Introducción:

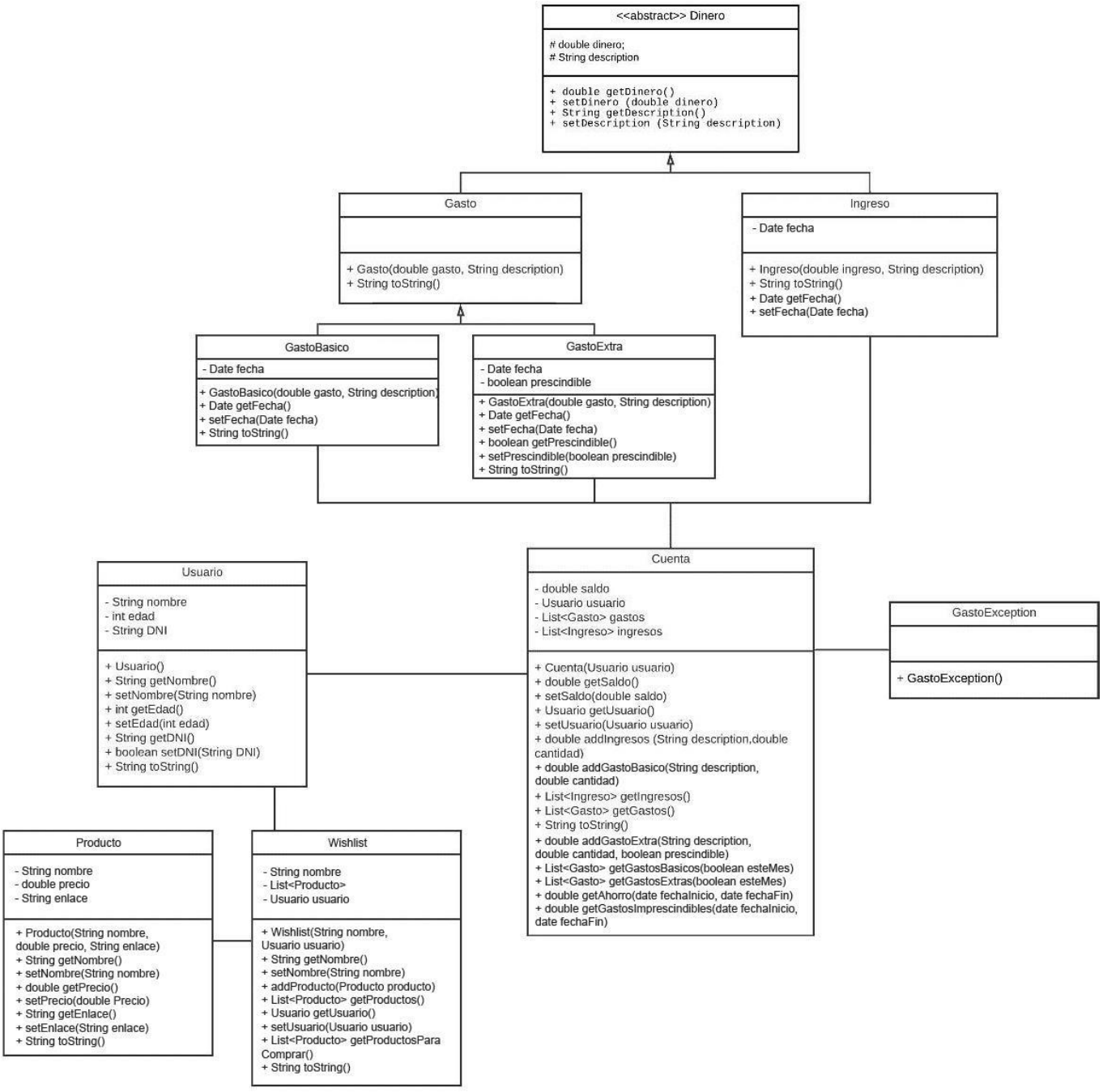
Tras un tiempo usando nuestra aplicación, nos damos cuenta de que necesitamos mejorarla y ampliar su funcionalidad. Para ello vamos a implementar dos tipos de gastos distintos para poder consultarlos y ver en qué podemos mejorar en nuestra economía.

