Laboratorio 2

Programación Orientada a Objetos, equals, Genéricos

Estructura de Dades

2n Curs GEI
Grup 2

3/11/2020

Ton Llucià Senserrich
47125160T

Eduard Sales Jové
49539818A

Exercici 1: Jerarquia de clases i equals

Apartat 1a:

Equals classe Person:

La primera línia mira is l'objecte que volem comprovar té la mateixa referència que el que ens passen per paràmetre, en cas que tinguin la mateixa referència, serà obvi que són iguals, per tant retorna true.

La segona comprova tant que l'objecte que ens passen per paràmetre no sigui null i, a més que els dos objectes que volem comprovar siguin de la mateixa classe, en cas que no es compleixi alguna d'aquestes dues condicions poden assegurar que l'objecte no és igual i per tant retornem false.

A la tercera línia només entrarem en cas de no haver entrat en cap dels ifs anteriors, ja que no podem modificar l'objecte que ens passen per paràmetre, el que farem és utilitzar una variable auxiliar anomenada person i la castejarem a tipus Person, d'aquesta manera a la quarta línia podrem cridar a la funció estàtica de la classe Object per tal de comparar els dos strings.

Equals classe Employee:

La primera Lina, igual que abans, mira si els dos objectes a comprovar si tenen la mateixa referència.
En la següent línia mira que el que ens passen per paràmetre no sigui null, i que els dos objectes a comprovar pertanyen a la mateixa classe.
Si no compleix cap de les condicions anteriors, arribarà a la línia tres on, castejem la variable que ens passen per paràmetre a la classe Employee mitjançant una variable auxiliar.
En l'última línia retornem un boolea, depenent de si el salari de la persona que estem comprovant és igual al de l'employee que ens passen per paràmetre.
La diferència entre els dos equals és que cada un casteja a la classe a la qual pertany i una compara el salari i l'altre el nom.

Prova notInteroperable:

Com hem vist en les línies 2 de les funcions anteriors una de les condicions era que els dos objectes havien de ser de la mateixa classe, al test notInteroperable, p és un objecte de la

classe Person i e de la classe Employee, per tant al no ser de la mateixa classe és obvi que la funció equals retornarà fals.

Apartat 1b:

Equals classe Person apartat b:

En la primera línia mirem si els dos objectes són el mateix objecte, o si l'objecte que ens passen per paràmetre és o no, instància de Person.
És en la segona línia de codi on trobem la diferència dels dos equals, ja que l'anterior mirava si ens passaven un objecte null o un d'una classe que no tocava, en aquest cas mirarà si l'objecte es instància de Person, en cas que no, retornarà fals.
D'aquesta manera t'assegures que:
1- Els objectes que passin aquesta comprovació, podem assegurar que seran instàncies de Person.
2- Ja no ens fa falta fer la comprovació que l'objecte sigui null, ja que un objecte null mai serà instància de Person.

Utilitzarem una variable auxiliar anomenada person i la castejarem a tipus Person, ja que no podem modificar l'objecte que ens passen per paràmetre, i finalment retornarem un boolea.

Equals classe Employee apartat b

Per començar farem la primera comprovació per a mira si l'objecte que volem comprovar és el mateix que el rebut per paràmetre.
En la segona línia mirarem si l'objecte que ens passen es instància d'Employee, en aquest cas com que Employee exten de Person, tant employee com person seran instància d'Employee, per tant la segona línia es compleix sempre que li passis un objecte de tipus Person o Employee.
En la tercera línia farem servir la funció if (!super.equals(o)) return false;
La qual farà una crida a la funció equals de la classe pare, en aquest cas la classe Person, per tal de comprovar si el nom és el mateix.

Apartat 1c

L'enunciat ens demana que un objecte tipus Employee es pugui comparar a un objecte de tipus Person així que canviarem la segona línia utilitzada en la funció de l'apartat anterior (if

(!(o instanceof Employee)) return false) perquè aquest equals compari tant Employee com Person.
Per aconseguir això farem un canvi en l'if mencionat anteriorment, en lloc que retorni false en cas que el que volem comparar no sigui una instància d'Employee, el que farem és que quan no ho sigui cridarem a la funció equals de la classe pare, en aquest cas Person, això ho farem amb la instrucció (return super.equals(o)), d'aquesta manera quan ens passin un Person com que no serà instància d'Employee, anirem a buscar a la funció de Person que comparará els dos objectes com si fossin de la classe Person.
El cas en què no entri en l'if mencionat anteriorment, voldrà dir que els dos objectes són de la mateixa classe, Employee.
Per tant primerament cridarem l'equals de la classe pare per comprovar que els noms siguin iguals i finalment comprovem el salari.

-Vam tenir problemes al moment de desenvolupar els tests, concretament ens va costar trobar un cas que no complís la transitivitat, ja que nosaltres estàvem buscant trencar la transitivitat amb 2 objectes Employee amb exactament els mateixos valors, llavors és impossible trencar-la, al final vam pensar en què havíem de canviar alguna cosa entre els 2 Employees, per tant vam canviar el salari i vam posar nombres diferents.

Exercici 2: Iteradors i interficie

La funció countIf el que fa és iterar sobre d'un iterador de manera que recorrem tots els seus elements i contem tots aquells elements que retornin true en executar la funció test de l'objecte anomenat test recollit per paràmetre.

Exercici 3: Comparadors i comodins

En la nostra generalització fent servir comodins utilitza dos comodins diferents, ja que es requereix que la funció tracti amb 2 paràmetres de diferent classe, i funciona en tots els casos.
-cas 1: En el primer cas ens donen un iterador d'Employees i un objecte Person, i la nostra generalització funciona, ja que tant E serà de tipus Employee i la N serà de tipus Person
-cas 2: En el segon cas ens donen un iterador de Person i un objecte Employee, i la nostra generalització funciona, ja que E serà de tipus Person i N serà de tipus Employee.

-cas 3: En el tercer cas ens donen un iterador d'Employees i un objecte Employee, i la nostra generalització funciona, ja que tant E serà de tipus Employee i la N serà de tipus Employee