19/40 => 10/20

PILOT.cpp


```
/**********************
Capteur.h
**********************/
#pragma once
#include <string>

using namespace std;

class Capteur
{
protected :
        static int nombreDeCapteurs;
        string type;
        string noSerie;

public :
        Capteur(string type, string noSerie);
        string getType() { return type; }
        string getNoSerie() { return noSerie; }
        static int getNombreCapteurs() { return nombreDeCapteurs; }
        ~Capteur();
};



/**********************
CapteurTemperature.h
**********************/
#pragma once
#include "Capteur.h"
class CapteurTemperature : public Capteur
{
protected:
        double temperatureMinAutorisee;
        double temperatureMaxAutorisee;
public:
        CapteurTemperature(string numeroSerie, double min, double max);
        double getTemperature();
};



/**********************
CapteurTemperatureExterieur.h
**********************/
#pragma once
#include "CapteurTemperature.h"
class CapteurTemperatureExterieure : public CapteurTemperature
{
public:
        CapteurTemperatureExterieure(string derniersChiffresNumeroSerie, double min, double max);
        bool verifieConformite();
};



/**********************
Capteur.cpp
**********************/
#include "Capteur.h"

int Capteur::nombreDeCapteurs = 0;

Capteur::Capteur(string type, string noSerie)
{
        this->type = type;
        this->noSerie = noSerie;
}

Capteur::~Capteur()
{
        nombreDeCapteurs--;
}


/**********************
CapteurTemperature.cpp
**********************/
#include "CapteurTemperature.h"

CapteurTemperature::CapteurTemperature(string numeroSerie, double min, double max) : Capteur("TEMP",numeroSerie
)
{
        this->temperatureMinAutorisee = min;
        this->temperatureMaxAutorisee = max;
}
```

5/5

2/2

2/2

3/4

3/7

```
double CapteurTemperature::getTemperature()
{
	return 1000;
}


/**********************
CapteurTemperatureExterieur.cpp
**********************/
#include "CapteurTemperatureExterieur.h"

CapteurTemperatureExterieure::CapteurTemperatureExterieure(string derniersChiffresNumeroSerie, double min, doubl
e max) : CapteurTemperature(derniersChiffresNumeroSerie,min,max)
{
	this->noSerie = derniersChiffresNumeroSerie;
}

bool CapteurTemperatureExterieure::verifieConformite()
{
	return false;
}


/**********************
test.cpp
**********************/
#include <windows.h>
#include <vector>
#include <iostream>
#include "Capteur.h"
#include "CapteurTemperature.h"
#include "CapteurTemperatureExterieur.h"

using namespace std; // utilisation de l'espace de nommage standard


/*============================================
Fonction principale
============================================*/
int main()
{
	// Prise en compte des accents
	// Il faudra choisir la police de caractères Consolas ou Lucida
	SetConsoleOutputCP(1252);

	//==== Déclaration d'éventuelles variables locales ====



	// Affichage du nombre de capteurs

	cout << "ETAPE1 : NOMBRE DE CAPTEURS = " << Capteur::getNombreCapteurs() << " (attendu 0)" << endl << endl;

	CapteurTemperature* test[20];

	for (int i = 0; i < CapteurTemperature::getNombreCapteurs(); i++)
	{
		cout << test[i]->getNoSerie() << test[i]->getTemperature() ;
	}

	cin.get(); cin.ignore();
	return 0;
}

/**********************
TestCapteurs_A_Completer.cpp
**********************/
/*********************************************
Programme principal à compléter
Au final, l'exécution du programme doit
produire un résultat similaire à l'exécutable fourni
*********************************************/
#define _CRT_SECURE_NO_WARNINGS

#include <iostream> // bibliothèque de gestion des E/S
#include <conio.h>      // gestion de la console (ici _getch())
#include <windows.h>
#include <vector>

//...

using namespace std; // utilisation de l'espace de nommage standard
```

1/3

1

1

1

```
/*============================================
Fonction principale
============================================*/
int main()
{
	// Prise en compte des accents
	// Il faudra choisir la police de caractères Consolas ou Lucida
	SetConsoleOutputCP(1252);

	//==== Déclaration d'éventuelles variables locales ====
	//...

	// Affichage du nombre de capteurs
	cout << "ETAPE1 : NOMBRE DE CAPTEURS = " << ... << "  (attendu 0)" << endl << endl;

	// Créer un tableau de 20 capteurs de température (modèles de base)
	// Ces  capteurs de temperature  ont des numéros de séries allant de "111101" à "111120"
	// de temperature mini −20.5 et temperature maxi +80.5
	// ...

	// Affichage du nombre de capteurs
	cout << "ETAPE2 : NOMBRE DE CAPTEURS = " << ... << "  (attendu 20)" << endl << endl;

	// Afficher 8 mesures par capteurs (20 lignes de 8 valeurs)
	cout << "CAPTEURS DE TEMPERATURE (MODELE DE BASE) " << endl;
	cout.precision(4);	// 4 digits de précision
	//...

	// Supprimer de la mémoire 10 capteurs
	//...

	// Affichage du nombre de capteurs
	cout << "ETAPE3 : NOMBRE DE CAPTEURS = " << ... << "  (attendu 10)" << endl << endl;

	// Créer 12 capteurs de température extérieure dont les numéros de séries finissent par "5501" à "5512"
	// Les 11 premiers capteurs ont une temperature mini −85.5 et temperature maxi +300.5
	// le dernier capteur a une température mini de −45.5 et maxi de 60.6
	//...

	// Afficher 8 mesures par capteurs (12 lignes de 8 valeurs) et afficher pour chaque capteur s'il est con
forme ou non
	cout << "\n\nCAPTEURS DE TEMPERATURE (MODELE EXTERIEUR) " << endl;
	//...

	// Affichage du nombre de capteurs
	cout << "ETAPE4 : NOMBRE DE CAPTEURS = " << ... << "  (attendu 22)" << endl << endl;

	// Supprimer tous les capteurs
	//...

	// Affichage du nombre de capteurs
	cout << "ETAPE5 : NOMBRE DE CAPTEURS = " << ... << "  (attendu 0)" << endl << endl;

	_getch(); // attente d'appui sur une touche
	return 0; // sortie du programme
}
```