Brandon London Project 4.
2261-001

```
<terminated> Exercise_11_02 [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Apr 14, 2019, 3:54:48 PM)

Name: Steve
Address: 123 ABC Street
Phone number: 123458978541
Email address: stevenhawking@aol.com

Name: Sally
Address: 13423 nowhere st.
Phone number: 7894651616
Email address: hit420@aol.com
Status: freshman

Name: IamTheNight
Address: 2834 weast st
Phone number: 66642498952
Email address: Batman@Gmail.com
Office: 910
Salary: $60000.00
Date hired: 04/14/2019

Name: SueMe
Address: 28 yeet st
Phone number: 66666666666
Email address: NOextraCredit@aol.com
Office: 101
Salary: $50000.00
Date hired: 04/14/2019
Office hours: 4pm to 6pm
Rank: Professor

Name: Tom
Address: 90 home ave
Phone number: 23432340000
Email address: tommymickil@aol.com
Office: 12
Salary: $65000.00
Date hired: 04/14/2019
Title: Executive Assistant
```

Brandon London Project 4.
2261-001

```
/*********************************************************
*                      Faculty                           *
*--------------------------------------------------------*
* -officeHours: int                                      *
* -rank: String                                          *
* +Faculty(name: String, address: String, phone: String, *
*  email: String, office: int, salary: double,           *
*  officeHours: int, rank: String)     *
* +getOfficeHours(): int                                 *
* +setOfficeHours(officeHours: int): void                *
* +getRank(): String                                     *
* +setRank(rank: String): void                           *
* +toString(): String                                    *
*********************************************************/

// Implement Faculty class
public class Faculty
        extends Employee {
    // Data fields
    private String officeHours;
    private String rank;

    // Constructors
    /** Construct a Faculty object with specified name, address, phone number,
      * email address, office, salary, office hours and rank */
    public Faculty(String name, String address, String phone, String email,
        int office, double salary, String officeHours, String rank) {
        super(name, address, phone, email, office, salary);
        this.officeHours = officeHours;
        this.rank = rank;
    }

    /** Return officeHours */
    public String getOfficeHours() {
        return officeHours;
    }
```

Brandon London Project 4.
2261-001

```
37
38      /** Set new officeHours */
39      public void setOfficeHours(String officeHours) {
40          this.officeHours = officeHours;
41      }
42
43      /** Return rank */
44      public String getRank() {
45          return rank;
46      }
47
48      /** Set new rank */
49      public void setRank(String rank) {
50          this.rank = rank;
51      }
52
53      /** Return a string discription of the class */
54      public String toString() {
55          return super.toString() + "\nOffice hours: " + officeHours +
56          "\nRank: " + rank;
57      }
58  }
```

