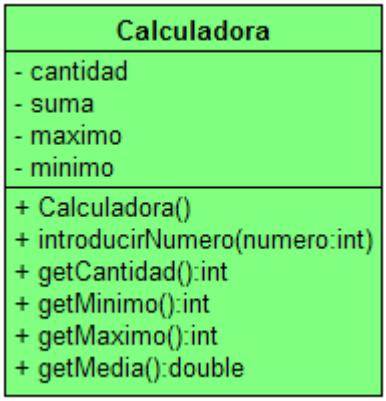
Ejercicios adicionales UT3 (III) - *Estructura condicional*

Ejercicio 11.

Crea en C:\Programacion\UT3 un nuevo proyecto Calculadora y añade una clase con el mismo nombre. Defínela tal como muestra el diagrama UML. La calculadora registra uno a uno una serie de n^{os} y va calculando al mismo tiempo el valor máximo y mínimo de entre los introducidos y la suma total de todos ellos.

La clase define los siguientes atributos:



- *cantidad* de tipo entero – aquí se van contando cada uno de los n^{os} que se introducen
- *suma* de tipo entero – aquí se van acumulando los n^{os} introducidos
- *maximo* de tipo *int* – registra el valor máximo de entre todos los introducidos hasta el momento
- *minimo* de tipo *int* – es el mínimo de los valores introducidos hasta el momento

Los métodos son:

- el constructor sin parámetros que inicializa todos los atributos a 0
- accesores para la cantidad, mínimo y máximo de la calculadora
- introducirNumero(int numero) – añade un nuevo número a un objeto Calculadora. El efecto será que el estado de la calculadora cambia puesto que habrá que contar el número, añadirlo a la suma y actualizar el valor del máximo y del mínimo
- double getMedia() – devuelve la media de todos los valores introducidos a la calculadora

Prueba la clase creando un objeto *miCalculadora*, añade varios números seguidos y comprueba cuál es la media, el máximo y mínimo.

Ejercicio 12.

Crea en C:\Programacion\UT3 un nuevo proyecto Hora. Añade una clase Hora que almacenará en tres atributos enteros la hora, los minutos y los segundos (en formato 24h.). Suponemos todos los valores correctos.

Incluye en la clase los siguientes métodos:

- el constructor con tres parámetros
- accesores y mutadores para la hora, los minutos y los segundos
- aSegundos() devuelve la hora que guarda el objeto Hora en segundos
- el método avanzarSegundo() que incrementa un segundo la hora
- el método atrasarSegundo() que decrementa un segundo la hora
- el método toString() que devuelve una cadena representando la hora en formato "**hh:mm:ss**". Por ejemplo, "19:07:30" o bien "04:05:03"

Ejercicio 13.

Crea en C:\Programacion\UT3 un nuevo proyecto Fecha. Añade una clase Fecha con tres atributos enteros para el día, mes y año.

La clase incluye los siguientes métodos:

- el constructor con parámetros que inicializa los objetos con los valores que aquellos proporcionan
- no incluiremos en este ejercicio accesores ni mutadores (por comodidad)

- un método esBisiesto() que devuelve *true* si el año es bisiesto, *false* en otro caso. Un año es bisiesto si es múltiplo de 4 pero no de 100 aunque si de 400. (No utilices en este método la sentencia *if*)
- el método diasMes() que devuelve la cantidad de días que tiene el mes (influirá si año es bisiesto en el caso de febrero). Utiliza la sentencia *switch*.
- un método printCorta() que escribe la fecha en formato corto: "3 – 11 – 2005"
- un método printLarga() que escribe la fecha en formato largo: "3 de Noviembre de 2005". Utiliza la sentencia *switch*.
- Añade un método esCorrecta() que devuelva *true* si la fecha que contiene el objeto es correcta. Una fecha se considera correcta si:
  - el día está comprendido entre 1 y 31
  - el mes está entre 1 y 12
  - si el mes no es febrero el día estará bien en relación al nº de mes, es decir, entre 1 y 31 o entre 1 y 30
  - si el mes es febrero el día estará entre 1 y 28 o entre 1 y 29 si el año es bisiesto

Ejercicio 14.

Añade a la clase Fecha del ejercicio anterior los siguientes métodos:

- public boolean precedeA(int queDia, int queMes, int queAño) - devuelve *true* si la fecha que guarda el objeto actual precede a la fecha que representan los tres parámetros que se pasan al método. Suponemos que la fecha representada por los tres parámetros es correcta.

- public void avanzarDia() - avanza un día la fecha.

Fecha actual	Nueva fecha
2 – 11 - 2007	**3 – 11 -2007**
30 – 11 - 2007	**1 – 12 - 2007**
29 – 2 - 2008	**1 – 3 - 2008**
31 – 1 - 1962	**1 – 2 - 1962**
31 – 12 - 2000	**1 – 1 - 2001**

Ejercicio 15.

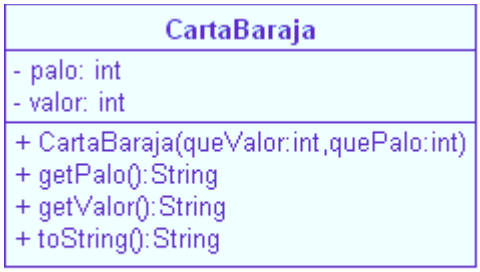
Crea en C:\Programacion\UT3 un nuevo proyecto TresNumeros. Añade una clase TresNumeros. Los objetos de esta clase guardarán tres números enteros en los atributos *numero1*, *numero2* y *numero3*.

Añade a la clase:

- un constructor con parámetros que inicialice los objetos
- no incluiremos accesores ni mutadores por comodidad
- un método getMaximo() que devuelve el valor máximo de los tres números. Haz una versión sin utilizar el operador lógico && y usando variables auxiliares.
- un método getMinimimo() que devuelve el mínimo valor de los tres.
- un método sonIguales() que devuelve true si los tres nºs son iguales, false en otro caso
- un método cuantosPares() que devuelve cuántos de los tres números son pares
- un método cuantosAcabanEn7() que devuelve cuántos de los tres números terminan con la cifra 7
- un método ordenar() que modifica los atributos dejándolos en orden ascendente. Haz una versión sin utilizar el operador lógico && y usando variables auxiliares.

Ejercicio 16.

Crea en C:\Programacion\UT3 un nuevo proyecto CartaBaraja y añade la clase CartaBaraja. Esta clase modela una carta de la baraja española.



Cada carta tiene un palo y un valor, ambos atributos de tipo entero. Define dentro de la clase cuatro constantes: OROS, COPAS, ESPADAS, BASTOS y asócialas con los valores 1, 2, 3 y 4.

La clase incluye constructor con parámetros y accesores para el palo y el valor. Los accesores devuelven un String: "OROS", "SOTA", "CABALLO", ... (utiliza la sentencia *switch* en getPalo() y la sentencia *if* en getValor()).

No hay mutadores.

El método toString() devuelve una cadena conteniendo una representación del objeto de la forma: " *sota de espadas*", "*tres de oros*", "*cuatro de copas*", ...

Ejercicio 17.

Añade a la clase Hora del ejercicio 12 el método enFormatoAmPm() que devuelve la hora guardada como un String pero en formato de 12h, AM / PM.

Por ejemplo:

22:12:07 se devuelve como 10:12:07 PM
06:06:13 se devuelve como 06:06:13 AM
12:12:07 se devuelve como 12:12:07 PM
00:12:07 se devuelve como 12:12:07 AM