25,5/40 => 12,25/20

# JEAN.cpp



```
/**********************
Capteur.h
**********************/
```
5/5

```
#pragma once
#include <string>
using namespace std;

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

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

};



/**********************
CapteurTemperature.h
**********************/
```
2/2

```
#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
**********************/
```
2/2

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



/**********************
Capteur.cpp
**********************/
```
4/4

```
#include "Capteur.h"

int Capteur::nomberDeCapteurs=0;
Capteur::Capteur(string type, string noSerie)
{
        this->type = type;
        this->noSerie = noSerie;
        nomberDeCapteurs++;
}

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

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

int Capteur::getNombreCapteurs()
{
        return nomberDeCapteurs;
}
```

```
Capteur::~Capteur()
{
	nomberDeCapteurs--;
}


/**********************
CapteurTemperature.cpp
**********************/ 3/7
#include "CapteurTemperature.h"


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

double CapteurTemperature::getTemperature()
{
	return rand();

}

/**********************
CapteurTemperatureExterieur.cpp 0/3
**********************/
#include "CapteurTemperatureExterieur.h"


CapteurTemperatureExterieure::CapteurTemperatureExterieure(string derniersChiffresNumeroSerie, double min, doubl
e max):CapteurTemperature(derniersChiffresNumeroSerie, -50, 60)
{
}

bool CapteurTemperatureExterieure::verifieConformite()
{
	if (this->noSerie.length() == 6 && this->noSerie[0] == 3 && this->noSerie[1] == 3)
	{
		return true;
	}
	else return false;
}

/**********************
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>
#include "Capteur.h"
#include "CapteurTemperature.h"
#include "CapteurTemperatureExterieur.h"
//...

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


/*=============================================
Fonction principale 9,5/17
=====================================*/
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 = " << CapteurTemperature::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* lesVingtsCapteurs[20];
        for (int i = 0; i < 20; i++)
        {

                lesVingtsCapteurs[i] = new CapteurTemperature("1111" + i, -20.5, 80.5);
        }

        // Affichage du nombre de capteurs
        cout << "ETAPE2 : NOMBRE DE CAPTEURS = " << CapteurTemperature::getNombreCapteurs() << " (attendu 20)" << end
l << 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 (int i = 0; i < CapteurTemperature::getNombreCapteurs(); i++)
        {
                for (int y = 0; y < 8; y++)
                {
                        double x = lesVingtsCapteurs[i]->getTemperature();
                }
                cout << endl;
        }

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

        // Affichage du nombre de capteurs
        cout << "ETAPE3 : NOMBRE DE CAPTEURS = " << CapteurTemperature::getNombreCapteurs() << " (attendu 10)" << end
l << 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* lesDouzesCapteurs[12];
        for (int i = 1; i < 13; i++)
        {
                if (i < 12)
                {
                        lesDouzesCapteurs[i] = new CapteurTemperatureExterieure("55" + i, -85.5, 300.5);
                }
                if (i == 12)
                {
                        lesDouzesCapteurs[i] = new CapteurTemperatureExterieure("55" + i, -45, 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;
        for (int i = 0; i < 13; i++)
        {
                for (int y = 0; y < 8; y++)
                {
                        cout << lesDouzesCapteurs[i]->getTemperature() << "     ";
                }
                cout << lesDouzesCapteurs[i]->verifieConformite() << endl;
        }

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

        // Supprimer tous les capteurs
        for (int i = 0; i < CapteurTemperature::getNombreCapteurs(); i++)
        {
                delete lesDouzesCapteurs[i];
                delete lesVingtsCapteurs[i];
        }

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


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