EJERCICIOS DE CLASES USANDO EXCEPCIONES:

Crear un proyecto con nombre: proyect_coche

1. Diseña la clase Coche, de carácter pública, en el paquete: package_coche.
    - Propiedades privadas: numPuertas, pecio de tipo int. Marca, modelo, color, combustible, cambio de tipo String.
    - Constructores de carácter públicos:
        - Por defecto El número de puertas:3, precio:0, marca:"", modelo:"",color:"BLANCO",combustible:"GASOLINA" y cambio:"MANUAL".
        - Dando los valores de todas las propiedades.
        - Dando otro Coche.
    - **NOTA:**
        - **Es necesario lanzar excepciones de tipo IllegalArgumentException cuando:**
            - El número de puertas no sea 3,4 o 5.
            - El precio es menor de 0.
            - El combustible NO sea "DIESEL" ni "GASOLINA".
            - El cambio NO sea "MANUAL" ni "AUTOMATICO".
        - **Las propiedades de tipo String deben guardarse sin espacios en blanco tanto a la izqda. como a la derecha, y además en mayúsculas.**
    - Métodos:
        - Todos los setter.
        - Todos los getter.
        - toString() devuelve una cadena con los valores del coche actual.
        - clone() que devuelve un coche con las mismas propiedades que el actual.
        - finalize() que muestra un mensaje de destrucción del coche actual.
        - Métodos sobrecargados:
            - equals , dada otro coche, devuelve true o false si son iguales o no lo son.
            - equals , dados los datos correspondientes a las siete propiedades y en el mismo orden, devuelve true o false si son iguales o no lo son con los datos del coche atual.
            - setCoche , dado otro coche, cambia todas las propiedades del coche actual por las propiedades del otro coche dado.
            - setCoche , dados los datos correspondientes a las siete propiedades y en el mismo orden, cambia todas las propiedades del coche actual por esos valores dados.
            -

Diseño de la clase Leer con los métodos: entero y cadena en el paquete: package_coche.

Diseño de la clase principal que contiene el método main en el paquete_ejercicios.

- Declara un coche c1.

- Crea el coche c1 con los valores: 3,15000,”SEAT”,”ALTEA”,”BLANCO”,”DIESEL”,”MANUAL”.
- Muestra las propiedades del coche c1.
- Declara un coche c2.
- Introduce por teclado todas sus propiedades, dentro de un try, y crea c2, mostrando a continuación sus propiedades.
- Muestra fin del programa.

Clase Circulo

1. Diseñar la clase: **Circulo**.

Paquete: package_circulo.
La clase Pedir ha de estar en el package_pedir.
La clase Principal en package_circulo.
Propiedades:

El punto centro **c** y un valor entero: radio, **r**, de carácter privado.

Constructores: (utilizando **this** donde creais conveniente):

Circulo, t.q. inicialice el centro a (-1,-1) y radio 0. Constructor por defecto.
Circulo, t.q. inicialice el centro a (-1,-1) y un valor entero considerado como entrada para el radio.
Circulo, t.q. inicialice el círculo, con dos valores enteros considerados como datos de entrada, para el centro y otro para el radio. ¡¡Cuidado ¡! El radio nunca puede ser negativo, es necesario tratarlo con una excepción.
Circulo, t.q. inicialice el centro del círculo a un punto considerado como entrada, y el radio, a un valor entero tb. Considerado como entrada.
Circulo, t.q. inicialice el círculo a los valores de otro círculo considerado como dato de entrada.

Métodos de acceso a las propiedades:

getCentro, t.q. devuelve el punto centro del círculo actual.
getRadio, t.q. devuelve el radio del círculo actual.
setCentro, t.q. cambia el valor del centro del círculo actual por un punto considerado como entrada.
setCentro, t.q. camabia el valor del centro del círculo actual por dos valores enteros considerados como entrada.
setRadio, t.q. cambia el valor del radio del círculo actual por un valor entero considerado como entrada.// Lanza una excepción IllegalArgumentException.
setCirculo, t.q. dados tres valores enteros, cambie las coordenadas del centro y el radio del círculo actual por ellos.
setCirculo, t.q. dado un círculo, cambie el valor de las propiedades del círculo actual por las respectivas de ese círculo.
setCirculo, t.q. dado un punto, cambie el valor del centro por ese punto, y otro valor entero, considerado como entrada, cambie el radio por ese valor.