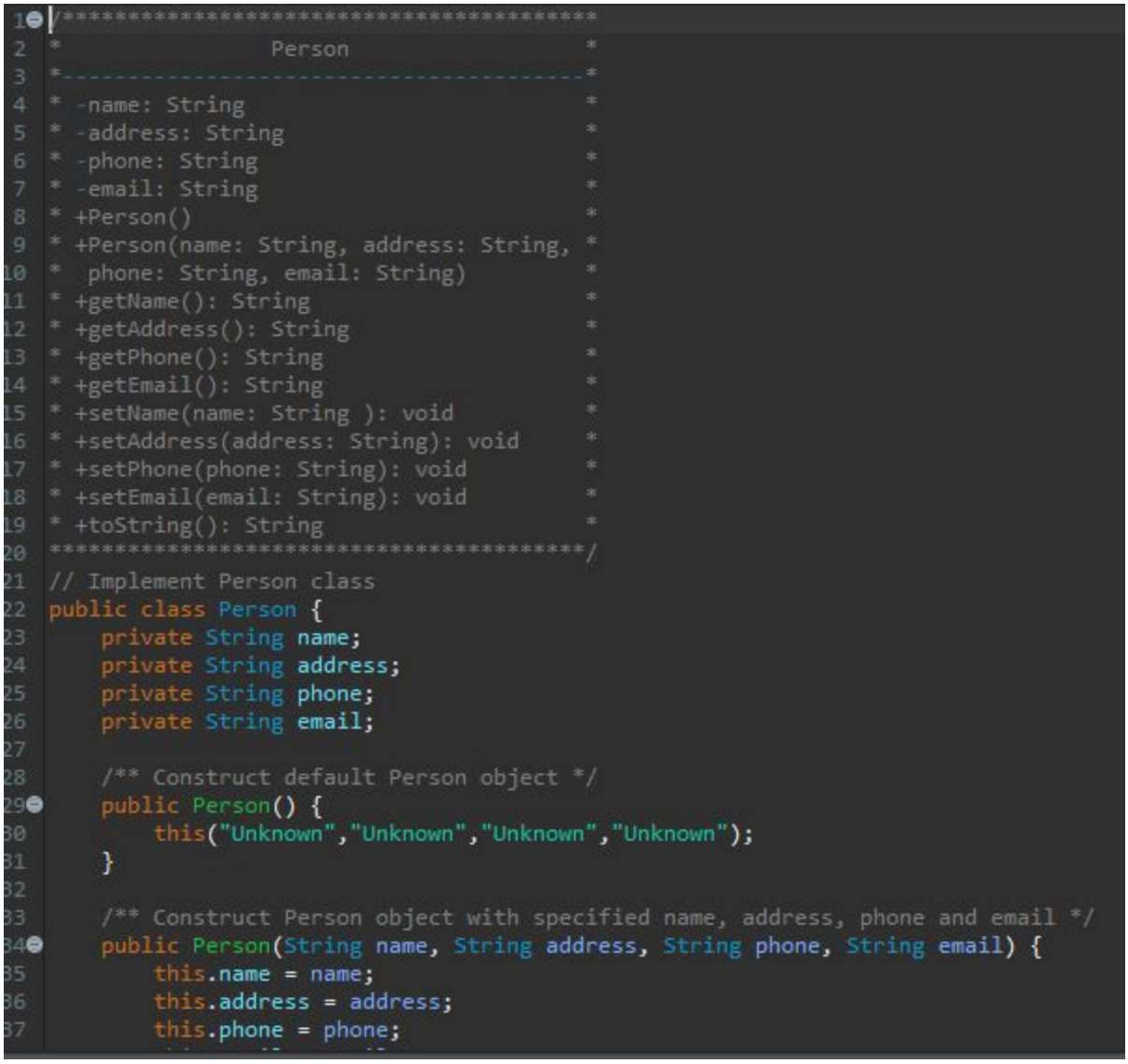
Brandon London Project 4.
2261-001

```
/*****************************************
*                MyDate                 *
*----------------------------------------*
* -year: int                            *
* -month: int                           *
* -day: int                             *
* +MyDate()                             *
* +MyDate(elapsedTime: long)            *
* +MyDate(year: int, month: int, day: int) *
* +getYear(): int                       *
* +getMonth(): int                      *
* +getDay(): int                        *
* +setDate(elapsedTime: long)           *
*****************************************/

import java.util.GregorianCalendar;

// Implement MyDate class
 public class MyDate {
    // Data Fields
    private int year;
    private int month;
    private int day;

    /** Creates a MyDate object for the current date */
    MyDate() {
        GregorianCalendar calander = new GregorianCalendar();
        year = calander.get(GregorianCalendar.YEAR);
        month = calander.get(GregorianCalendar.MONTH);
        day = calander.get(GregorianCalendar.DAY_OF_MONTH);
    }

    /** Creates a MyDate object with a specified elapsed time
    *    since midnight, January 1, 1970, in milliseconds */
    MyDate(long elapsedTime) {
        setDate(elapsedTime);
    }
```

Brandon London Project 4.
2261-001

```
    }

    /** Creates a MyDate object with the
    *   specified year, month, and day */
    MyDate(int year, int month, int day) {
        this.year = year;
        this.month = month;
        this.day = day;
    }

    /** Return year */
    public int getYear() {
        return year;
    }

    /** Return month */
    public String getMonth() {
        String m = String.valueOf(month + 1);
        return (month < 10 ? "0" + m : m);
    }

    /** Return day */
    public String getDay() {
        String d = String.valueOf(day);
        return (day < 10 ? "0" + d : d);
    }

    /** Sets a new date for the object using the elapsed time */
    public void setDate(long elapsedTime) {
        GregorianCalendar calander = new GregorianCalendar();
        calander.setTimeInMillis(elapsedTime);
        year = calander.get(GregorianCalendar.YEAR);
        month = calander.get(GregorianCalendar.MONTH);
        day = calander.get(GregorianCalendar.DAY_OF_MONTH);
    }
}
```

