Esercizio sull'ereditarietà

Esercizio

- Si definisca una classe `Dottore` i cui oggetti rappresentano le schede dei dottori di una clinica.
- Si derivi questa classe dalla classe `Persona` nella prossima slide
- Un dottore ha un nome, definito nella classe `Persona`, una specializzazione descritta tramite una stringa (per esempio Pediatra, Ostetrico, Medico generale e così via) e una parcella per le visite in ufficio (si usi il tipo double).
  Si definiscano gli appropriati costruttori, i metodi *get* e un metodo `equals`.

Esercizio

- Si definiscano due classi, `Paziente` e `Fattura`, i cui oggetti sono schede per la clinica. Si derivi `Paziente` dalla classe `Persona`
- Un `Paziente` ha un nome (definito nelle classe `Persona`) e un numero identificativo (si usi il tipo String).
- Si definiscano i costruttori appropriati, i metodi *get* e un metodo `equals`

Esercizio

- Un oggetto `Fattura` conterrà un oggetto `Paziente` e un oggetto `Dottore`.
- Si definiscano i costruttori appropriati, i metodi *get* e un metodo `equals` e un metodo `importo` che resituisce l'importo della fattura (uguale alla parcella del medico).
- Si scriva un programma di prova che crei almeno due pazienti, almeno due dottori e almeno due fatture e che visualizzi il totale dell'importo delle fatture.

Codice

```
class Persona {
    private String nome;

    public Persona() {
        nome = "Ancora nessun nome";
    }

    public Persona(String nomeIniziale) {
        nome = nomeIniziale;
    }

    public void setNome(String nuovoNome) {
        nome = nuovoNome;
    }

    public String getNome() {
        return nome;
    }

    public void scriviOutput() {
        System.out.println("Nome: " + nome);
    }

    public boolean equals(Persona altraPersona) {
        return equalsIgnoreCase(altraPersona.nome);
    }
}
```

Codice

```
class Dottore extends Persona {
    private String specializzazione;
    private double parcella;

    public Dottore() {
        super();
    }

    public Dottore(String nome, String type, double cost) {
        super(nome);
        specializzazione = type;
        parcella = cost;
    }

    public String getSpecializzazione() {
        return specializzazione;
    }

    public double getParcella() {
        return parcella;
    }

    public boolean equals(Dottore d) {
      return super.equals(d) && specializzazione.equals(d.specializzazione);
    }
}
```

Codice

```
lass Paziente extends Persona {
   private String id;

    public Paziente() {
        super();
    }

    public Paziente(String nome, String identità) {
        super(nome);
        id = identità;
    }

    public String getId() {
        return id;
    }

    public boolean equals(Paziente p) {
      return id.equals(p.id);
    }
}
```

Codice

```
class Fattura {
    private Dottore d;
    private Paziente p;

    public Fattura(Dottore dott, Paziente paz) {
        d = dott;
        p = paz;
    }

    public String getNomePaziente() {
        return p.getNome();
    }

    public String getIdPaziente() {
        return p.getId();
    }

    public String getNomeDottore() {
        return d.getNome();
    }

    public double Importo() {
        return d.getParcella();
    }

    public boolean equals(Fattura f) {
      return d.equals(f.d) && p.equals(f.p);
  }
}
```

Codice

```
public class es2 {
    public static void main(String[] args) {
        Paziente p1 = new Paziente("Franco Peppi", "183746kshdf");
        Paziente p2 = new Paziente("Mauro Bressi", "9483tbchwyf");

        Dottore d1 = new Dottore("Franco Milani", "Pediatra", 1234.44);
        Dottore d2 = new Dottore("Giuseppe Neri", "Chirurgo", 99994.44);

        Fattura f1 = new Fattura(d1, p1);
        Fattura f2 = new Fattura(d2, p2);

        System.out.println("Totale importo fatture: " + (f1.Importo() + f2.Importo()));
    }
}
```