//#include <iostream>

//using namespace std;

//

//class Exception

//{

//protected:

// string text;

// string source; // file

// int line;

// string date;

//public:

// Exception(string text,string source, int line, string date)

// :text(text),source(source),line(line),date(date)

// {

//

// }

//

// virtual void PrintMessage() const

// {

// cout << " ========= ERROR INFO ========= " << endl;

// cout << " Error Content : " << text << endl;

// cout << " Source : " << source << endl;

// cout << " Line Number : " << line << endl;

// cout << " Date : " << date << endl;

// }

//};

//

//

//class OutOfRangeException :public Exception

//{

//public:

// OutOfRangeException(string text, string source, int line, string date)

// :Exception(text, source, line, date)

// {

//

// }

//

// void PrintMessage() const override

// {

// cout << " Error type : " << "OutOfRangeException" << endl;

// Exception::PrintMessage();

// }

//};

//

//class InvalidArgumentException : public Exception

//{

//public:

// InvalidArgumentException(string text, string source, int line, string date)

// :Exception(text, source, line, date)

// {

//

// }

//

// void PrintMessage() const override

// {

// cout << " Error type : " << "InvalidArgumentException" << endl;

// Exception::PrintMessage();

// }

//};

//

//template<typename T>

//class Array

//{

//private:

// T\* data;

// int size;

//public:

// int GetSize() const

// {

// return size;

// }

// Array<T>& operator=(const Array<T>& other) = delete;

// Array(const Array<T>& other) = delete; // kimsenin copy assignment (basqa seyler de ola biler) etmesine icaze vermir

//

// Array(int size) throw(InvalidArgumentException) // Bu kod blokunda cixa bilecek exceptionlari gosterir

// {

// if (size < 0 || size > 1000000)

// {

// throw new InvalidArgumentException("Size should be between 0 and 1000000",

// \_\_FILE\_\_, \_\_LINE\_\_, \_\_DATE\_\_);

// }

// this->size = size;

// data = new T[size]{};

// }

//

// T& operator[](int index) throw(OutOfRangeException) // Bu kod blokunda cixa bilecek exceptionlari gosterir

// {

// if (index < 0 || index >= size)

// {

// throw new OutOfRangeException("Index should be more than 0 and less than " << size,

// \_\_FILE\_\_, \_\_LINE\_\_, \_\_DATE\_\_);

// }

// return data[index];

// }

//};

//

//int main()

//{

// try

// {

// Array<int> a(-10);

// }

// catch (const Exception \*ex)

// {

// ex->PrintMessage();

// }

//

// return 0;

//}

// Task

#include <iostream>

using namespace std;

class Exception

{

protected:

string text;

string source; // file

int line;

string date;

public:

Exception(string text, string source, int line, string date)

:text(text), source(source), line(line), date(date) {}

string GetText() const

{

return text;

}

string GetSource() const

{

return source;

}

int GetLine() const

{

return line;

}

string GetDate() const

{

return date;

}

virtual void PrintMessage() const

{

cout << " ========= ERROR INFO ========= " << endl;

cout << " Error Content : " << text << endl;

cout << " Source : " << source << endl;

cout << " Line Number : " << line << endl;

cout << " Date : " << date << endl;

}

};

// Melumatda sehvlik oldu deye yazila bilinmedi

class DatabaseException : public Exception

{

public:

DatabaseException(string text, string source, int line, string date)

: Exception(text, source, line, date) {}

void PrintMessage() const override

{

cout << " Error type : " << "DatabaseException" << endl;

Exception::PrintMessage();

}

};

class User

{

string username;

string password;

string name;

string surname;

public:

User()

{

username = "";

password = "";

name = "";

surname = "";

}

User(const string& username, const string& password, const string& name, const string surname)

{

}

string GetUsername() const

{

return username;

}

string GetPassword() const

{

return password;

}

string GetName() const

{

return name;

}

string GetSurname() const

{

return surname;

}

void Show() const

{

cout << " ======== USER ======== " << endl;

cout << " Username : " << username << endl;

cout << " Password : ";

int size = password.length();

for (int x = 0; x < size; x++)

{

cout << "\*";

} cout << endl;

cout << " Name : " << name << endl;

cout << " Surname : " << surname << endl;

}

};

class RegistrationValidation

{

public:

bool UsernameExists(const string& username) const throw(DatabaseException)

{

}

};

class Database

{

private:

User\*\* users;

int user\_count;

public:

User\*\* GetUsers();

int GetUserCount();

void AddUser();

User\* GetUserByUsername(const string& username);

void UpdateUser(const User& old\_user, const User& new\_user);

void ShowUsers() const;

};

class Registration

{

Database \_database;

public:

Registration(const Database& database)

{

\_database = database;

}

};

int main()

{

return 0;

}