Brandon London Project 4.
2261-001



```
/*****************************************
*                Person                 *
*---------------------------------------*
* -name: String                         *
* -address: String                      *
* -phone: String                        *
* -email: String                        *
* +Person()                             *
* +Person(name: String, address: String, *
*  phone: String, email: String)        *
* +getName(): String                    *
* +getAddress(): String                 *
* +getPhone(): String                   *
* +getEmail(): String                   *
* +setName(name: String ): void         *
* +setAddress(address: String): void    *
* +setPhone(phone: String): void        *
* +setEmail(email: String): void        *
* +toString(): String                   *
*****************************************/
// Implement Person class
public class Person {
    private String name;
    private String address;
    private String phone;
    private String email;

    /** Construct default Person object */
    public Person() {
        this("Unknown","Unknown","Unknown","Unknown");
    }

    /** Construct Person object with specified name, address, phone and email */
    public Person(String name, String address, String phone, String email) {
        this.name = name;
        this.address = address;
        this.phone = phone;
```

Brandon London Project 4.
2261-001

```
50
51      /** Return phone */
52      public String getPhone() {
53          return phone;
54      }
55
56      /** Return email */
57      public String getEmail() {
58          return email;
59      }
60
61      /** Set new name */
62      public void setName(String name) {
63          this.name = name;
64      }
65
66      /** Set new address */
67      public void setAddress(String address) {
68          this.address = address;
69      }
70
71      /** Set new phone number */
72      public void setPhone(String phone) {
73          this.phone = phone;
74      }
75
76      /** Set new email */
77      public void setEmail(String email) {
78          this.email = email;
79      }
80
81      /** Return a string discription of the class */
82      public String toString() {
83          return "\nName: " + name + "\nAddress: " + address +
84                  "\nPhone number: " + phone + "\nEmail address: " + email;
85      }
86  }
```

Brandon London Project 4.
2261-001

```java
/**********************************************************************
*                                Staff
*---------------------------------------------------------------------
* -title: String
* +Staff(name: String, address: String, phone: String, email: String,
*  office: int, salary: double, dateHired: MyDate, title: String)
* +getTitle(): String
* +setTitle(title: String): void
* +toString(): String
**********************************************************************/
// Implement Staff class
public class Staff
        extends Employee {
    // Data Fields
    private String title;

    // Constructors
    /** Construct a Staff object */
    public Staff(String name, String address, String phone,
        String email, int office, double salary, String title) {
        super(name, address, phone, email, office, salary);
        this.title = title;
    }

    /** Return title */
    public String getTitle() {
        return title;
    }

    /** Set new title */
    public void setTitle(String title) {
        this.title = title;
    }

    /** Return a string discciption of the class */
    public String toString() {
        return super.toString() + "\nTitle: " + title;
```

Brandon London Project 4.
2261-001

```
/**************************************************
*                    Student                      *
*-------------------------------------------------*
* -status: String                                 *
* ---------------                                 *
* +Student(name: String, address: String,         *
*  phone: String, email: String, status: String)  *
* +getStatus(): String                            *
* +setStatus(status: String): void                *
* +toString(): String                             *
**************************************************/
// Implement Student class
public class Student
        extends Person {
    private int status;
    public final static int FRESHMAN = 1;
    public final static int SOPHOMORE = 3;
    public final static int JUNIOR = 2;
    public final static int SENIOR = 4;

    public Student(String name, String address,
        String phone, String email, int status) {
        super(name, address, phone, email);
        this.status = status;
    }

    /** Set new status */
    public void setStatus(int status) {
        this.status = status;
    }

    /** Return status */
    public String getStatus() {
        switch (status) {
            case 1 : return "freshman";
            case 2 : return "sophomore";
            case 3 : return "junior";
```

```
            case 3 : return "junior";
            case 4 : return "senior";
            default : return "Unknown";
        }
    }

    /** Return a string discription of the class */
    public String toString() {
        return super.toString() + "\nStatus: " + getStatus();
    }
}
```

Brandon London Project 4.
2261-001

```
            case 3 : return "junior";
            case 4 : return "senior";
            default : return "Unknown";
        }
    }

    /** Return a string discription of the class */
    public String toString() {
        return super.toString() + "\nStatus: " + getStatus();
    }
}
```

```
ComparableCircle1:
created on Sun Apr 14 15:55:37 CDT 2019
color: while and filled: false
Date created: Sun Apr 14 15:55:37 CDT 2019
Radius: 12.5
Area: 490.8738521234052

ComparableCircle2:
created on Sun Apr 14 15:55:37 CDT 2019
color: while and filled: false
Date created: Sun Apr 14 15:55:37 CDT 2019
Radius: 18.3
Area: 1052.0879637606859

ComparableCircle2 is the larger of the two Circles
```