Métodos:

toString, t.q. devuelva una cadena con el valor de las propiedades del círculo actual, siendo de carácter público. Atención!, utilizad el método toString de la clase Punto.

equals, t.q dado un círculo considerado como entrada, devuelve true si es igual que el círculo actual, o false si son distintos. Método de carácter público.
equals, t.q. dado un dato de tipo entero y un punto considerados como entrada, devuelve true si son iguales, o false si no lo son. Método de carácter público.
equals, t.q. dados tres valores de tipo entero considerados como datos de entrada, devuelve true si son iguales, o false, si son distintos. Método de carácter público.
area, t.q. devuelva el área del círculo actual.
longitud, t.q. devuelva el perímetro del círculo actual.
ampliaCirculo, t.q. añada una cantidad entera considerada como entera al radio de ese círculo.
distancia, t.q. dando un círculo como entrada, devuelve la distancia existente entre el punto centro del círculo actual y otro, considerado como entrada.
distancia, t.q. dado un punto considerado como entrada, devuelve la distancia existente entre ese punto y el centro del círculo actual.
distancia, t.q., dados dos valores enteros considerados como datos de entrada, devuelve la distancia existente entre esas coordenadas y el centro del círculo actual.

USO DE LA CLASE: CIRCULO

a) Crear los puntos:
Punto **p**, cuyas coordenadas toman el valor por defecto.
Punto **q**, cuyas coordenadas toman los valores: 5 y 4.
Punto **r**, cuyas coordenadas se introducen por teclado.
Punto **t**, cuyas coordenadas toman el valor de las coordenadas de q.

b) Crea los círculos:
Círculo **c1**, cuyas coordenadas del centro son por defecto y el radio es 0.
Circulo **c2**, cuyas coordenadas del centro son las del punto q, y el valor del radio es 4.
Circulo **c3**, cuyas coordenadas del centro son 7 y 10, y el valor del radio es 12.
Circulo **c4**, cuyas coordenadas del centro son por defecto, y el valor del radio es 7.
Circulo **c5**, cuyas coordenadas del centro y el valor del radio coinciden con el del circulo c3.
Circulo **c6**, cuyas coordenadas del centro son las del centro del circulo c3, y el valor del radio es el del radio del circulo c4.
Circulo **c7**, cuyas coordenadas del centro son las del punto opuesto de q, y el valor del radio es 19.
Circulo **c8**, cuyas coordenadas del centro son las del punto intermedio de los puntos p y el centro del circulo c7, y el valor del radio es 24.

c) Mostrar las propiedades de todos los circulos.

d) Mostrar las coordenadas del centro del circulo c8.

e) Se desea cambiar las coordenadas del centro del circulo c5 por las del centro del circulo c7.
Mostrar las propiedades del circulo c5.
Se desea cambiar las coordenadas del centro del circulo c4 por las del punto r.
Mostrar las propiedades de c4.

f) Se desea cambiar las coordenadas del centro del circulo c2 por los valores: 4 y 5 (coordenadas).

Mostrar las propiedades de c2.

g) Se desea cambiar las coordenadas del centro del circulo c4 por las coordenadas de punto suma entre el centro del circulo c4 y el punto opuesto de r.
Mostrar las propiedades de c4.

h) Se desea cambiar el valor del radio del circulo c5 por el valor de la coordenada x del centro del circulo c4.
Mostrar las propiedades de c5.

i) Se desea cambiar las propiedades del circulo c6 por: centro-> centro del circulo c5 , radio-> el valor de la coordenada x del punto r.
Mostrar las propiedades de c6.

j) Se desea cambiar las propiedades del circulo c4 por: coordenadas del centro-> x-> coordenada y del centro del circulo c5, y->7, como radio-> el radio del circulo c4.
Mostrar las propiedades de c4.

k) Se desea cambiar las propiedades del circulo c7 por las del circulo c4.
Mostrar las propiedades de c7.
Calcular la longitud del circulo c8 y mostradla por pantalla.

