Explicación de clases

Explicación de las clases tipo Fecha y Ubicacion

```
#ifndef FECHA_H
#define FECHA_H
```

Nombre de la clase acabandp en _H para poder identificarlo como un .h más rapidamente, ifdef y define para evitar inclusiones múltiples

```
#include <iostream>
using namespace std;
class Fecha {
private:
    int dia;
    int mes;
```

Estableciendo los atributos (que como la mayoría) son privados para que tengan que ser editados con punteros y asi evitar que se cambien valores de día o mes sin querer.

```
public:
    Fecha();
    Fecha(int dia, int mes);
```

Método creador

```


    int getDia() const;
    int getMes() const;
    void setDia(int dia);
    void setMes(int mes);
};
#endif
```

Definición de los métodos de la clase (que como el resto) son públicos para que pueda accesar desde culaquier lado, y gracias a esto puedo obtener y modificar el dia o mes de un objeto. El código .h de ubicación es igual pero con atributos de nombres diferentes.

Explicación de la clase central : Bodega

```
#ifndef BODEGA_H
#define BODEGA_H
#include <iostream>
using namespace std;
#include "Fecha.h"
#include "Ubicacion.h"
```

Incluyendo Fecha y Ubicación pues Bodega tiene una relción de composición con ambos.

```
class Bodega {
protected:
    Ubicacion ubicacion;
    Fecha ultimoUso;
    string tamaño;
    string dureza;
    string nombre;
    bool descompuesto;
    int cantidad;
```

Definiendo los atributos que seguirán los objetos de las clases derivadas de Bodega, y son protegidos para que las clases derivadas puedan acceder sin tanta complicación a ellos.

```
public:
    Bodega();
    Bodega(const Ubicacion& ubic, const Fecha& fecha,
           const string& tamaño, const string& dureza,
           bool descompuesto, int cantidad,string nombre);
    Ubicacion getUbicacion() const;
    Fecha getUltimoUso() const;
    string getNombre() const;
    string getTamaño() const;
    string getDureza() const;
    bool getDescompuesto() const;
    int getCantidad() const;
    void setUbicacion(const Ubicacion& ubic);
    void setUltimoUso(const Fecha& fecha);
    void setCantidad(int cantidad);
```

Método constructor que establece que varibales no serán modificadas junto con métodos get/set para poder acceder a los atributos

```
    virtual void VisualizarInf() const = 0;
    virtual void MenuSecundario()const = 0;
    virtual void CrearObjetos() = 0;
};
#endif
```

Definición de las funciones virtuales puras que luego serán instanciadas a tras de la sobrescritura.

Explicación de archivo.h de las 3 sublcases de Bodega

```
#ifndef ALMACENAMIENTO_H
#define ALMACENAMIENTO_H
```

Nombre de la clase (H) porque hereda y ifndef y define

```
#include <iostream>
using namespace std;

#include "Bodega.h"
class Almacenamiento : public Bodega {
```

Creando la clase y su relación de herencia con Bodega

```
private:
    string tipo;
    bool digital;
    string contenido;
```

Creación de Atributos

```
public:
    Almacenamiento();
    Almacenamiento(const Ubicacion& ubic, const Fecha&fecha, const string&
const string& tipo, bool digital,const string& contenido);
```

Método constructor con parametros predeterminados y estableciedno cuales van a poder ser cambiados por el usuario

```
    void VisualizarInf() const;
    void MenuSecundario()const;
    void CrearObjetos();
    void BuscarObjeto(const vector<Almacenamiento>& lista) const;
    void LeerLista(const vector<Almacenamiento>& lista)const;
    void ModificarObjeto(vector<Almacenamiento>&lista);
};
#endif
```

Definición de las funciones puras y definicón de funciones que usaran el vector, esto se repite casi igual (solo cambian unos atributos) en Decoraciones y Almacenamiento.