Conceptos Herencia

1. A)

Performs Person's tasks

Invoke Employee's

Performs Employee's task

Performs Faculty's tasks

En el código se puede apreciar que la clase Person es el padre de todas las clases, esta clase solo tiene como único método su constructor y este se encarga de mostrar por pantalla “Performs Person tasks”.

El hijo de Person es la clase Employee la cual extiende a Person y este contiene 2 metodos sobercargados, uno sin parámetros que se llama a si mismo pero con el parámetro “invoke employees” y muestra en pantalla “Performs Employees tasks”, el otro método recibe un string e imprime en pantalla el mismo String.

A partir de Employee se extiende Faculty , es decir que el padre e faculty es Employee, el constructor de Faculty imprime en pantalla “ Performs Faculty’s tasks”, en esta clase también se encuentra el main el cual crea un objeto tipo Faculty, como Faculty extiende de Employee , también es llamado su constructos, y este a su vez llama el constructor de Person, generando una pila que va en el siguiente orden  
-Constructor Person

-Constructor() Employee ( Este constructor llama al constructor (s) Employee)

-Constructor Faculty

2.

G

G 1 G 2

F

G 1 F 2 G 2

F

H 1 E 2

H 1

F

G 1 E 2

G 1

Todas la impresión por pantalla se realiza en el main en la clase ABCDMain, allí se crea un arreglo de objetos de clase G, unos son objetos pertenecientes a la clase G y otros son objetos de clases que se heredan de G, el ciclo se encarga de para cada objeto en ese arreglo, imprimir su “ToString” y después llamar los métodos 1 y 2.

Inicia con el objeto de clase G, llama su método toString (G) y después llama sus métodos 1 y 2 (G 1 G 2); después recorre el objeto de clase F, el método toString esta reescrito , por lo tanto ya no imprime (G) como su padre , sino imprime en pantalla (F), después alno tener método 1 , utiliza el de su padre “ method1 G” que imprime en pantalla (G 1) después llama al método 2 , F tiene sobrescrito el método de manera que ahora imprime en pantalla (F 2 ) y después de eso llama al método 2 de su papa (G) y es aquí donde se acaba la cuarta línea.

El ciclo procede a analizar el objeto de clase H, donde no existe un toString, por lo tanto busca toString en su padre (E) pero ya que E tampoco cuenta con un toString, recurre al toString del padre de su padre (F) el cual si cuenta con un toString y su salida es (F), después busca el método 1 que es un método reescrito en su propia clase e imprime en pantalla (H 1), después de esto busca el método2, pero como él no cuenta con ese método , lo busca en su padre (E) el cual si cuenta con este método e imprime (E 2) en pantalla, hasta aquí termina la sexta línea.

El ciclo for sigue a analizar el objeto de clase E, como no cuenta con un toString lo busca en su padre (F) e imprime (F), después al tampoco contra con el método1 lo busca en su padre, como no lo encuentra recurre al padre de su padre (G) e imprime (G 1) , el método 2 de E imprime (E 2) y llama a su método1, pero como este no se encuentra ni en el mismo ni en su padre, vuelve a recurrir al padre de su padre (G) e imprime nuevamente (G 1), dando final al programa