Ejercicios propuestos T1 - Kotlin

Título

Realizar los ejercicios de los enunciados propuestos en Kotlin

Tiempo estimado de ejecución

120 minutos.

Objetivo de la actividad:

- Entender el comportamiento del lenguaje kotlin.
- Entender la estructura de herencia y polimorfismo.

Recursos necesarios

- Un entorno de desarrollo (IntelliJ).
- Java Development Kit.

Enunciado

1. (MayorQue) Hágase un programa que lea dos enteros y compruebe si el primero es o no mayor que el segundo. En la salida se mostrará “El primero es mayor que el segundo” o “El primero no es mayor que el segundo”.

2. (Condiciones) Léase un número entero. Se comprobará si dicho número es o no múltiplo de 20, y también se comprobará si está o no entre -100 y 100. En la salida se mostrará uno de los siguientes mensajes:

   “Es múltiplo de 20 y está entre -100 y 100”. “Es múltiplo de 20 y no está entre -100 y 100”. “No es múltiplo de 20 y está entre -100 y 100”- “No es múltiplo de 20 y no está entre -100 y 100”.

3. Escribe un programa que muestre por consola (con un print) los números de 1 a 100 (ambos incluidos y con un salto de línea entre cada impresión), sustituyendo los siguientes:

   1. Múltiplos de 3 por la palabra "fizz".
   2. Múltiplos de 5 por la palabra "buzz".
   3. Múltiplos de 3 y de 5 a la vez por la palabra "fizzbuzz".

4. Crea un programa que genere un número aleatorio (entre 0 y 20) de entrada y lo guarde en una variable. El sistema irá pidiendo números al usuario hasta que coincida con el aleatorio generado (en caso de no coincidir mostrará el mensaje "Lo siento, inténtalo de nuevo").

Cuando coincida el sistema deberá mostrar el mensaje "Enhorabuena, has acertado el número en 5 intentos" (Adivina)

5. Modifica el ejercicio anterior para que una vez adivinado el número el sistema muestre el siguiente mensaje: "Quieres volver a jugar (S/N)":

6. Realizar una aplicación para la gestión de trabajadores. La aplicación podrá registrar trabajadores (asalariados, autónomos y/o jefes). Cada uno de los anteriores tiene las siguientes características:

    1. Jefes: nombre, apellido, din, acciones, beneficio. Los jefes tendrán la capacidad de despedir un trabajador.

    2. Asalariados: nombre, apellido, dni, sueldo, número de pagas, contratado (booleano)

    3. Autónomo: nombre, apellido, dni, sueldo, contratado (booleano)

    Adicionalmente existirá una clase Empresa que tendrá una lista de trabajadores, donde se podrán registrar todos los trabajadores de la empresa

    La aplicación tendrá la capacidad de:

    1. Registrar un trabajador. Para ello preguntará si se quiere registrar un asalariado, autónomo o jefe y pedirá los datos del trabajador

    2. Listar trabajadores. Para ello preguntará si se quiere listar los asalariados, los autónomos o todos

    3. Mostrar datos de trabajador. Para ello pedirá el DNI y mostrará el siguiente formato:

        1. Si es jefe: Nombre: XXX Apellido: XXX DNI: XXX Acciones: XXX Beneficio: XXX

        2. Si es asalariado: Nombre: XXX Apellido: XXX DNI: XXX Salario Anual: XXX Salario Mensual: XXX Número de Pagas: XX

        3. Si es autónomo Nombre: XXX Apellido: XXX DNI: XXX Salario Anual: XXX

    Realizar un menú para gestionar todos los casos anteriores, teniendo en cuenta las siguientes restricciones:

    - En una empresa no pueden existir dos trabajadores con el mismo DNI
    - En una empresa no puede haber registrado más que un jefe
    - Para poder despedir un trabajador, solo lo puede hacer un jefe
    - Utilizar la herencia y el polimorfismo para juntar el máximo de propiedades y métodos .

7. Desarrollar una aplicación de control de llamadas realizadas en una centralita telefónica.

La centralita mostrará por pantalla todas las llamadas según las vaya registrando. Existen tres tipos de llamadas:

- Las llamadas locales que no tienen coste
- Las llamadas provinciales que cuestan 15 céntimos el segundo.

- Las llamadas nacionales que dependiendo de la franja horaria en la que se realicen cuestan: 20 céntimos en franja 1, 25 céntimos en franja 2 y 30 céntimos en franja 3, cada segundo.
- Todas las llamadas tienen como datos el número origen de la llamada, el número destino y su duración en segundos.

Con la centralita se podrá

- Registrar llamadas, mostrar llamadas realizadas (número origen, número destino, duración y coste)
- Mostrar Costes totales
- Mostrar llamadas realizadas

Para la realización de la práctica se desarrollarán las siguientes clases:

- Llamada (Abstracta): nOrigen, nDestino, coste
- LlamadaProvincial
- LlamadaLocal
- LlamadaNacional: franja
- Centralita: arraylist
- Entrada: main

Decide cuales son los métodos que se deberían de poner en herencia