Brandon London Project 4.
2261-001

Circle.java | GeometricObj... | Faculty.java | Person.java | *MyDate.java | Staff.java | Student.java

```java
public class Circle
        extends GeometricObject {
    private double radius;

    public Circle() {
    }

    public Circle(double radius) {
        this.radius = radius;
    }

    public Circle(double radius,
        String color, boolean filled) {
        this.radius = radius;
        setColor(color);
        setFilled(filled);
    }

    /** Return radius */
    public double getRadius() {
        return radius;
    }

    /** Set a new radius */
    public void setRadius(double radius) {
        this.radius = radius;
    }

    @Override /** Return area */
    public double getArea() {
        return radius * radius * Math.PI;
    }

    /** Return diameter */
    public double getDiameter() {
        return 2 * radius;
    }
    }

    @Override /** Return perimeter */
    public double getPerimeter() {
        return 2 * radius * Math.PI;
    }

    @Override /** Implement the toString method in GeometricObject */
    public String toString() {
        return super.toString() + "\nDate created: " + getDateCreated() +
            "\nRadius: " + radius;
    }
}
```

Brandon London Project 4.
2261-001

```java
/**********************************
*         ComparableCircle         *
*----------------------------------*
* +ComparableCircle()              *
* +ComparableCircle(radius: double)*
* +ComparableCircle(radius: double,*
*       color: String, filled: boolean) *
**********************************/

public class ComparableCircle extends Circle
        implements Comparable<ComparableCircle> {

    public ComparableCircle() {
    }

    /** Construct a CoparableCircle with specified radius */
    public ComparableCircle(double radius) {
        super(radius);
    }

    /** Construct a CoparableCircle with specified properties */
    public ComparableCircle(double radius, String color, boolean filled) {
        super(radius, color, filled);
    }

    @Override // Implement the compareTo method defined in Comparable
    public int compareTo(ComparableCircle o) {
        if (getArea() > o.getArea())
            return 1;
        else if (getArea() < o.getArea())
            return -1;
        else
            return 0;
    }

    @Override // Implement the toString method defined in Circle
    public String toString() {
```

```java
    @Override // Implement the compareTo method defined in Comparable
    public int compareTo(ComparableCircle o) {
        if (getArea() > o.getArea())
            return 1;
        else if (getArea() < o.getArea())
            return -1;
        else
            return 0;
    }

    @Override // Implement the toString method defined in Circle
    public String toString() {
        return super.toString() + "\nArea: " + getArea();
    }
}
```

Brandon London Project 4.
2261-001

```java
/**************************************************************************
* (The ComparableCircle class) Define a class named ComparableCircle that *
* extends Circle and implements Comparable. Draw the UML diagram and implement *
* the compareTo method to compare the circles on the basis of area. Write a test *
* class to find the larger of two instances of ComparableCircle objects. *
**************************************************************************/
public class Exercise_13_06 {
    /** Main method */
    public static void main(String[] args) {
        // Create two instances of ComparableCircle objects
        ComparableCircle comparableCircle1 = new ComparableCircle(12.5);
        ComparableCircle comparableCircle2 = new ComparableCircle(18.3);

        // Display comparableCircles
        System.out.println("\nComparableCircle1:");
        System.out.println(comparableCircle1);
        System.out.println("\nComparableCircle2:");
        System.out.println(comparableCircle2);

        // Find and display the larger of the two ComparableCircle objects
        System.out.println((comparableCircle1.compareTo(comparableCircle2) == 1
            ? "\nComparableCircle1 " : "\nComparableCircle2 ") +
            "is the larger of the two Circles");
    }
}
```

Brandon London Project 4.
2261-001

```java
public abstract class GeometricObject {
    private String color = "while";
    private boolean filled;
    private java.util.Date dateCreated;

    /** Construct a default geometric object */
    protected GeometricObject() {
        dateCreated = new java.util.Date();
    }

    /** Construct a geometric object with color and filled value */
    protected GeometricObject(String color, boolean filled) {
        dateCreated = new java.util.Date();
        this.color = color;
        this.filled = filled;
    }

    /** Return color */
    public String getColor() {
        return color;
    }

    /** Set a new color */
    public void setColor(String color) {
        this.color = color;
    }

    /** Return filled. Since filled is boolean,
     *  the get method is named isFilled */
    public boolean isFilled() {
        return filled;
    }

    /** Set a new filled */
    public void setFilled(boolean filled) {
        this.filled = filled;
    }
```

Brandon London Project 4.
2261-001

```
26      }
27
28      /** Return filled. Since filled is boolean,
29       *  the get method is named isFilled */
30      public boolean isFilled() {
31          return filled;
32      }
33
34      /** Set a new filled */
35      public void setFilled(boolean filled) {
36          this.filled = filled;
37      }
38
39      /** Get dateCreated */
40      public java.util.Date getDateCreated() {
41          return dateCreated;
42      }
43
44      @Override
45      public String toString() {
46          return "created on " + dateCreated + "\ncolor: " + color +
47              " and filled: " + filled;
48      }
49
50      /** Abstract method getArea */
51      public abstract double getArea();
52
53      /** Abstract method getPerimeter */
54      public abstract double getPerimeter();
55  }
```

