	Lycée de l'Hyrôme - Chemillé	2010 - 2011
	Classes	Langage C++
BTS IRIS		TP5

Objectifs

- Savoir déclarer une classe.
- Savoir définir une classe.
- Savoir instancier une classe.
- Maîtriser les constructeurs et les destructeurs.
- Mettre en œuvre les opérateurs new et delete.

Ressources disponibles

- Un PC.
- Un RAD C++ Builder 6.

1. Remarques

Il faudra pour chaque programme :

- créer un répertoire de travail (dans lequel seront enregistrés les différents fichiers),
- créer un projet,
- créer un « .h » et un « .cpp » pour chaque classe.

2. Premier programme

2.1. Présentation

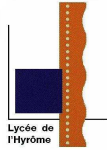
Cette classe permet de gérer un caractère.

Le projet se nommera « testCaractere ».

2.2. Déclaration de la classe

Dans le fichier Caractere.h nous déclarerons la classe.

```
#ifndef CaractereH                // pour éviter les inclusions multiples
#define CaractereH
//-----
class Caractere
{
    private:                        // donnée encapsulée
        char car;
    public:
        void initialise(char valCar); // permet d'initialiser le caractère
        void afficheCar();           // permet d'afficher le caractère
};
//-----
#endif
```

	Lycée de l'Hyrôme - Chemillé	2010 - 2011
	Classes	Langage C++
	BTS IRIS	TP5

2.3. Définition de la classe

Dans le fichier Caractere.cpp nous définirons la classe.

```
#include <ctype.h>
#include <iostream.h>
//-----
#pragma hdrstop
#include "caractere.h"
//-----
void Caractere::initialise(char valCar)
{
    car=toupper(valCar);
}

//-----
void Caractere::afficheCar()
{
    cout<<"Le caractere est : "<<car<<endl;
}
```

Ecrire ce fichier et le sauvegarder sous votre répertoire.

Inclure ce fichier dans votre projet.

Compiler le fichier et corriger les erreurs éventuelles.

2.4. Programme principal

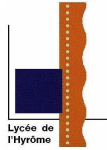
Ce fichier se nommera « princi.cpp ».

Le programme principal sera un programme en C permettant de tester notre classe.

```
//-----
#include <conio.h>
#pragma hdrstop
#include "caractere.h"
//-----
#pragma argsused

void main()
{
    Caractere c1;
    Caractere c2;
    clrscr();

    c1.initialise('a');
    c1.afficheCar();
    c2=c1;
```

	Lycée de l'Hyrôme - Chemillé	2010 - 2011
	Classes	Langage C++
BTS IRIS		TP5

```

        c2.afficheCar();
    }

```

Ecrire ce fichier et le sauvegarder sous votre répertoire.

Inclure ce fichier dans votre projet.

Compiler le fichier et corriger les erreurs éventuelles.

Réaliser une édition de lien et corriger les erreurs éventuelles.

Exécuter le programme en pas à pas, en affichant les différents attributs et variables.

3. Deuxième programme

3.1. Classe Caractere

Nous allons améliorer la classe Caractere.

3.2. Déclaration de la classe

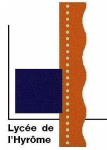
Dans le fichier Caractere.h nous déclarerons la classe.
Le caractère sera en majuscule.

```

#ifndef CaractereH
#define CaractereH
//-----
#include "minmaj.h"

class Caractere
{
    private:
        char car;
    public:
        Caractere(char valCar);
        char minuscule();
        char majuscule();
        void afficheCar(TypeCara minMaj);
        void afficheCodeAscii(TypeCara minMaj);
        char carInferieur();
        char carSuperieur();
};
//-----
#endif

```

	Lycée de l'Hyrôme - Chemillé	2010 - 2011
	Classes	Langage C++
	BTS IRIS	TP5

Remarque:

Avant d'écrire la déclaration de la classe, il faudra créer dans un fichier « minmaj.h » un type énuméré « TypeCara ».

```
#ifndef MinmajH
#define MinmajH
//-----
enum TypeCara
{
    MIN,
    MAJ
};
//-----
#endif
```

Ecrire ce fichier.

