Volem implementar un conjunt de classes que ens permetin gestionar els lliuraments que els estudiants fan dels exercicis d’una assignatura en una plataforma similar a Caronte utilitzant l’eina de lliurament d’exercicis avaluables.

Per poder guardar la informació de cadascun dels exercicis que s’han de lliurar us donem ja creada una classe Exercici que guarda la descripció i la data límit de lliurament d’un exercici, un array dinàmic amb tots els lliuraments dels estudiants i el nº total d’estudiants que poden lliurar l’exercici:

class Exercici

{

public:

   Exercici();

   Exercici(const string& descripcio, const string& dataLimit);

   Exercici(const Exercici& e);

   ~Exercici();

   Exercici& operator=(const Exercici& e);

   void inicialitzaEstudiants(const string& fitxerEstudiants);

   bool afegeixTramesa(const string& niu, const string& fitxer,

         const string& data);

   bool consultaTramesa(const string& niu, const string& data,

         string& fitxer);

   bool eliminaTramesa(const string& niu, const string& data);

private:

   string m\_descripcio;

   string m\_dataLimit;

   LliuramentsEstudiant\* m\_lliuraments;

   int m\_nEstudiants;

};

A l’array dinàmic m\_lliuraments tindrem un element per cadascun dels estudiants que poden lliurar l’exercici. A cada element de l’array hi guardarem un objecte de la classe LliuramentEstudiant que guarda la informació de l’estudiant (niu i nota) i una llista amb totes les trameses que ha fet l’estudiant (un estudiant pot fer moltes trameses d’un mateix exercici, tantes com vulgui):

class LliuramentsEstudiant

{

public:

   LliuramentsEstudiant() : m\_niu(""), m\_nota(0.0) {};

   void setNiu(const string& niu) { m\_niu = niu; }

   string getNiu() const { return m\_niu; }

   float getNota() const { return m\_nota; }

   void afegeixTramesa(const string& fitxer, const string& data);

   bool consultaTramesa(const string& data, string& fitxer);

   bool eliminaTramesa(const string& data);

private:

   string m\_niu;

   float m\_nota;

   forward\_list<Tramesa> m\_trameses;

};

Cada tramesa que fa l’estudiant consta d’un fitxer amb la solució de l’exercici i una data de lliurament:

class Tramesa

{

public:

   Tramesa();

   Tramesa(const string& fitxer, const string& data) : m\_fitxer(fitxer), m\_data(data) {}

   void setFitxer(const string& fitxer) { m\_fitxer = fitxer; }

   void setData(const string& data) { m\_data = data; }

   string getFitxer() const { return m\_fitxer; }

   string getData() const { return m\_data; }

private:

   string m\_fitxer;

   string m\_data;

};

A partir d’aquestes definicions us demanem el següent:

**1.       Implementeu els mètodes següents de  la classe LliuramentEstudiant:**

-          **Implementeu el** **mètode afegeixTramesa** per afegir les dades d’una tramesa (nom del fitxer i data) a la llista de trameses de l’estudiant. El mètode tindrà aquesta capçalera:

void afegeixTramesa(const string& fitxer, const string& data);

-           **Implementeu el mètode eliminaTramesa** per eliminar una tramesa que hagi fet l’estudiant. El mètode tindrà aquesta capçalera:

bool eliminaTramesa(const string& data);

S’ha d’eliminar la tramesa amb la data especificada com a paràmetre. Podeu suposar que un estudiant no farà dues trameses amb la mateixa data. El mètode retorna true si la tramesa existeix i es pot eliminar i false, en cas contrari.

-          **Implementeu el mètode consultaTramesa** per consultar una tramesa que hagi fet l’estudiant. El mètode tindrà aquesta capçalera:

bool consultaTramesa(const string& data , string& fitxer);

S’ha de retornar al paràmetre per referència fitxer, el nom del fitxer que correspon a la tramesa amb la data especificada com a paràmetre. Podeu suposar que un estudiant no farà dues trameses amb la mateixa data. El mètode retorna true si la tramesa existeix i false, en cas contrari.

**2.       Implementeu els mètodes següents de la classe Exercici:**

-          **Implementeu el mètode inicialitzaEstudiants** per inicialitzar les dades dels estudiants que poden lliurar l’exercici a partir d’un fitxer de text. El mètode tindrà aquesta capçalera:

void inicialitzaEstudiants(const string& fitxerEstudiants);

La primera línia del fitxer tindrà el nº total d’estudiants i a continuació hi haurà una línia per estudiant que contindrà només el niu de cadascun dels estudiants. S’haurà d’inicialitzar l’array dinàmic m\_lliuraments amb el nº d’estudiants que indiqui la primera línia del fitxer i afegir a l’array un objecte del tipus LliuramentsEstudiant per cadascun dels estudiants del fitxer.

-          **Implementeu el mètode afegeixTramesa** per afegir les dades d’una nova tramesa d’un estudiant. El mètode tindrà aquesta capçalera:

bool afegeixTramesa(const string& niu, const string& fitxer,

      const string& data);

S’ha de buscar l’estudiant amb el niu especificat dins de l’array m\_lliuraments i, si existeix, afegir-hi una nova tramesa amb les dades (fitxer i data) que es passen com a paràmetre. Si l’estudiant existeix i es pot afegir la tramesa retorna true, i si no, retorna false.

Si és la primera tramesa que fa l’estudiant, s’haurà d’afegir un nou objecte de tipus LliuramentEstudiant al conjunt de lliuraments i afegir la tramesa corresponent als lliuraments d’aquest estudiant utilitzant el mètode afegeixTramesa. Si l’estudiant ja havia fet prèviament alguna tramesa, simplement s’haurà de recuperar l’objecte LliuramentEstudiant que correspon a aquest estudiant i afegir-hi la tramesa corresponent.

-          **Implementeu el mètode consultaTramesa** per consultar les dades d’una tramesa d’un estudiant. El mètode tindrà aquesta capçalera:

bool consultaTramesa(const string& niu, const string& data,

string& fitxer);

S’ha de buscar l’estudiant amb el niu especificat dins de l’array m\_lliuraments i, si existeix, recuperar les dades (nom del fitxer) de la tramesa amb la data que es passa com a paràmetre. Si l’estudiant existeix i es pot recuperar la tramesa retorna true, i si no, retorna false.

-          **Implementeu el mètode eliminaTramesa** per eliminar les dades d’una tramesa d’un estudiant. El mètode tindrà aquesta capçalera:

bool eliminaTramesa(const string& niu, const string& data);

S’ha de buscar l’estudiant amb el niu especificat dins de l’array m\_lliuraments i, si existeix, eliminar les dades de la tramesa amb la data que es passa com a paràmetre. Si l’estudiant existeix i es pot eliminar la tramesa retorna true, i si no, retorna false.

-          **Implementeu** el **constructor de còpia**, el **destructor** i l’**operador d’assignació**.