l) Calcular el área del circulo c1 y mostradla por pantalla.

m) Ampliar el circulo c7 en 10 unidades, Mostar sus propiedades por pantalla.

n) Calcular y mostrar la distancia existente entre el centro del circulo c4 y el del circulo c5.

o) Calcular y mostrar la distancia existente entre el centro del circulo c4 y el del punto t.

p) Calcular y mostrar la distancia existente entre el centro del circulo c3 y el del centro del circulo c1.

q) Calcular y mostrar la distancia existente entre el centro del circulo c7 y el punto formado por las coordenadas del punto generado por las coordenadas, x-> la coordenada x del centro del circulo c2 y la coordenada y del centro del circulo c3.

Clase Nif

2. Diseñar la clase **Nif**:
   Proyecto: Proyecto_Personas.
   Paquete: package_Nif.
   Clase Pedir en package_fecha.
   Clase Principal en package_Nif.

- **Propiedades:**
  - De tipo String **nif,** de carácter privado. Donde guarda el dni de una persona junto con su letra de NIF.
  - De tipo cadena **letras**, de carácter privado, cuyo contenido es común para todos los objetos de la clase NIF y es: “TRWAGMYFPDXBNJZSQVHLCKE”. (Cadena donde se almacenan las 23 posibles letras de NIF).

  Tanto en los constructores como métodos, tenéis que comprobar si el dni correcto, porque la cadena introducida no posee una longitud de 9 letras(una vez eliminados los espacios en blanco y pasado a mayúsculas).
  Enviad una excepción en tal caso, del tipo **IllegalArgumentException**.
  Los 8 primeros caracteres representan un número entero (excepción).
  ¿Cómo calcular la letra? El resto de dividir el dni por 23 nos indica la posición que ocupa la letra correcta en el vector de letras.

- **Constructores:**

  Nif tal que inicialice con una cadena considerada como dato de entrada.
  - Quitad los espacios en blanco a esa cadena y pasadla a mayúsculas.
  - Si la longitud de la cadena no es 9, lanzad una excepción.
  - Si los 8 primeros dígitos no representan un número entero, lanzad una excepción.
  - Extraer la última letra y comprobar si coincide con el resto de dividir el dni (8 dígitos) entre 23 de la cadena **letras de la clase Nif**.

  Nif tal que dado un número entero dni, calcule la letra que le corresponde y lo guarde como nif.
  Nif tal que inicialice el nif con el de otro objeto de tipo Nif.

- **Métodos de acceso a la s propiedades:**

  GetNif tal que devuelva el valor del Nif actual.
  SetNif tal que cambia el valor del Nif actual por un valor String considerado como entrada, controlad si es correcto.
  SetNif tal que dado un dni, calcula la letra y la guarda como nuevo nif del objeto actual.

- **Métodos:**

  toString tal que devuelva una cadena con el valor del Nif actual, siendo de carácter público.
  equals tal que devuelve true o false, si el dni del Nif actual coinide con uno considerado como dato de entrada. Método de carácter público.

Uso de la clase Nif:

a) Introduce el nif de una persona por teclado.
b) Mostrar ese mismo dni.
c) Crea un Nif con valor: 23027712C.
d) Mostrar ese mismo Nif.
e) Crea un Nif con valor: 23027712.
f) Comprobar si los Nif c) y e) son iguales.

Clase Nie

3. Diseñar la clase **Nie**:
   Proyecto: Proyecto_Personas.
   Paquete: package_Nif.
   Clase Pedir en package_fecha.
   Clase Principal en package_Nif.

- **Propiedades:**

  - De tipo String **nie,** de carácter privado. Donde guarda el dni de una persona junto con su letra de NIE.
  - De tipo cadena **letras**, de carácter privado, cuyo contenido es común para todos los objetos de la clase NIE y es: “TRWAGMYFPDXBNJZSQVHLCKE”. (Cadena donde se almacenan las 23 posibles letras de NIE).