Brandon London Project 4.
2261-001

```
/*********************************************************************
*                          Employee                                  *
*--------------------------------------------------------------------*
* -office: int                                                       *
* -salary: double                                                    *
* -dateHired: MyDate                                                 *
* +Employee(name: String, address: String, phone: String,            *
*  email: String, office: int, salary: double, dateHired: MyDate);   *
* +getOffice(): int                                                  *
* +getSalary(): double                                               *
* +getDateHired(): MyDate                                            *
* +setOffice(office: int): void                                      *
* +setSalary(salary: double): void                                   *
* +setDateHired(dateHired: MyDate): void                             *
* +toString():String                                                 *
*********************************************************************/
// Implement Employee class
public class Employee
        extends Person {
    private int office;
    private double salary;
    private MyDate dateHired;

    /** Construct Employee object */
    public Employee(String name, String address, String phone,
        String email, int office, double salary) {
        super(name, address, phone, email);
        this.office = office;
        this.salary = salary;
        this.dateHired = new MyDate();
    }

    /** Return office */
    public int getOffice() {
        return office;
    }
```

Brandon London Project 4.
2261-001

```
33      /** Return office */
34      public int getOffice() {
35          return office;
36      }
37
38      /** Return salaray */
39      public String getSalary() {
40          return String.format("%.2f", salary);
41      }
42
43      /** Return date hired */
44      public String getDateHired() {
45          return dateHired.getMonth() + "/" + dateHired.getDay()
46                  + "/" + dateHired.getYear();
47      }
48
49      /** Set new office */
50      public void setOffice(int office) {
51          this.office = office;
52      }
53
54      /** Set new salary */
55      public void setSalary(double salary) {
56          this.salary = salary;
57      }
58
59      /** Set new dateHired */
60      public void setDateHired() {
61          dateHired = new MyDate();
62      }
63
64      /** Return a string discription of the class */
65      public String toString() {
66          return super.toString() + "\nOffice: " + office +
67                  "\nSalary: $" + getSalary() + "\nDate hired: " + getDateHired();
68      }
69  }
```

Brandon London Project 4.
2261-001

```java
/**********************************************************************
* (The ComparableCircle class) Define a class named ComparableCircle that        *
* extends Circle and implements Comparable. Draw the UML diagram and implement   *
* the compareTo method to compare the circles on the basis of area. Write a test *
* class to find the larger of two instances of ComparableCircle objects.         *
**********************************************************************/
public class Exercise_13_06 {
    /** Main method */
    public static void main(String[] args) {
        // Create two instances of ComparableCircle objects
        ComparableCircle comparableCircle1 = new ComparableCircle(12.5);
        ComparableCircle comparableCircle2 = new ComparableCircle(18.3);

        // Display comparableCircles
        System.out.println("\nComparableCircle1:");
        System.out.println(comparableCircle1);
        System.out.println("\nComparableCircle2:");
        System.out.println(comparableCircle2);

        // Find and display the larger of the two ComparableCircle objects
        System.out.println((comparableCircle1.compareTo(comparableCircle2) == 1
            ? "\nComparableCircle1 " : "\nComparableCircle2 ") +
            "is the larger of the two Circles");
    }
}
```

```java
public class Exercise_11_02 {
    // Main method
    public static void main(String[] args) {
        // Create a Person, Student, Employee, Faculty, and Staff objects
        Person person = new Person("Steve", "123 ABC Street",
            "123458978541", "stevenhawking@aol.com");

        Student student = new Student("Sally", "13423 nowhere st.", "7894651616",
            "hit420@aol.com", Student.FRESHMAN);

        Employee employee = new Employee("IamTheNight", "2834 weast st", "66642498952",
            "Batman@Gmail.com", 910, 60000);

        Faculty faculty = new Faculty("SueMe", "28 yeet st", "66666666666",
            "NOextraCredit@aol.com", 101, 50000, "4pm to 6pm", "Professor");

        Staff staff = new Staff("Tom", "90 home ave", "2343234 9000",
            "tommymickil@aol.com", 12, 65000, "Executive Assistant");

        // Invoke toString of Person, Student, Employee, Faculty and Staff
        System.out.println(person.toString());
        System.out.println(student.toString());
        System.out.println(employee.toString());
        System.out.println(faculty.toString());
        System.out.println(staff.toString());
    }
}
```