3.3. Définition de la classe

Dans le fichier Caractere.cpp nous définirons la classe.

```
#include <ctype.h>
#include <iostream.h>
//-----
#pragma hdrstop
#include "caractere.h"
//-----
/* Initialise le caractère en majuscule */

Caractere::Caractere(char valCar)
{
}

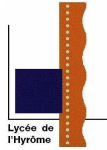
/* *****

****/
/* Retourne le caractère en minuscule */

char Caractere::minuscule()
{
}

/* *****
/* Retourne le caractère en majuscule */

char Caractere::majuscule()
{
}
```

	Lycée de l'Hyrôme - Chemillé	2010 - 2011
	Classes	Langage C++
		TP5

```

/*****
****/

/* Affiche le caractère en majuscule ou en minuscule suivant le paramètre.
L'affichage se présentera sous la forme : Le caractère est A */

void Caractere::afficheCar(TypeCara minMaj)
{
}

/*****
/

/* Affiche le code ASCII suivant le paramètre.
L'affichage se présentera sous la forme : Le code ascii de A est 65 */

void Caractere::afficheCodeAscii(TypeCara minMaj )
{
}

/*****
****/

/* Renvoie éventuellement le caractère inférieur, sinon elle renvoie 0 */

char Caractere::carInferieur()
{
}

/*****
****/

/* Renvoie éventuellement le caractère supérieur, sinon elle renvoie 0 */

char Caractere::carSuperieur()
{
}

```

Ecrire ce fichier et le sauvegarder sous votre répertoire.

3.4. Programme principal

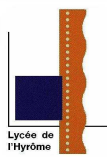
Ce fichier se nommera « princi.cpp ».

Le programme principal sera un programme en C permettant de tester notre classe.

```

//-----
#include <iostream.h>
#include <conio.h>
#pragma hdrstop
#include "caractere.h"
//-----
#pragma argsused

```

	Lycée de l'Hyrôme - Chemillé	2010 - 2011
	Classes	Langage C++
BTS IRIS		TP5

```

void main()
{
    Caractere c1('j');
    clrscr();

    c1.afficheCar(MIN);
    c1.afficheCar(MAJ);
    c1.afficheCodeAscii(MIN);
    c1.afficheCodeAscii(MAJ);
    cout<<endl<<"Le caractere est : "<<c1.minuscule();
    cout<<endl<<"Le caractere est : "<<c1.majuscule();
    cout<<endl<<"Le caractere superieur est : "<<c1.carSuperieur();
    cout<<endl<<"Le caractere inferieur est : "<<c1.carInferieur();
}

```

Tester le programme.

4. Troisième programme

4.1. Classe Chaîne

Nous allons écrire une classe nommée Chaîne qui permet de gérer un objet chaîne.

Il est interdit dans la suite du problème d'utiliser les fonctions « c » relatives au traitement des chaînes (ne pas inclure <string.h>)

Le programme « .exe » est fourni à titre d'exemple.

4.2. Déclaration de la classe

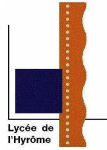
Dans le fichier Chaîne.h nous déclarerons la classe.

```

#ifndef ChaîneH
#define ChaîneH
//-----
class Chaîne
{
    private:
        int nbMaxCar;           // nombre maxi de caractères
        int nbCar;              // nombre courant de caractères
        char * tab;              // chaîne (en dynamique)

    public:
        Chaîne(int valNb);       // constructeur
        ~Chaîne();               // destructeur
        int taille();            // taille de la chaîne
        void vide();              // vide la chaîne
        void affiche();           // affiche la chaîne
        bool ajouteCaractere(char cara); // ajoute un caractère en fin
        char retireCaractere();   // tire le caractère de fin
}

```

	Lycée de l'Hyrôme - Chemillé	2010 - 2011
	Classes	Langage C++
BTS IRIS		TP5

```

        void minuscule();           // mise en minuscule de la chaîne
        void majuscule();          // mise en majuscule de la chaîne
    private:
        void init();

};
//-----
#endif

```

Ecrire ce fichier et le tester.

