



Zayed University
Department of Computer Science
SWE-320 Software Engineering
1st Semester 2017-2018
Submission Date: 18 Nov 2017

Assignment 2

Authors

Name	ID	Section

1 Problem Description

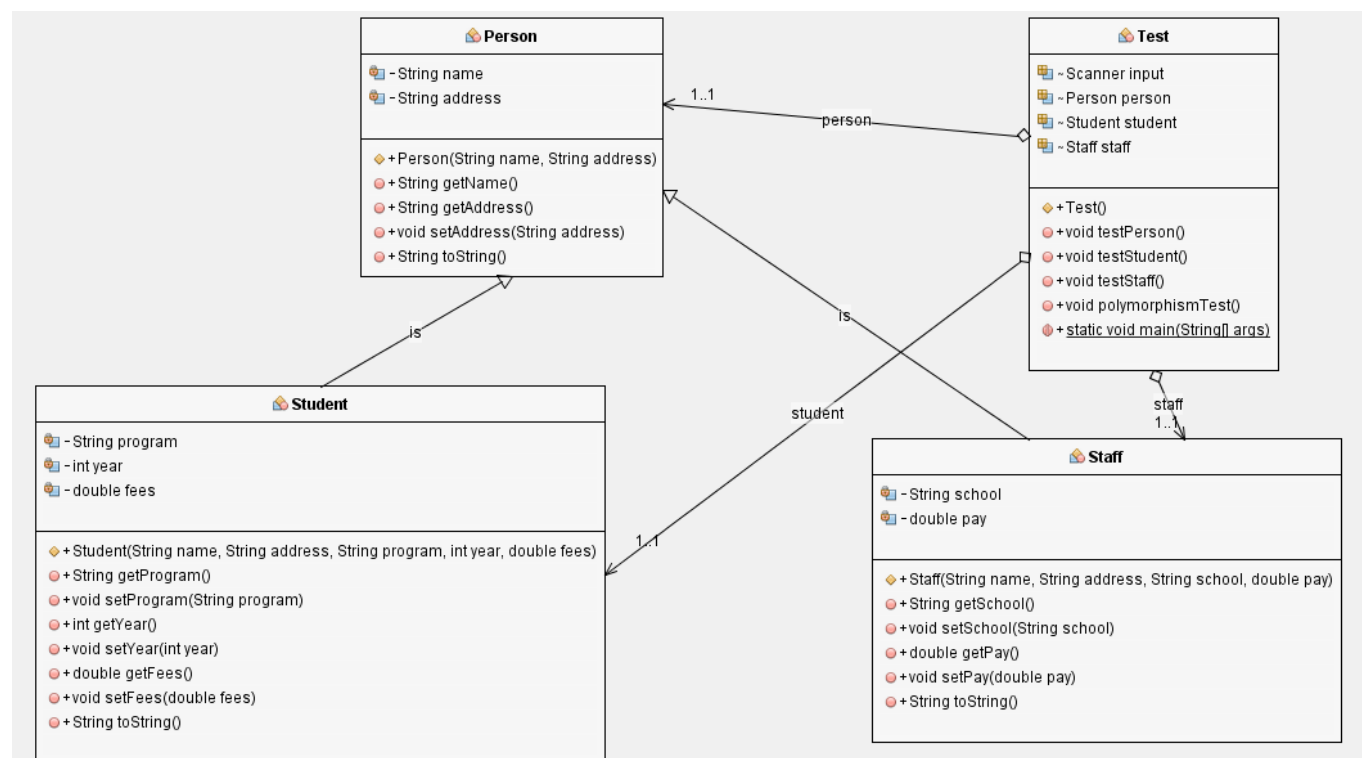
This project involved Inheritance and Polymorphism. It was a good project to get grasp on basic OOP concepts. We used standard ways to assign and retrieve values of instance variables using getters and setters methods. We used standard naming conventions for classes and variables. We learned a lot from this project.

