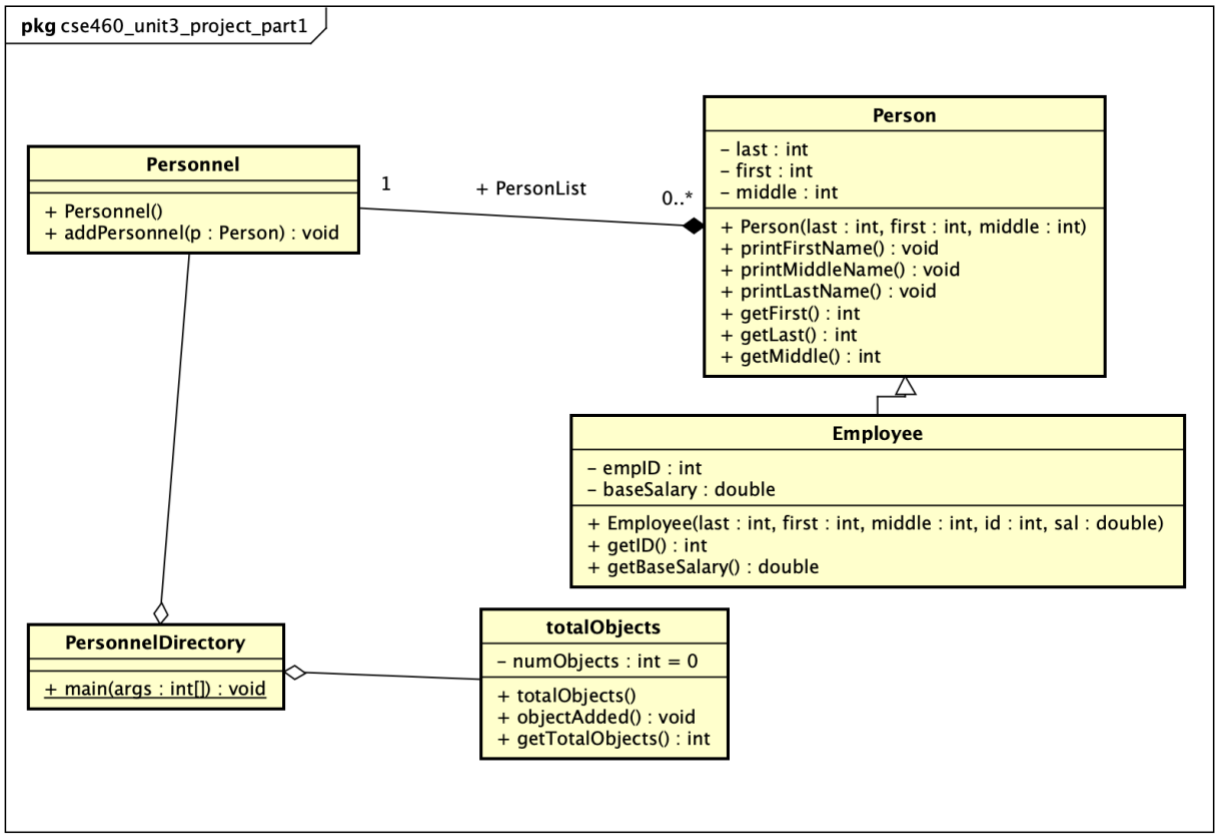
Phase I, Part 1

Use the Astah tool to draw the class diagram for the current implementation of the university system. Use correct UML notations. When you have completed the diagram, take a clear screenshot and paste it in the space provided.



Phase II, Part 1

In the code, identify object-oriented concept violations, content coupling, common coupling, control coupling and stamp coupling situations. Copy and paste the code segments the shows each coupling situation in the space provided. You may use additional space as necessary.

(At this part, I copied original part code at Object Oriented Concept Violations, Content Coupling, Common Coupling, Control Coupling and Stamp Coupling parts, I also copied changed code after "How would you fix" with red color at every part among five parts. Every explanation has green color.)

1.Object Oriented Concept Violations

First problem:

Case 1:

```
public class Person {
	public String last;
	public String first;
	public String middle;


	public Person(String last, String first, String middle) {
		this.last = last;
		this.first = first;
		this.middle = middle;

	}


	public void printName(int order)
	{

	  if(order == 0)
	  {
	    System.out.println(first + "  " + middle + "  " + last);

	  }
	  else if(order == 1)
	  {

	   System.out.println(last + " ," + middle + " " + first);

	  }
	  else if(order == 2)
		{

		  System.out.println(last + " ," + first + " " + middle);

		 }
	}


}
```

Second problem:

```
  private static int numObjects = 0;

  public totalObjects()
  {
    numObjects=0;
  }
```

How would you fix these violations:

First change:

Explanation: every last and first and middle name should have getting function, and using private to hiding variable in order to prevent another function invoke these variables, we should use public methods to return every value of variables

Case 1.

```
public class Person {
        private String last;
        private String first;
        private String middle;


        public Person(String last, String first, String middle) {
                this.last = last;
                this.first = first;
                this.middle = middle;

        }


        public void printName(int order)
        {

          if(order == 0)
          {
            System.out.println(first + "  " + middle + "  " + last);

          }
          else if(order == 1)
          {

           System.out.println(last + " ," + middle + " " + first);

          }
```

```
    else if(order == 2)
            {

              System.out.println(last + " ," + first + " " + middle);

     }
   }

   public String getFirst(){
            return first;
   }

   public String getLast(){
            return last;
   }

   public String getMiddle(){
            return last;
   }

}
```