Tanto en los constructores como métodos, tenéis que comprobar si el dni correcto, porque la cadena introducida no posee una longitud de 9 letras(una vez eliminados los espacios en blanco y pasado a mayúsculas).
Enviad una excepción en tal caso, del tipo **IllegalArgumentException**.
La primera letra ha de ser 'X','Y' o 'Z'. Si es 'X' entonces se sustituye por el dígito '0', si es 'Y' por el '1' y si es 'Z' por el '2'.
A partir de la 2ª letra, los otros 7 restantes caracteres representan un número entero (excepción).
¿Cómo calcular la letra? El resto de dividir el dni por 23 nos indica la posición que ocupa la letra correcta en el vector de letras.

- **Constructores:**

  Nie tal que inicialice con una cadena considerada como dato de entrada.
  - Quitad los espacios en blanco a esa cadena y pasadla a mayúsculas.
  - Si la longitud de la cadena no es 9, lanzad una excepción.
  - La primera letra ha de ser 'X','Y' o 'Z'.
  - Si los 7 caracteres restantes no representan un número entero, lanzad una excepción.
  - Extraer la última letra y comprobar si coincide con el resto de dividir el dni (8 dígitos) entre 23 de la cadena **letras de la clase Nie**.

  Nie tal que inicialice el nie con el de otro objeto de tipo Nie.

- **Métodos de acceso a la s propiedades:**

  GetNie tal que devuelva el valor del Nie actual.
  SetNie tal que cambia el valor del Nie actual por un valor String considerado como entrada, controlad si es correcto.

- **Métodos:**

  toString tal que devuelva una cadena con el valor del Nie actual, siendo de carácter público.

Uso de la clase Nie:

g) Introduce el nif de una persona por teclado.
h) Mostrar ese mismo dni.
i) Crea un Nif con valor: X3027712C.
j) Mostrar ese mismo Nif.
k) Crea un Nif con valor: X3027713F

Clase Fecha

4. Diseñar la clase: **Fecha**.

Proyecto: Proyecto_Personas.
Paquete: package_fecha.
Clase Pedir en package_fecha.
Clase Principal en package_fecha.

- ✓ **Propiedades:**

  De tipo entero **dia**, **mes** y **anio** y carácter privado.
  Un vector de 12 elementos de tipo entero, llamado **meses**. Ese vector será una sola copia para todos los objetos de tipo Fecha y contiene los días de cada mes del año. Un vector de 12 elementos de tipo String, llamado **nombres**. Ese vector será una sola copia para todos los objetos de tipo Fecha y contiene los nombres de los meses del año. Sus valores nunca cambiarán.

Tanto en los constructores como métodos, tenéis que comprobar si la fecha es correcta, utilizando el método: **fechaCorrecta**.// El año se considera correcto a partir del 2000 (siglo XXI).
Si la fecha es incorrecta, entonces lanzad una excepción del tipo: **IllegalArgumentException**.

- ✓ **Constructores:** (utilizando **this** donde creais conveniente):

  Fecha, t.q. inicialice la fecha al 1-1-2000. Constructor por defecto.
  Fecha, t.q. inicialice la fecha, con tres valores enteros considerados como datos de entrada. Es necesario tratarlo con una excepción.
  Fecha, t.q. inicialice la fecha a los valores de otra fecha considerada como dato de entrada.

- ✓ **Métodos de acceso a las propiedades:**

  getDia, t.q. devuelve el valor del día de la fecha actual.
  getMes, t.q. devuelve el valor del mes de la fecha actual.
  getAño, t.q. devuelve el valor del año de la fecha actual.
  setDia, t.q. cambia el valor del dia de la fecha actual por un valor entero considerado como entrada.
  setMes, t.q. cambia el valor del mes de la fecha actual por un valor entero considerado como entrada.
  setAño, t.q. cambia el valor año de la fecha actual por un valor entero considerado como entrada.
  setFecha, t.q. dados tres valores enteros, cambie las propiedades de la fecha actual por ellos.
  setFecha, t.q. dada una fecha, cambie el valor de las propiedades de esa fecha por la fecha actual.

