OOP Lab-10 Task

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

Lab10: Abstract Class and Interface

To gain experience with:

1. **Abstract Classes and their purpose**
2. **Interfaces and their purpose**
3. **Exercise for practice**

1. Abstract Classes (in Java)

An abstract class usually defines a concept. Abstract classes have no implementation and they are only used as generic superclasses. An abstract class may contain:

- An abstract method (no implementation)
- A non-abstract method (or concrete method)
- Instance variables

Note that it is not necessary for an abstract class to have an abstract method. An abstract class is always preceded by the keyword **abstract.** As an example we provide the following hierarchy of an abstract class `Shape` and its non-abstract children:

In this lab we'll walk through the following class hierarchy:

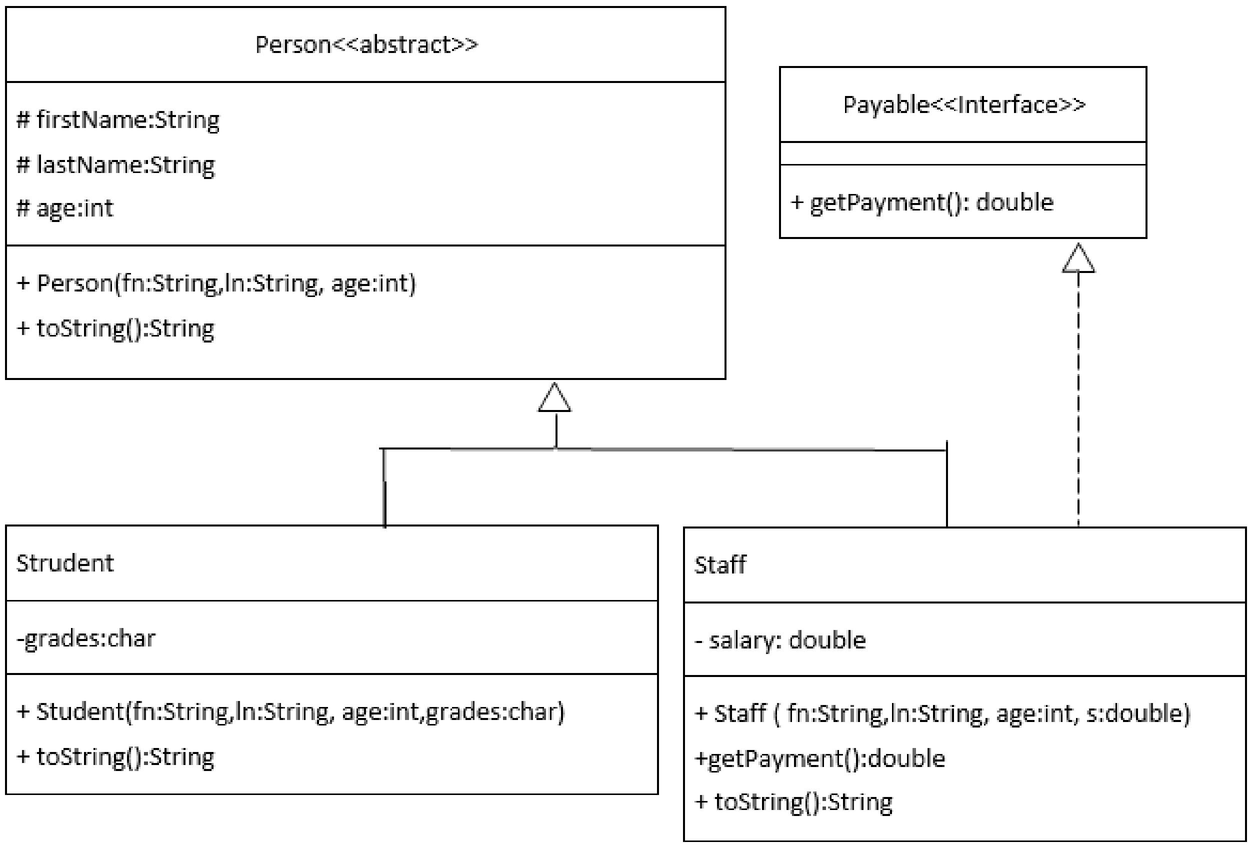
2. Interfaces (in Java)

Interfaces in java are a solution to the multiple-inheritance problem. In instances where a class exhibits a behavior of more than one class, interfaces are used.

Interfaces have only abstract methods and final (constant) variables. All methods in an interface are by default public and abstract.

Take Home Assignment

Consider the following UML class diagram:



- Consider an **abstract class Person** with three private attributes FirstName,LastName and Age. It has one abstract method

  String toString();
- Also, consider an **interface Payable** that contains only one method

  double getPayment().
- Class **Staff** has an additional attribute that is **salary**, which represents the monthly salary.
- Class **Student**has an additional attribute that is **grades**, which represents the grade character like A, B, C, D or F.
- In addition, the class **Staff** implements the interface **Payable**.
- The payment of the staff must return its **annual salary** (salary *12).
- The toString method of the each class must return all attributes of the class in text format. For a student, it returns the following string
  - Student: *FirstName*, *LastName*, *Age*, Grade
- For the Staff, it must return the following string
  - Staff: *FirstName, LastName, Age, Salary*.

Write the application class and create 3 objects of Student and 3 objects of Staff class and print the details.

