Traccia di Object Orientation – 10 Settembre 2020

Esercizio 1

Vi è commissionata la realizzazione di una piattaforma che permetta ai gestori di un ristorante di informatizzare il tracciamento dei propri avventori in ottemperanza alle recenti disposizioni anti-contagio.
Il sistema permette a un cameriere di registrare l'arrivo di una tavolata. In questa fase, il sistema tiene traccia della data e orario di arrivo, del tavolo assegnato, nonché dei nominativi e dei numeri di telefonino di tutti gli avventori. Ad un tavolo, possono sedere al più 6 persone. Successivamente, nel caso uno dei clienti risulti positivo al Covid-19, il sistema permette ad un responsabile della struttura di inserire la segnalazione di contagio di uno degli avventori presente in una certa data. In seguito all'inserimento della segnalazione di contagio, il sistema invia automaticamente un SMS di avviso a tutti i clienti che hanno visitato il locale il giorno della visita del cliente infetto, e nelle date successive, fino alla data corrente.

Definire un Class Diagram per modellare la registrazione di una tavolata, inteso come modello di dominio.

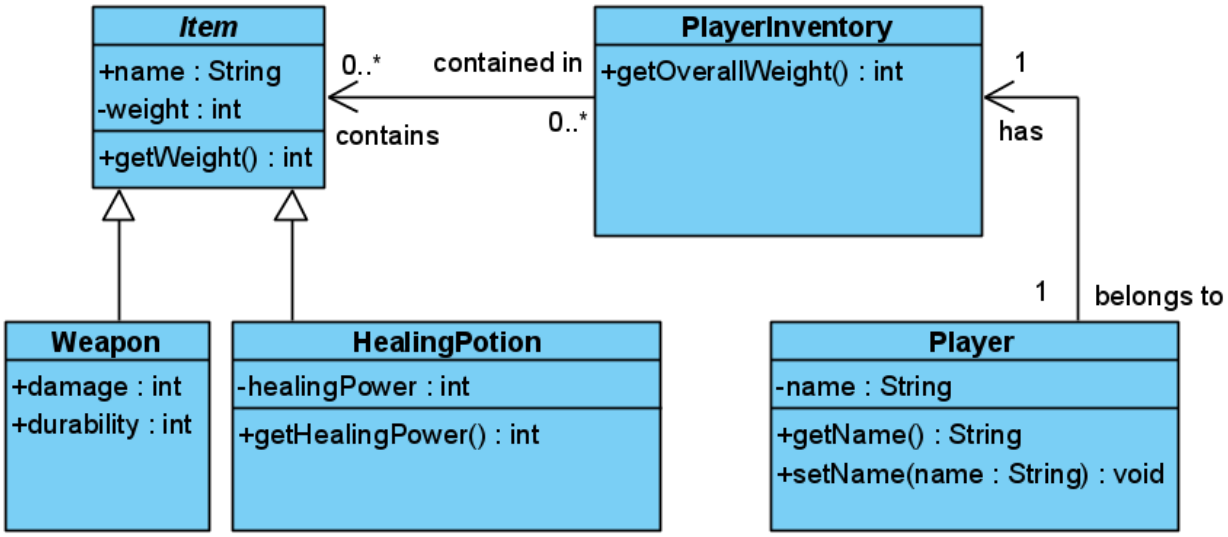
Esercizio 2

Vi è commissionata la realizzazione di una piattaforma che permetta ai gestori di un ristorante di informatizzare il tracciamento dei propri avventori in ottemperanza alle recenti disposizioni anti-contagio.
Il sistema permette a un cameriere di registrare l'arrivo di una tavolata. In questa fase, il sistema tiene traccia della data e orario di arrivo, del tavolo assegnato, nonché dei nominativi e dei numeri di telefonino di tutti gli avventori. Ad un tavolo, possono sedere al più 6 persone. Successivamente, nel caso uno dei clienti risulti positivo al Covid-19, il sistema permette ad un responsabile della struttura di inserire la segnalazione di contagio di uno degli avventori presente in una certa data. In seguito all'inserimento della segnalazione di contagio, il sistema invia automaticamente un SMS di avviso a tutti i clienti che hanno visitato il locale il giorno della visita del cliente infetto, e nelle date successive, fino alla data corrente.

In relazione all'esercizio precedente, fornire i Mock-up e un Sequence Diagram di analisi dello scenario relativo all'inserimento di una tavolata.

Esercizio 3

Si scriva tutto il codice Java che è possibile desumere dal seguente class diagram.



Esercizio 4

```
class Greeter {
  public void greet(Person p){
    System.out.println("Hi, "+p.getName());
  }

  public void greet(Pirate p){
    System.out.println("Ahoy, "+p.getName());
  }
}

class Person {
  String name;
  public Person(String name){
    this.name=name;
  }
  public String getName(){
    return name;
  }
}
```

```
class Pirate extends Person {
  public Pirate(String name){ super(name); }
  public String getName(){
    return name + " the pirate";
  }
}

public class TestClass{
  public void test(){
    Greeter g = new Greeter();

    Person person = new Person("Amy");
    Pirate pirate = new Pirate("Bill");
    Person test   = pirate;

    g.greet(person);
    g.greet(pirate);
    g.greet(test);
  }
}
```

Si consideri il codice Java sopra riportato.

- Il codice compila correttamente (si assuma che gli import necessari siano definiti)? In caso negativo, come si potrebbe modificare il codice affinché la compilazione abbia successo?
- Qual è (considerando anche le eventuali modifiche apportate al punto precedente) l'output su `stdout` di un'invocazione del metodo `test()` in `TestClass`?

Esercizio 5

```
class Greeter {
  public void greet(Person p){
    System.out.println("Hi, "+p.getName());
  }

  public void greet(Pirate p){
    System.out.println("Ahoy, "+p.getName());
  }
}

class Person {
  String name;
  public Person(String name){
    this.name=name;
  }
  public String getName(){
    return name;
  }
}
```

```
class Pirate extends Person {
  public Pirate(String name){ super(name); }
  public String getName(){
    return name + " the pirate";
  }
}

public class TestClass{
  public void test(){
    Greeter g = new Greeter();

    Person person = new Person("Amy");
    Pirate pirate = new Pirate("Bill");
    Person test   = pirate;

    g.greet(person);
    g.greet(pirate);
    g.greet(test);
  }
}
```

Si consideri il codice Java sopra riportato.

- È presente un esempio di *overriding*? Motivare la risposta. In caso affermativo, indicare in quale/quali parti di codice è presente.
- È Presente un esempio di *overloading*? Motivare la risposta. In caso affermativo, indicare in quale/quali parti di codice è presente.