- ✓ **Métodos:**

  toString, t.q. devuelva una cadena con el valor de las propiedades de la fecha actual, siendo de carácter público. El formato con el que se devuelva será dependiendo del valor de un valor entero considerado como entrada:
  Si ese valor es 1, formato: dd-mm-yy
  Si es 2, formato: dd-mm-yyyy
  Si es 3, formato: dd-Nombre mes-yyyy
  Si es 4, formato: dd de NombeMes de yyyy
  Si es 5, formato: dd/mm/yy
  Si es 6, formato: dd/mm/yyyy
  equals, t.q. dada una fecha considerada como dato de entrada, devuelve true si es igual a la actual, o fals, si es distinta. Método de carácter público.
  equals, t.q. dados tres valores considerados como datos de entrada de tipo entero, devuelve true si coinciden con la fecha actual, o false, si no lo hacen. Método de carácter público.
  clone, t.q. devuelva una fecha con los mismos valores que la fecha actual.
  finalize, tq muestre un mensaje de destrucción de la fecha actual.
  fechaCorrecta, t.q. devuelve true si la fecha actual es correcta o false si no la es. De carácter privado. Aplicad este método en aquellos en los que sean necesarios.
  bisiesto, t.q. dado un año considerado como dato de entrada, devuelve true si es bisiesto o false si no lo es. El método no trabaja con la fecha actual, con lo q se puede considerar como independiente.
  toDias, t.q. devuelve el número total de días pasado desde la fecha el 1 de Enero de 2000 hasta la fecha actual.

fechaMayor, t.q. devuelva cuál es la fecha mayor de la actual y una considerada como entrada.
fechaMenor, t.q. devuelva cuál es la fecha menor de la actual y una considerada como entrada.
diasEntreFechas, t.q. devuelva los días comprendidos entre una fecha considerada como entrada y la actual.
fechaSiguiente, t.q. incremente en un dia la fecha actual.
fechaAnterior, t.q. decremente en un día la fecha actual.

USO DE LA CLASE: FECHA

***** Es necesario capturar las posibles excepciones que puedan ocurrir.

a) Cread las fechas:
Fecha **f1**, cuyas propiedades son por defecto.
Fecha **f2**, cuyos valores son: 20, 2,1. Si la fecha no es correcta, f2 tomará los valores por defecto 01-01-2000.
Fecha **f3**, cuyos valores son introducidos por teclado. Obligar a que la fecha f3 sea correcta.
Fecha **f4**, cuyo valor coincide con la de la fecha f2.

b) Fecha **f5**, cuyo valor del día y el año,se introduce por teclado, y el valor del mes es 12. Si la fecha f5 no es correcta, f5 tomará los valores por defecto.

c) Fecha **f6**, cuyo día coincide con el día de la fecha f4, mes es el mes de la fecha f5 y año 2000. Controlar si se produce alguna excepción del tipo fecha incorrecta, si esto es así, f6 tomará los valores por defecto.

d) Mostrad por pantalla los valores de las fechas:
f1 con el formato de salida dd-mm-yyyy
f2 con el formato de salida dd-NombreMes-yyyy
f3 con el formato de salida dd de NombreMes de yyyy
f4 con el formato de salida dd/mm/yyyy
f5 con el formato de salida dd-mm-yyyy
f6 con el formato de salida dd/mm/yyyy.

e) Cambiar la fecha f4 por los valores de la fecha f5.

f) Cambiar la fecha f6 por el día de f1, mes de f 1 y año de f3.

g) Mostrad el número de días transcurridos desde 1 de Enero del 2000 hasta la fecha f5.

h) Mostrad cuál es la fecha siguiente de la fecha f3.

i) Mostrad los días transcurridos entre las fechas f5 y f4.

j) Mostrad cuál es la fecha más antigua generada de comparar la fecha f5 y f2.

k) Mostrad cuál es la fecha más nueva generada de comparar la fecha f1 y f5.

l) Mostrad cuál es la fecha anterior a la fecha f4.

m) Cread una fecha **f7** será la fecha resultante de la fecha mayor entre la fecha siguiente de f4 y la anterior a la f3.

Clase Persona

5. Diseño de la clase: **Persona**
Proyecto: Proyecto_Personas.
Paquete: package_persona.
Clase Pedir en package_fecha.

Clase Principal en package_personas.