2 Problem Analysis

Attribute Names	Attribute Types
name	String
address	String
program	String
year	int
fees	double
school	String
pay	double
input	Scanner
person	Person
student	Student
staff	Staff

3 Solution Design

UML Class diagram



4 Implementation

4.1 Person

```
public class Person {  
    private String name;  
    private String address;  
  
    public Person(String name, String address){  
        this.name = name;  
        this.address = address;  
    }  
  
    public String getName() {  
        return name;  
    }  
  
    public String getAddress() {  
        return address;  
    }  
  
    public void setAddress(String address) {  
        this.address = address;  
    }  
  
    @Override  
    public String toString() {  
        return "Person[" + "name = " + name + ", address = " + address + "];"  
    }  
}
```

4.2 Student

```
public class Student extends Person {  
  
    private String program;  
    private int year;  
    private double fees;  
  
    public Student(String name, String address, String program, int year, double fees) {  
        super(name, address);  
        this.program = program;  
        this.year = year;  
        this.fees = fees;  
    }  
  
    public String getProgram() {  
        return program;  
    }  
  
    public void setProgram(String program) {
```

```
        this.program = program;
    }

    public int getYear() {
        return year;
    }

    public void setYear(int year) {
        this.year = year;
    }

    public double getFees() {
        return fees;
    }

    public void setFees(double fees) {
        this.fees = fees;
    }

    @Override
    public String toString() {
        return "Student[Person[name=" + getName() + ", address=" + getAddress() + "], program=" +
        getProgram() + ", year=" + getYear() + ", fees=" + getFees() + ']';
    }
}
```

4.3 Staff

```
public class Staff extends Person {

    private String school;
    private double pay;

    public Staff(String name, String address, String school, double pay) {
        super(name, address);
        this.school = school;
        this.pay = pay;
    }

    public String getSchool() {
        return school;
    }

    public void setSchool(String school) {
        this.school = school;
    }

    public double getPay() {
        return pay;
    }
}
```

```

    public void setPay(double pay) {
        this.pay = pay;
    }

    @Override
    public String toString() {
        return "Staff[Person[name=" + getName() + ", address=" + getAddress() + "], school=" + school
+ ", pay=" + pay + "];"
    }
}

```

4.4 Test

```
import java.util.Scanner;
```

```

public class Test {

    Scanner input = new Scanner(System.in);

    Person person;//attribute person will store Person class reference
    Student student;// attribute student will store Student class reference
    Staff staff;//attribute staff will store Staff class reference

    public Test() {
        person = new Person("Ali Abdul Rahman", "UAE");
        student = new Student("Ali Abdul Rahman", "UAE", "Computer Science", 2017, 2000.0);
        staff = new Staff("Salman", "UAE", "ZU", 3500.0);
        testPerson();
        testStudent();
        testStaff();
        polymorphismTest();
    }

    //#####
    #####//
    //In this method we will do some communication with Person class
    public void testPerson() {

        System.out.println(person.toString());//printing String representation of the Person object

        System.out.println("Lets change " + person.getName() + "'s address using setter");
        System.out.print("Enter " + person.getName() + "'s address: ");
        person.setAddress(input.nextLine());//setting Person's address to user entered address

        System.out.println(person.toString() + "\n");//Printing updated values of the Person object
    }
}

```

```

//#####
#####//
//In this method we will do some communication with Student class
public void testStudent() {
    System.out.println("*****\n*   Student   *\n*****");
    System.out.println(student.toString()); //printing String representation of the Student object

    System.out.println("Lets test Student class once again using setters");
    System.out.print("Enter " + student.getName() + "'s address: ");
    student.setAddress(input.nextLine()); //setting address to user entered address

    System.out.print("Enter " + student.getName() + "'s program: ");
    student.setProgram(input.nextLine()); //setting name to user entered name

    try {
        System.out.print("Enter " + student.getName() + "'s year: ");
        student.setYear(input.nextInt()); //setting year to user entered year

        System.out.print("Enter " + student.getName() + "'s fees: ");
        student.setFees(input.nextDouble()); //setting fees to user entered fees
    } catch (Exception inputMismatchException) { //If user enters String instead of int or double for
year or fees then inputMismatchException will be thrown
        System.out.println("Please enter valid values next time");
    }
    System.out.println(student.toString() + "\n"); //printing new values of Student object, \n shifts
cursor to new line.
}

//#####
#####//
//In this method we will do some communication with Staff class
public void testStaff() {
    System.out.println("*****\n*   Staff   *\n*****");
    System.out.println(staff.toString()); //printing String representation of the Staff object

    System.out.println("Lets test Staff class once again using setters");
    System.out.print("Enter " + staff.getName() + "'s address: ");
    staff.setAddress(input.nextLine()); //setting address to user entered address

    System.out.print("Enter " + staff.getName() + "'s school: ");
    staff.setSchool(input.nextLine()); //setting school to user entered school

    try {
        System.out.print("Enter " + staff.getName() + "'s pay: ");
        staff.setPay(input.nextDouble()); //setting pay to user entered pay
    } catch (Exception e) { //If user enters String instead of int or double for pay then
inputMismatchException will be thrown