Además, añadiremos una lista de deseos de productos que queremos comprar conforme tengamos dinero ahorrado.

Por último, ampliaremos la clase Cuenta con nuevos métodos para ayudarnos a ahorrar en nuestro día a día.

Para realizar la práctica se reutilizará el programa que ya se ha hecho, pero para acostumbrarnos a leer código de otros y trabajar sobre él, se intercambiará el programa con los compañeros que también hagan esta práctica.

DIAGRAMA DE CLASES



Clases GastoBasico y GastoExtra

Ambas heredan de **Gasto**, pero añaden un nuevo atributo llamado *fecha*, que se utilizará para guardar la fecha en que se anota el gasto.

GastoExtra:

Tendrá otro atributo booleano llamado *prescindible,* que marcaremos a verdadero si creemos que este gasto nos lo podríamos ahorrar para el mes siguiente. Por ejemplo, ir al gimnasio no sería un gasto básico, pero podemos considerarlo como que no es prescindible, porque queremos seguir yendo los siguientes meses.

Clase Ingreso

Añadiremos un nuevo atributo *fecha,* que se utilizará para guardar la fecha en que se anota el ingreso. Tened en cuenta este cambio para el resto del código.

Clase Cuenta

Al crear un gasto en **Cuenta**, ahora será de uno de los 2 tipos, pero se añadirá al mismo listado de gastos que ya tenemos aquí.

El método addGastos(), lo renombraremos a addGastoBasico()

Crearemos uno nuevo para gastos extras: addGastoExtra()

-> Tened en cuenta que ambos métodos deberán guardar la fecha de ese instante en el que se crea el apunte del gasto.

Lo mismo para addIngresos().

Se crearán estos dos nuevos métodos:

getGastosBasicos(boolean esteMes)
getGastosExtras(boolean esteMes)

Estos métodos devolverán un listado mostrando el gasto, descripción, fecha de cada uno.

Los métodos recibirán un parámetro booleano, si es verdadero, se mostrarán solo los gastos del mes en el que estamos.

Por último, añadiremos nuevas funcionalidades a **Cuenta** para gestionar nuestro ahorro, crearemos los siguientes métodos:

getAhorro(): Calculará el ahorro del período indicado.

getGastosImprescindibles() - del período indicado, cogerá los gastos básicos, y los gastos extras que no son considerados prescindibles, y devolverá el valor total.

Clase Producto

Una clase sencilla, con 3 atributos y su constructor.

Clase Wishlist

Será una lista de deseos, compuesta por productos que queremos comprar.

Además de los métodos del diagrama tendremos el siguiente:

getProductosParaComprar() – La Wishlist pertenece a un Usuario, así que consultaremos el saldo de su cuenta para hacer el cálculo. Este método devolverá los productos que podríamos comprar, con la condición de que nos quede dinero para cubrir los gastos básicos del siguiente mes. (Los productos se comprarían de forma individual, no hay que hacer una suma de todos esos productos)

Main:

El menú lo modificaremos para adaptarse a las nuevas funcionalidades, quedando de la siguiente manera:

```
Realiza una nueva acción
1 Introduce un nuevo gasto básico
2 Introduce un nuevo gasto extra
3 Introduce un nuevo ingreso
4 Mostrar gastos
5 Mostrar ingresos
6 Mostrar saldo
7 Mostrar ahorro de un período
8 Mostrar gastos imprescindibles
9 Mostrar posibles ahorros del mes pasado
10 Mostrar lista de deseos
11 Mostrar productos que podemos comprar
0  Salir
```

`- Mostrar gastos:` Una vez seleccionada esta opción, pedirá si quieres ver los gastos totales, los gastos básicos o los gastos extras.

`-Mostrar posibles ahorros del mes pasado:` Calculará de forma automática los gastos extras marcados como prescindibles del mes anterior.