

Activitat 2. Refacció i Optimització de codi

1) Donat el següent codi:

```
import java.util.ArrayList;
import java.util.Collection;

public class CostPersonal {

    static float CostDelPersonal(Treballador treballadors[]) {
        float costFinal = 0;
        Treballador treballador;

        for (int i = 0; i < treballadors.length; i++) {
            treballador = treballadors[i];

            if (treballador.getTipusTreballador() == Treballador.DIRECTOR || \
                treballador.getTipusTreballador() == Treballador.SUBDIRECTOR) {
                costFinal = costFinal + treballador.getNomina();
            } else {
                costFinal = costFinal + treballador.getNomina() + \
                    (treballador.getHoresExtres() * 20);
            }
        }
        return costFinal;
    }
}
```

- Creeu un projecte que implementi aquesta classe. Creeu també la classe treballador a fi, que el projecte sigui funcional. No codifiqueu els mètodes de la classe Treballador
- Genereu la documentació de la classe CostPersonal. Genereu la documentació del mètode CostDelPersonal indicant que realitza. (1 punt)
- Creeu les proves unitàries que testegin aquest mètode. (1 punt)
- Utilitzeu les tècniques de refacció més usuals per modificar el codi (extraieu constants, descomponeu el codi en mètodes estàtics més senzills de la classe CostPersonal (per exemple: calculaSouTreballador, es Directiu). (1 punt)
- Creeu les proves unitàries pels mètodes obtinguts de la refacció de d). (1 punt)
- Expliqueu com milloraríeu el disseny de les vostres classes emprant herència i polimorfisme. Recodifiqueu de nou el codi del mètode CostPersonal amb el nou disseny. (2,5 punts)

2) El següent mètode de la classe Nif, (atributs privats String nif i String missatge,...) comprova si un nif és vàlid:

```
public void esValid() {  
  
    char[] llista = {'T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X', 'B', 'N', 'J', 'Z', 'S', 'Q', 'V', 'H', 'L', 'C',  
    'K', 'E'};  
  
    int num = 0;  
    String letra;  
    String numero;  
    char[] charnif;  
  
    letra = nif.substring(nif.length() - 1);  
    numero = nif.substring(0, nif.length() - 1);  
  
    // primer mirem que la cadena tingui 8 o 9 caràcters:  
  
    if (nif.length() < 8 || nif.length() > 9) {  
        missatge = "Format Incorrecte"  
    } // Després mirem que el número sigui convertible a enter.  
    else {  
        try {  
            num = Integer.parseInt(numero);  
  
            if (String.valueOf(llista[num % 23]).equals(letra)) {  
                missatge = "Nif Vàlid";  
            } else {  
                missatge = "Nif Invàlid";  
            }  
        }  
  
        } catch (NumberFormatException e) {  
            missatge = "Format Incorrecte";  
        }  
    }  
}
```

Utilitzeu les tècniques més habituals de refacció i refeu aquest codi (podeu fer les modificacions/suposicions de la classe nif que creieu adients). Al final heu de posar el codi final d'aquest mètode i documentar l'estructura global de la classe.

a) Refacció del codi: 1,5

b) Documentació general de la Classe: 1

c) Completeu la classe Nif i testegeu el mètode esValid() emprant la llibreria JUnit: 1