123 Main St  Phone Number: (301) 555-5555  Email Address: mike@someemail.com | Y |
| Case 2 | Student or Employee: Employee  Name: Sierra Thomas  Address: 321 Main St  Phone Number: (240) 777-7777  Email Address: [sierrat@newemail.edu](mailto:sierrat@newemail.edu)  Faculty or Staff: Faculty  Office Number: 400  Salary: 54000  Hire Date: 01/20/2019  Office Hours: Mon-Fri 6:00am-10:00am  Rank: Art Professor | Faculty: Sierra Thomas  Rank: Art Professor  Salary: 54000  Hire Date: 01/20/2019  Office Hours: Mon-Fri 6:00am-10:00am  Office Number: 400  Address: 321 Main St  Phone Number: (240) 777-7777  Email Address: sierrat@newemail.edu | Faculty: Sierra Thomas  Rank: Art Professor  Salary: 54000  Hire Date: 01/20/2019  Office Hours: Mon-Fri 6:00am-10:00am  Office Number: 400  Address: 321 Main St  Phone Number: (240) 777-7777  Email Address: sierrat@newemail.edu | Y |
| Case 3 | Student or Employee: Employee  Name: Nathan Huddlestone  Address: 35 manor road, Griston, Thetford, IP25 6RW  Phone Number: 07521911992  Email Address: [n.huddlestone@hotmail.co.uk](mailto:n.huddlestone@hotmail.co.uk)  Faculty or Staff: Staff  Office Number: 666  Salary: 420069  Hire Date: 04/20/2014  Title: Chocolate Tester | Staff: Nathan Huddlestone  Title: Chocolate Tester  Salary: 420069  Hire Date: 04/20/2014  Office Number: 666  Address: 35 manor road, Griston, Thetford, IP25 6RW  Phone Number: 07521911992  Email Address: n.huddlestone@hotmail.co.uk | Staff: Nathan Huddlestone  Title: Chocolate Tester  Salary: 420069  Hire Date: 04/20/2014  Office Number: 666  Address: 35 manor road, Griston, Thetford, IP25 6RW  Phone Number: 07521911992  Email Address: n.huddlestone@hotmail.co.uk | Y |

Screenshots:

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

UML Diagrams:

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Lessons Learned

I learned how to work with superclasses and subclasses. I also learned that working with UML is very important in order to make It easier to make a program. UML helps organize everything when creating subclasses and working attributes with every class.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** |  | **Y/N** | **Comments** |
|  | **Source java files** | **Y** |  |
|  | **Compressed files:** | **Y** |  |
|  | FirstInitialLastName\_Project8\_Moss.zip | **Y** |  |
|  |  |  |  |
|  | FirstInitialLastName\_Project8\_doc.zip | **Y** |  |
|  | **Program compiles** | **Y** |  |
|  | **Program runs** | **Y** |  |
|  | **Checklist is completed and included in the Documentation** | **Y** |  |
|  | **Documentation file:** | **Y** |  |
|  | **Comprehensive Test Plan** | **Y** |  |
|  | **Screenshots based on Test Plan** | **Y** |  |
|  | **UML Diagram** | **Y** |  |
|  | **Algorithms/Pseudocode** | **Y** |  |
|  | **Flowchart** | **Y** |  |
|  | **Lessons Learned** | **Y** |  |