4.3. Définition de la classe

Dans le fichier Chaine.cpp nous définirons la classe.

```

#include <ctype.h>
#include <iostream.h>
//-----
#pragma hdrstop
#include "chaine.h"

//-----
/*****
/* Initialise nbMaxCar
   Initialise nbCar
   Réserve la mémoire pour la chaîne (en fonction du paramètre, ne pas oublier l'indicateur de fin de
   chaîne)
   Met à Zéro le tableau en utilisant la méthode init()*/

Chaine::Chaine(int valNbCar)
{
}

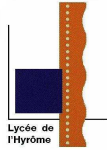
/*****
/* Met à Zéro les cases de la chaîne */

void Chaine::init()
{
}

/*****
/* libère la mémoire */

Chaine::~Chaine()
{
}

```

	Lycée de l'Hyrôme - Chemillé	2010 - 2011
	Classes	Langage C++
BTS IRIS		TP5

```

/*****/
/* Vide la chaîne (appel à la méthode init()) */

void Chaine::vide()
{
}

/*****/
/* Ajoute, si possible, un caractère en fin de chaîne
si l'opération a réussi on renvoie true, sinon on renvoie false */

bool Chaine::ajouteCaractere(char cara)
{
}

/*****/
/* Retire, si possible le dernier caractère de la chaîne
si l'opération a réussi on renvoie le caractère retiré, sinon on renvoie 0 */

char Chaine::retireCaractere()
{
}

/*****/
*****/
/* Affichage de la chaîne ou du message « chaîne vide » */

void Chaine::affiche()
{
}

/*****/
/* Transformation de la chaîne en minuscule (sans utiliser de fonction) */

void Chaine::minuscule()
{
}

/*****/
/* Transformation de la chaîne en majuscule (sans utiliser de fonction) */

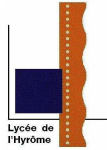
void Chaine::majuscule()
{
}

/*****/
*****/
/* Retourne la taille actuelle de la chaîne */

int Chaine::taille()
{
}

```

Ecrire ce fichier et le tester.

	Lycée de l'Hyrôme - Chemillé	2010 - 2011
	Classes	Langage C++
BTS IRIS		TP5

4.4. IHM

Ce fichier se nommera « IHM.cpp ».

L'interface sera un programme en « C » permettant de tester notre classe.

Bien qu'étant écrit en « C », il utilisera les possibilités de l'environnement « C++ » (cin, cout).

Il se présentera sous une forme de menu.

Menu

1. Ajoute caractere
2. Retire caractere
3. Affiche chaîne
4. Vide chaîne
5. Taille de la Chaîne
6. Caracteres en minuscule
7. Caracteres en majuscule
8. Fin

Le fichier « h » est fourni ci-dessous:

```
//-----
#ifndef IHMH
#define IHMH
//-----
#pragma hdrstop
#include "chaîne.h"
//-----
void lancer(void);
void traiteChoix(char choix, Chaîne &c1);
char saisirChoix(void);
//-----
#endif
```

Contraintes :

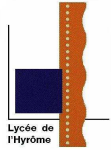
Avant d'instancier la classe « Chaîne », on demandera à l'utilisateur la taille maxi de la chaîne.

L'Ajout d'un caractère se fera en fin de chaîne si possible. Si l'opération a échoué on affichera un message d'erreur (chaîne pleine).

La suppression d'un caractère se fera en fin de chaîne si possible. Si l'opération a échoué on affichera un message d'erreur (chaîne vide).

Ecrire et tester ce programme.

5. Programme principal

 Lycée de l'Hyrôme	Lycée de l'Hyrôme - Chemillé	2010 - 2011
	Classes	Langage C++
BTS IRIS		TP5

Ce programme nommé « princi.cpp » fera uniquement appel à la fonction « lancer » du fichier « ihm.cpp ».

Tester le programme complet.