```

```

        System.out.println("Please enter valid values next time");
    }

    System.out.println(staff.toString() + "\n");
}

//#####
#####//
//In this method we will update address of student and staff using Person reference
//The beauty of polymorphism is that an object can take many forms.
public void polymorphismTest() {
    // Instantiating child_class(Student) using parent_class(Person) reference
    System.out.println("*****\n* Polymorphism *\n*****");
    Person stuPerson = new Student("Abdullah", "UAE", "Software Engineering", 2017, 1500.0);
    System.out.println(stuPerson.toString()); //printing String representation of the Student object

    System.out.println("Lets update attribute 'address' of Student(Child class) using
    Person(Parent) class reference");
    System.out.print("Enter " + stuPerson.getName() + "'s address: ");
    input.nextLine();
    stuPerson.setAddress(input.nextLine()); //setting address to user entered address

    //Now lets see updated address in Student class
    System.out.println(stuPerson.toString() + "\n");

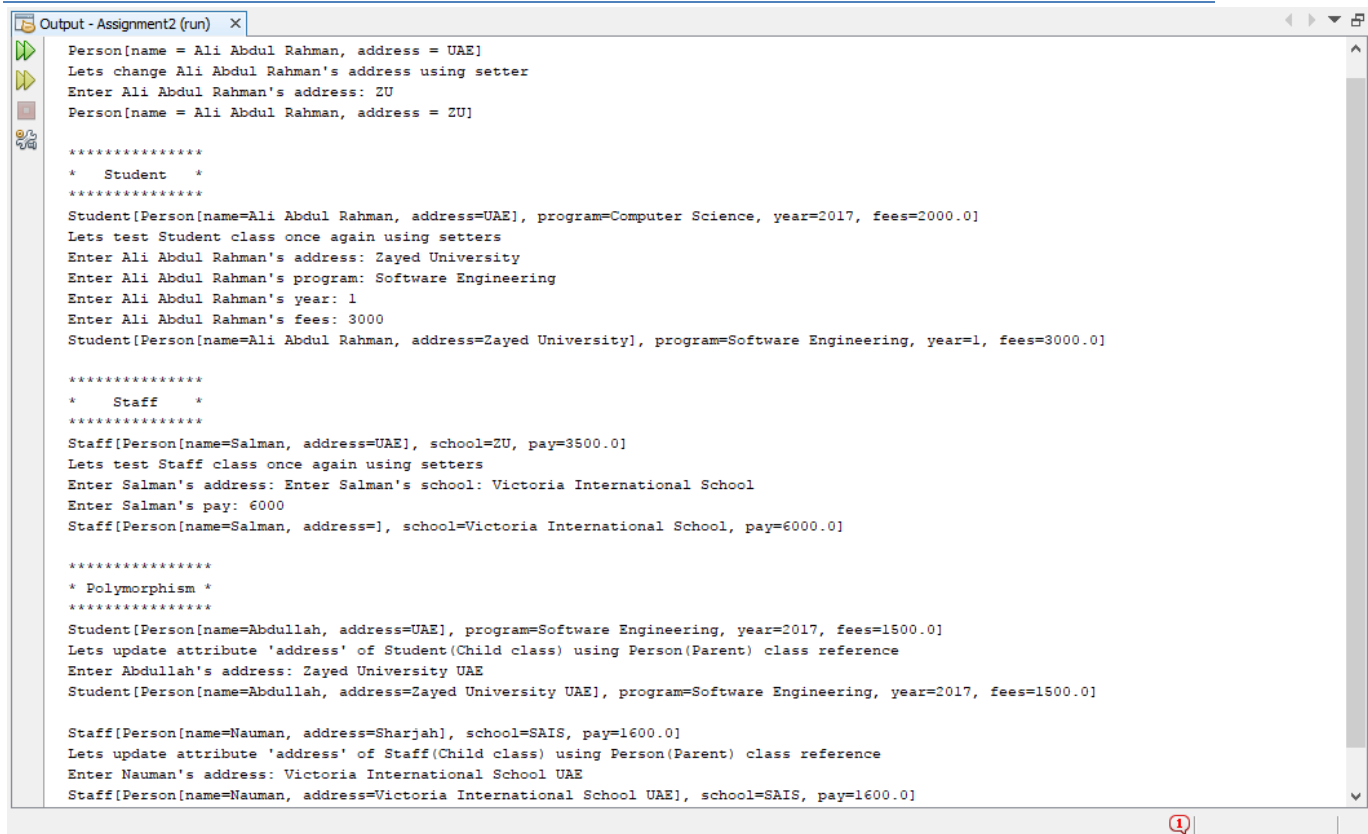
    //*****
    *****//
    // Instantiating child_class(Staff) using parent_class(Person) reference
    Person staPerson = new Staff("Nauman", "Sharjah", "SAIS", 1600.0);
    System.out.println(staPerson.toString()); //printing String representation of the Student object
    System.out.println("Lets update attribute 'address' of Staff(Child class) using Person(Parent)
    class reference");
    System.out.print("Enter " + staPerson.getName() + "'s address: ");
    staPerson.setAddress(input.nextLine()); //setting address to user entered address

    //Now lets see updated address in Staff class
    System.out.println(staPerson.toString());
}

public static void main(String[] args) {
    new Test(); //instantiating current class instance
}
}

