UNIVERSITAT DE LLEIDA

Escola Politècnica Superior

Enginyeria informàtica

Estructures de dades

PRÀCTICA 2
EQUALS I GENÈRICS

Marc Cervera Rosell 47980320C

Diego Martínez de Sanjuan 48051017S

Data de lliurament: dia 26 de Octubre de 2021.

INDEX

1. Exercici 1

1.1. Apartat 1a

En primer lloc, dins del equals de Person, es comprova si 'this' i 'o' referencien el mateix objecte, en cas que sí, òbviament, es retorna true. En segon lloc, es realitzen dues comprovacions diferents. La primera comprova si la referència 'o' apunta a null. La segona, comprova si els objectes referenciats per 'this' i 'o' són de classes diferents. En el cas que la referència 'o' apunti a null o els objectes pertanyin a classes diferents, es retronarà false. Seguidament, es defineix una nova referència que apunta al mateix objecte referenciat per 'o'. A aquesta nova referència se li assigna el tipus 'Person'. Finalment, retornem una crida al mètode estàtic 'equals' de la classe 'Objects'. El comportament d'aquest mètode es pot desglossar en 4 casos diferents:

1. Si els dos arguments són iguals entre si, es retorna true.
2. Si els dos arguments són diferents entre si, es retorna false.
3. Si els dos arguments són referències que apunten a null, es retorna true.
4. En cas que el primer argument no sigui una referència que apunta a null, s'invoca al mètode equals de la classe Object, que retorna true si i només si ambdós arguments fan referència al mateix objecte.

En la subclasse Employee, les dues primeres línies són les mateixes que en el mètode de la superclasse. És a dir, es comprova si 'this' i 'o' referencien el mateix objecte i es comprova si 'o' és una referència que apunta a 'null' o si els objectes referenciats per 'this' i 'o' pertanyen a classes diferents. En la tercera línia es troba la diferència principal entre els dos mètodes 'equals'. En aquesta línia, s'invoca al mètode 'equals' de la superclasse (Person) sobre l'objecte receptor, passant com a paràmetre la referència 'o'. Seguidament, es defineix una nova referència del tipus 'Employee', que apuntarà al mateix objecte referenciat per 'o'. Finalment, es retorna cert si i només si el salari (this) és igual al salari de la nova referència que s'acaba de crear.

El test notInteroperable passa de manera correcta, atès que s'està realitzant una comparació de dos objectes diferents. És a dir, s'està comparant una persona amb un empleat i, en el segon test, un empleat amb una persona. El mètode assertNotEquals, comprova si els objectes són diferents. Per tant, estem comparant una persona amb un empleat, però aquesta persona podria no ser un empleat.

1.2. Apartat 1b

Primerament, igual que en l'apartat anterior dins del equals de Person, es comporova si l'objecte referenciat per 'this' i l'objecte referenciat per 'o' són el mateix. En cas afirmatiu es retorna true, òbviament. Seguidament, es comprova si l'objecte referenciat per 'o' és una instància de 'Person' o d'alguna de les seves subclasses. En cas de no complir aquesta condició, es retorna false. A continuació, es crea una nova referència de tipus 'Person', aquesta apuntarà al mateix objecte refrenciat per 'o'. Finalment, es retorna una crida al mètode 'equals' de la classe 'Objects', el comportament del qual ja ha estat definit anteriorment.

En el equals de la classe Employee d'aquest apartat, les dues primeres línies són les mateixes que en la superclasse (comprovar si ambdues referències apunten al mateix objecte i comprovació de instanceof). Seguidament, s'invoca al mètode 'equals' de la superclasse 'Person', passant com a paràmetre l'objecte referenciat per 'o'. En cas que aquest mètode retorni fals, es retonarà fals com a resultat del mètode de la subclasse. A continuació, es crea una nova referència del tipus 'Employee' que apuntarà al mateix objecte que la referència 'o' i finalment, es retorna vertader si i

només si els dos salaris (el del 'this' i el de la nova referència) són iguals.

Els jocs de proves són els mateixos que en l'apartat anterior a excepció del mètode 'notInteroperable' que s'ha vist modificat a causa de l'acceptació de subclasses com a paràmetre del mètode 'equals'. Com s'especifica en l'informe, amb aquesta modificació es pot comparar una persona amb un empleat, però no es pot comparar un empleat amb una persona. Per tant, el mètode 'equals', tal com està codificat en aquest apartat, és semi-interoperable.