Code:

```
package com.mycompany.interfacesapplication;
```

Person Class

```
abstract class Person{
    //data members
    protected String firstName;
    protected String lastName;
    protected int age;
    //Constructor
    public Person(String firstName, String lastName, int age) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.age = age;
    }
    //method
    abstract void display();
}
```

Student Class

```
class Student extends Person{
    private char grade;
    //getter

    public char getGrade() {
        return grade;
    }
    //setter

    public void setGrade(char grade) {
        this.grade = grade;
    }
    //constructor

    public Student(String firstName, String lastName, int age, char grade) {
       super(firstName,lastName,age);
        this.grade = grade;
    }
    //method
    @Override
    void display() {
        System.out.println("======== Student Info ======== \nFirst Name : 
"+firstName+"\n"+"Last Name : "+lastName+"\n"+"Age : "+age
        +"\nGrade : "+grade);
    }
}
```

Staff Class

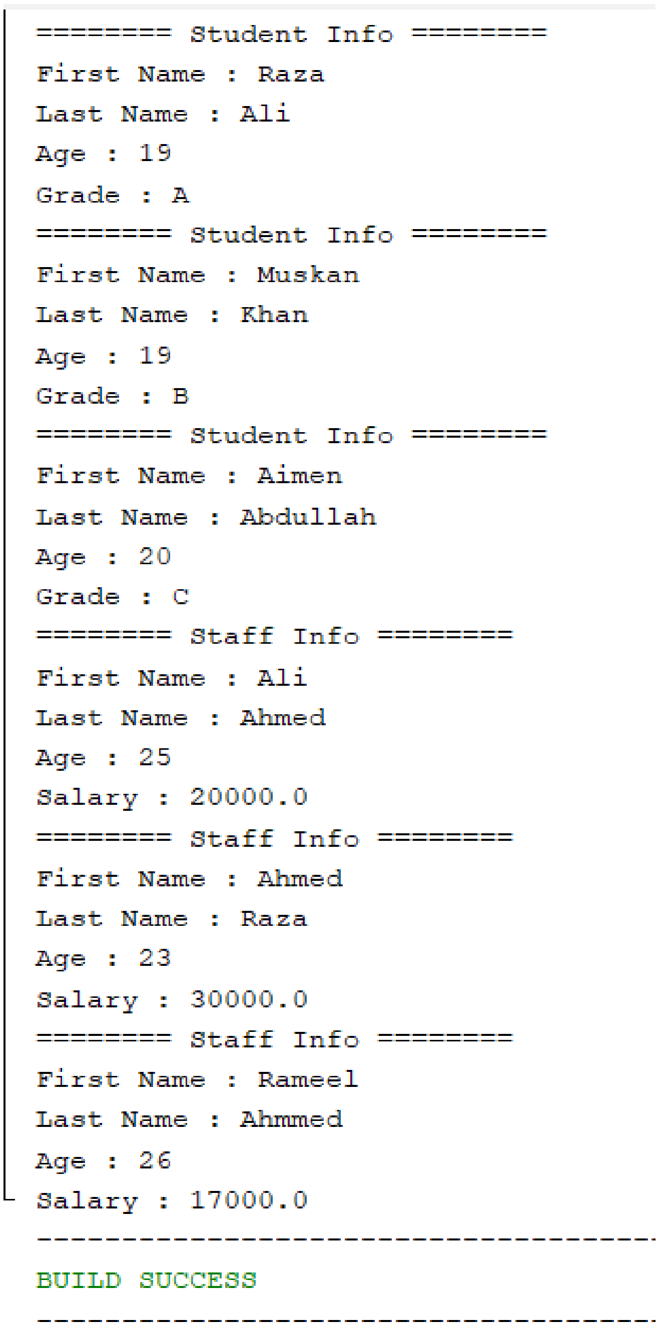
```
class Staff extends Person implements Payable{
    private double salary;
    //constructor
    public Staff(double salary, String firstName, String lastName, int age) {
        super(firstName, lastName, age);
        this.salary = salary;
    }
    //method
    @Override
    void display() {
        System.out.println("======== Staff Info ======== \nFirst Name : "+firstName+"\n"+"Last 
Name : "+lastName+"\n"+"Age : "+age
        +"\nSalary : "+salary);
    }
    public double getPayment(){
        return this.salary;
    }
}
```

Interface

```
interface Payable{
    double getPayment();
}
public class InterfacesApplication {

    public static void main(String[] args) {
       //three objs of Student
        Student student1 = new Student("Raza","Ali",19,'A');
        Student student2 = new Student("Muskan","Khan",19,'B');
        Student student3 = new Student("Aimen","Abdullah",20,'C');
        //three objs of Staff
        Staff member1 = new Staff(20000,"Ali","Ahmed",25);
        Staff member2 = new Staff(30000,"Ahmed","Raza",23);
        Staff member3 = new Staff(17000,"Rameel","Ahmmed",26);
        //printing the details of Students
        student1.display();
        student2.display();
        student3.display();
        //printing the details of Staff
        member1.display();
        member2.display();
        member3.display();
    }
}
```

Output:



```
======== Student Info ========
First Name : Raza
Last Name : Ali
Age : 19
Grade : A
======== Student Info ========
First Name : Muskan
Last Name : Khan
Age : 19
Grade : B
======== Student Info ========
First Name : Aimen
Last Name : Abdullah
Age : 20
Grade : C
======== Staff Info ========
First Name : Ali
Last Name : Ahmed
Age : 25
Salary : 20000.0
======== Staff Info ========
First Name : Ahmed
Last Name : Raza
Age : 23
Salary : 30000.0
======== Staff Info ========
First Name : Rameel
Last Name : Ahmmed
Age : 26
Salary : 17000.0
------------------------------------
BUILD SUCCESS
------------------------------------
```