```

5 Evaluation



```

Output - Assignment2 (run) x
Person[name = Ali Abdul Rahman, address = UAE]
Lets change Ali Abdul Rahman's address using setter
Enter Ali Abdul Rahman's address: ZU
Person[name = Ali Abdul Rahman, address = ZU]

*****
* Student *
*****
Student[Person[name=Ali Abdul Rahman, address=UAE], program=Computer Science, year=2017, fees=2000.0]
Lets test Student class once again using setters
Enter Ali Abdul Rahman's address: Zayed University
Enter Ali Abdul Rahman's program: Software Engineering
Enter Ali Abdul Rahman's year: 1
Enter Ali Abdul Rahman's fees: 3000
Student[Person[name=Ali Abdul Rahman, address=Zayed University], program=Software Engineering, year=1, fees=3000.0]

*****
* Staff *
*****
Staff[Person[name=Salman, address=UAE], school=ZU, pay=3500.0]
Lets test Staff class once again using setters
Enter Salman's address: Enter Salman's school: Victoria International School
Enter Salman's pay: 6000
Staff[Person[name=Salman, address=], school=Victoria International School, pay=6000.0]

*****
* Polymorphism *
*****
Student[Person[name=Abdullah, address=UAE], program=Software Engineering, year=2017, fees=1500.0]
Lets update attribute 'address' of Student(Child class) using Person(Parent) class reference
Enter Abdullah's address: Zayed University UAE
Student[Person[name=Abdullah, address=Zayed University UAE], program=Software Engineering, year=2017, fees=1500.0]

Staff[Person[name=Nauman, address=Sharjah], school=SAIS, pay=1600.0]
Lets update attribute 'address' of Staff(Child class) using Person(Parent) class reference
Enter Nauman's address: Victoria International School UAE
Staff[Person[name=Nauman, address=Victoria International School UAE], school=SAIS, pay=1600.0]

```

6 Self Reflection

1. How did you work on the project?

Answer: First i draw class diagram then i implemented whole project from that class diagram.

2. What did you learn out of the project?

Answer: I learned about Inheritance and Polymorphism in this project.

3. How much time did you spend working on the project?

Answer: I spent 6 hours working on this project.

4. And what is your personal opinion about the project?

Answer: Overall this project was very helpful in understanding some of the basic concepts of java. It was awesome.