25/40 => 12,5 /20

PETRUZZELLIS.cpp


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

using namespace std;

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

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



/**********************
CapteurTemperature.h
**********************/
#pragma once
#include "Capteur.h"

class CapteurTemperature :
    public Capteur
{
protected:
    double temperatureMinAutorisee = -50;
    double temperatureMaxAutorisee = 60;

public:
    CapteurTemperature(string numeroSerie, double min, double max);
    double getTemperature();
};



/**********************
CapteurTemperatureExterieure.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"

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

string Capteur::getType()
{
        return this->type;
}

string Capteur::getNoSerie()
{
        return this->noSerie;
}

int Capteur::getNombreCapteurs()
{
        return nombreDeCapteurs;
```

4/5

1/2

2/4

int Capteur::nombreDeCapteurs=0;

```
}

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


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

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

double CapteurTemperature::getTemperature()
{
	double tempMax;
	tempMax = rand();

	return tempMax;
}


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

CapteurTemperatureExterieure::CapteurTemperatureExterieure(string derniersChiffresNumeroSerie, double min, doubl
e max)
	: CapteurTemperature(derniersChiffresNumeroSerie, min, max)
{
}

bool CapteurTemperatureExterieure::verifieConformite()
{
	if (temperatureMinAutorisee <= -50 && temperatureMaxAutorisee >= 60)
	{
		return true;
	}
	else
	{
		return false;
	}

}


/**********************
main.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>

#include "Capteur.h"
#include "CapteurTemperature.h"
#include "CapteurTemperatureExterieure.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;

	// 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
	CapteurTemperature* capteurModeleDeBase[20];

	for (unsigned i = 0; i < 20; i++)
	{
		capteurModeleDeBase[i] = new CapteurTemperature(to_string(111101 + i), −20.5, 80.5);
	}

	// Affichage du nombre de capteurs
	cout << "ETAPE2 : NOMBRE DE CAPTEURS = " << Capteur::getNombreCapteurs << " (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
	for (unsigned i = 0; i < 20; i++)
	{
		cout << "Capteur de numero de serie : " << capteurModeleDeBase[i]->getNoSerie() << endl;
	}

	// Supprimer de la mémoire 10 capteurs
	for (unsigned i = 0; i < 10; i++)
	{
		delete capteurModeleDeBase[i];
	}

	// Affichage du nombre de capteurs
	cout << "ETAPE3 : NOMBRE DE CAPTEURS = " << Capteur::getNombreCapteurs << " (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
	CapteurTemperatureExterieure* capteurExt[12];
	for (unsigned i = 0; i < 11; i++)
	{
		capteurExt[i] = new CapteurTemperatureExterieure(to_string(5501 + i), −85.5, 300.5);
	}
	capteurExt[11] = new CapteurTemperatureExterieure(to_string(5512), −45.5, 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;
	cout.precision(4);	// 4 digits de précision
	for (unsigned i = 0; i < 12; i++)
	{
		cout << "Capteur de numero de serie : " << capteurExt[i]->getNoSerie() << endl;

	}

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

	// Supprimer tous les capteurs
	delete capteurExt;
	delete capteurModeleDeBase;

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

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