1.3. Apartat 1c

En la classe 'Employee', s'han realitzat un total de 3 canvis:

1. S'ha realitzat un canvi en l'operador 'instanceof'. En comptes de comprovar que 'o' sigui una instància de 'Employee' o quelcom inferior (cosa imopssible), ara es comprova que 'o' sigui una instància de 'Person' o inferior. Aquest canvi s'ha realitzar per poder realitzar la comparació empleat - persona, atès que 'Employee' no té sublcasses, i per tant sense aquest canvi el test fallaria.
2. Donat el canvi anterior, no es pot crear una referència de tipus 'Employee', s'ha de crear una referència de tipus 'Person'. Aquesta referència apuntarà al mateix objecte que 'o'.
3. Es retorna una crida al mètode 'equals' a nivell de nom. Per tant, es retornarà vertader si i només si els noms són iguals.

Un cop efectuats aquests canvis, el test 'interoperable' que proporciona l'enunciat, passa correctament.

També es demana que es realitzi un joc de proves que demostri la pèrdua de la transitivitat. Per tant, saben que aquesta propietat afirma que: $\forall \boldsymbol{x},\boldsymbol{y},\boldsymbol{z} \in \boldsymbol{A}: \boldsymbol{xRy\ i\ yRz} \rightarrow \boldsymbol{xRz}$ s'ha de trobar un exemple en què 'x' no estigui relacionat amb 'z'. Per tant, el que s'ha fet és crear dos empleats i una persona i comprovar que quan es comparen (a nivell de nom) el primer empleat amb la persona i el segon empleat amb la persona, es compleix que el primer empleat està relacionat amb la persona. En canvi, quan realitzem la comparació dels empleats a nivell de salari, es pot comprovar que, encara que ambdós empleats tinguin el mateix nom, tenen un salari diferent, per tant, no es tracta del mateix empleat. És per aquesta raó, que s'ha afegit el mètode 'getSalary()'.

2. Exercici 2

El mètode 'removeIf()', té un disseny força senzill. En primer lloc, es declara un bucle de tipus 'while' que donarà voltes mentre l'iterador tingui element següent, és a dir, donarà voltes mentre l'operació 'hasNext()' retorni 'true'. Dins d'aquest bucle, Es defineix una nova referència de tipus 'E' que apuntarà al valor que retorni l'operació 'next()'. La condició del bucle és la manera d'assegurar-se que el programa no intentarà fer una operació 'next()' quan l'iterador no pugui avançcar més. Seguidament, es comprova si el mètode test (el de l'interfície), retorna cert o fals. En cas de retornar cert, és a dir, compleix la condició, s'eliminarà aquell element de la llista, atès que compleix la condició especificada.

En la classe de tests, s'implementa el mètode de la interfície. En cada test s'implementa mitjançant una tècnica diferent.

En el primer test, s'implementa mitjançant una classe local. En el segon test, mitjançant una classe anònima i en el tercer mitjançant una funció lambda

3. Exercici 3

La primera part de l'exercici, consisteix a implementar el mètode 'compareTo()' en la classe 'Person'. La implementarció d'aquest mètode és ben senzilla, atès que només s'ha de retornar una crida al mètode 'compareToIgnoreCase()' amb el 'this.name' com a cadena que crida al mètode i el 'o.name' com a cadena paràmetre. El comportament del 'compareToIgnoreCase()' es divideix en:

1. Retorna un valor < 0 si la cadena que crida al mètode va primer, lexicogràficament.
2. Retorna 0 si les dues són lexicogràficament equivalents.
3. Retorna un valor > 0 si la cadena paràmetre va primer, lexicogràficament.

La segona part de l'exercici, consisteix a realitzar una classe de testing per comprovar el correcte funcionament de la implementació anterior.

La segona part de l'exercici, consisteix en la realització d'una classe de tests que comprovi que la correcta comparació entre noms independentment de les lletres majúscules i minúscules. Aquesta classe, en aquest cas, consta d'una totalitat de 3 tests diferents; El primer dels quals, realitza la comparació entre elements de la classe Person; el segon dels quals, compara elements de la classe Employee i el tercer i últim test, realitza comparacions mixtes, és a dir, realitza comparacions entre empleats i persones.

La tercera i última part d'aquest darrer exercici, consisteix a implementar una classe que contingui un mètode que dugui a terme la tasca descrita en l'enunciat de la pràctica. Com s'observa en el joc de proves proporcionat en l'enunciat, la classe s'anomena Comparisons i el mètode removeLower.