➔ **Propiedades de carácter privado:**

numPersonas de tipo int, y toma como valor inicial 0, lleva el recuento de personas en total (objetos que crean de tipo Persona) y es común para todos las personas de la clase Persona.
nombre, apellido1, apellido2 de tipo cadena.
sexo de tipo char, y sus valores serán 'V' o 'M'
estado de tipo String y sus valores serán "SOLTERO","CASADO","VIUDO" o "DIVORCIADO"
fechaNacimiento de tipo: Fecha.
nif de tipo: Nif
nie de tipo: Nie

Paquetes:
- La clase Persona está definida en el paquete package_persona
- La clase Fecha, está definida en el paquete package_fecha, por lo que la clase, constructores y métodos con el tipo de acceso amistoso deben de declararse como públicos.
- La clase Nif, está definido en el paquete package_Nif, por lo que la clase, constructores y métodos con el tipo de acceso amistoso deben de declararse como públicos.

➔ **Constructores**:

Excepciones:
- Fecha incorrecta
- Dni incorrecto
- Nombre, apellido1 y apellido2 con números

- Por defecto, las propiedades de tipo cadena: nombre, apellido1 y apellido2 en blanco, fechaNacimiento utiliza el constructor por defecto y dni toma el valor: 0. El sexo='V' y estado="SOLTERO". Las propiedades id y numPersonas se incrementan a 1.
- Con 7 datos de entrada en el mismo orden que en la declaración de las propiedades y del mismo tipo (para fechaNacimiento, una fecha, para nif un Nif). Las propiedades id y numPersonas se incrementan a 1.
- Con 7 datos de entrada en el mismo orden que en la declaración de las propiedades y del mismo tipo (para fechaNacimiento, una fecha, para dni (int)). Las propiedades id y numPersonas se incrementan a 1.
- Con 7 datos de entrada en el mismo orden que en la declaración de las propiedades y del mismo tipo (para fechaNacimiento, una fecha, para nie un Nie). Las propiedades id y numPersonas se incrementan a 1.
- Con 7 datos de entrada en el mismo orden que en la declaración de las propiedades y del mismo tipo (para fechaNacimiento, una fecha, para nie o nif el valor de un Objeto de tipo Object). Las propiedades id y numPersonas se incrementan a 1.
- Con un dato de entrada que sea otro objeto de tipo Persona. Las propiedades id y numPersonas se incrementan a 1.

➔ **Métodos de acceso:**

- Para la obtención de cada una de las propiedades.
- Para cambiar el valor a las propiedades de forma individual. //Con excepciones

- **setPersona**, t.q. dadas 7 valores de entrada en el mismo orden que en la declaración de las propiedades y del mismo tipo, cambie todos los valores de todas las propiedades, un objeto de tipo Nif como dato de entrada.
- **setPersona**, t.q. dadas 7 valores de entrada en el mismo orden que en la declaración de las propiedades y de tipo cadena, cambie todos los valores de todas las propiedades, un objeto de tipo Nie como dato de entrada.
- **setPersona**, t.q. dadas 7 valores de entrada en el mismo orden que en la declaración de las propiedades y de tipo cadena, cambie todos los valores de todas las propiedades, un objeto de tipo Object como dato de entrada.
- **setPersona**, t.q. dada otro objeto de tipo Persona, cambie los valores de la persona actual por los de la ese objeto.
- **toString,** t.q. devuelve una cadena con los valores de las propiedades de la persona actual.
- **finalize**, t.q. muestre un mensaje con la destrucción de la persona actual. Se decrementa el contador numPersonas.
- **totalPersonas**, t.q. devuelve el valor de numPersonas.

USO DE LA CLASE PERSONA.

- Crear la persona ag007 con los valores: “Bond”, “James”, “Bond”, “Hombre”,“Soltero”,fecha: 10-2-2001, dni=1234567.
- Crear la persona p con los datos introducidos por teclado, detectando los posibles errores.
- Mostrad la persona p, con sus posibles formato de salida en el nombre y cuya fecha de nacimiento tenga como formato: dd Nombre de Mes yy.
- Cambia la fecha de nacimiento de ag007 al día siguiente.