Second change:

Explanation: the static can be invoked by variable, it is Object Oriented Concept Violations. The static should be removed.

```
  private int numObjects = 0;

  public totalObjects()
  {
    numObjects=0;
  }
```

2.Content Coupling

First problem:

case 1:

```
System.out.println("Enter firts name : ");
firstN = scan.nextLine();

System.out.println("Enter last name : ");
lastN = scan.nextLine();

boolean found = false;
int loc =-1;

for(int i =0; i <per.personList.size(); i++){

        if( per.personList.get(i).first.equals(firstN) && per.personList.get(i).last.equals(lastN){
                found = true;
                loc = i;
        }
}

if(found){
        System.out.println("Found");
        per.personList.get(loc).printName(0);
        }else{
                System.out.println("not found");
                Person p1  = new Person(lastN, firstN, " ");
                per.personList.add(p1);
                total.objectAdded();
        }
break;
```

Second problem:

```
        boolean found = false;
                int loc =-1;
                for(int i =0; i <per.personList.size(); i++)
                {
                        if( per.personList.get(i).first.equals(firstN) &&
per.personList.get(i).last.equals(lastN))
                        {
                        found = true;
                  loc = i;
                  }
                }
```

```
                    if(found)
                    {
                            System.out.println("Found");
                            per.personList.get(loc).printName(0);

                    }else
                    {
                            System.out.println("not found");
                            Person p1  = new Person(lastN, firstN, " ");
                            per.personList.add(p1);
                            total.objectAdded();
        }
```

How would you fix this:

First change:

Explanation: in order to change personnel and person class data, main function should use "addPersonnel(Person p1)" method replace personList.add function.

case 1:

```
System.out.println("Enter firts name : ");
firstN = scan.nextLine();

System.out.println("Enter last name : ");
lastN = scan.nextLine();

boolean found = false;
int loc =-1;

for(int i =0; i <per.personList.size(); i++){

        if( per.personList.get(i).first.equals(firstN) && per.personList.get(i).last.equals(lastN){
                found = true;
                loc = i;
        }
```

```
}

if(found){
        System.out.println("Found");
        per.personList.get(loc).printName(0);
        }else{
                System.out.println("not found");
                Person p1  = new Person(lastN, firstN, " ");
                per.addPersonnel(Person p1);
                total.objectAdded();

        }
break;
```

Second change:

Explanation: put the code into for loop, module Personnel can get correct loc variable from module PersonnelDirectory

```
        boolean found = false;
                int loc =-1;
                for(int i =0; i <per.personList.size(); i++)
                {
                        if( per.personList.get(i).first.equals(firstN) &&
per.personList.get(i).last.equals(lastN))
                        {
                        found = true;
                        loc = i;
                        if(found)
                {
                          System.out.println("Found");
                          per.personList.get(loc).printName(0);

                }else
                {
                          System.out.println("not found");
                          Person p1  = new Person(lastN, firstN, " ");
                          per.personList.add(p1);
                          total.objectAdded();
```

```
                    }
                }
        }
```

3.Common Coupling

```
boolean found = false;
int loc =-1;
for(int i =0; i <per.personList.size(); i++)
{
        if( per.personList.get(i).first.equals(firstN) && per.personList.get(i).last.equals(lastN))
        {
        found = true;
        loc = i;
        }
}
```

How would you fix this:

Explanation: At this part, the first and last variable become to private, so every function should call getting function in order to get value of variables.

```
boolean found = false;
int loc =-1;
for(int i =0; i <per.personList.size(); i++)
{
        if( per.personList.get(i).getFirst().equals(firstN) && per.personList.get(i).getLast().equals(lastN))
        {
        found = true;
        loc = i;
        }
}
```

4.Control Coupling

```
public void printName(int order)
{

  if(order == 0)
  {
    System.out.println(first + "  ," + middle + "  " + last);

  }else if(order == 1)
     {

     System.out.println(last + " ," + middle + " " + first);

     }
     else if(order == 2)
                        {

                        System.out.println(last + " ," + first + " " + middle);

     }
}
```

How would you fix this:

Explanation: At this part, the one print name function should be changed to three print name functions, each of them print first, middle and last name. Another modules just use switch case function invoke print first, middle and last name functions to print different of orders of them.

```
public void printFirstName()
{
    System.out.println(first + " ");
}
public void printMiddleName()
{
    System.out.println(middle + " ");
}
public void printLastName()
{
    System.out.println(last+ " ");
}
```


5.Stamp Coupling

```
public class Employee extends Person{
	private int empID;
	private double baseSalary;
	public Employee(String last, String first, String middle, int id, double sal) {
		super(last, first, middle);
		empID = id;
		baseSalary = sal;
	}
	public int getID()
	{
		return empID;
	}
}
```

How would you fix this:

Explanation: At this part, baseSalary is a private variable, no other functions can get the value of this variable. So, Employee class should create a getBaseSalary function to return value of baseSalary variable.

```
public class Employee extends Person{
	private int empID;
	private double baseSalary;
	public Employee(String last, String first, String middle, int id, double sal) {
		super(last, first, middle);
		empID = id;
		baseSalary = sal;
	}
	public int getID()
	{
		return empID;
	}
	public double getBaseSalary(){
		return baseSalary;
	}
}
```
