	Lycée de l'Hyrôme - Chemillé	2012 - 2013
	<b>Classes de service</b>	<b>Langage C++</b>
<b>BTS IRIS</b>		<b>TPx</b>

## Objectifs

- Mise en œuvre des concepts de base du C++.
- Tester une classe.

## Ressources disponibles

- Un PC.
- Un EDI C++.

## 1. Présentation

Nous allons réaliser et tester deux classes de service qui pourront être utilisées ultérieurement dans différentes applications.

Ces deux classes seront une classe mettant en œuvre une temporisation et une classe permettant de générer des nombres aléatoires.

## 2. Classe de temporisation

### 2.1. Introduction

**Fonctionnalité :** Cette classe permettra de réaliser une temporisation dont l'unité sera soit la milliseconde, soit la seconde.

Cette classe utilisera deux fonctions de l'API :

Sleep pour les millisecondes (déclarée dans windows.h)  
\_sleep pour les secondes (déclarée dans dos.h)

**Nom de la classe :** Tempo

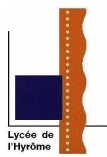
**Nom des fichiers :** Tempo.h Tempo.cpp

### 2.2. Les attributs

- unite :

Type : caractère

Rôle : unité de la temporisation ('s': seconde ou 'm': milliseconde)

	Lycée de l'Hyrôme - Chemillé	2012 - 2013
	<b>Classes de service</b>	<b>Langage C++</b>
<b>BTS IRIS</b>		<b>TPx</b>

- valeurTempo :

Type : entier non signé.

Rôle : Valeur de la temporisation courante.

### 2.3. Les méthodes

Toutes les méthodes sont publiques.

- getUnite

Fonctionnalité : Indique l'unité de la temporisation.

Signature : char getUnite();

- setUnite

Fonctionnalité : Modifie la valeur de l'unité 's' s'il le paramètre est 's' ou 'S', 'm' dans les autres cas.

Signature : void setUnite(char paraUnite);

- getValeurTempo

Fonctionnalité : Indique la valeur de la tempo.

Signature : unsigned int getValeurTempo();

- setValeurTempo

Fonctionnalité : Modifie la valeur de la tempo.

Signature : void setValeurTempo(unsigned int val);

- Tempo

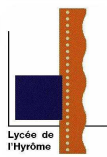
Fonctionnalité : Ce constructeur permet d'initialiser la valeur de la temporisation, ainsi que l'unité de temporisation (valeur par défaut 100 pour la valeur et milliseconde pour l'unité).

Signature : Tempo(unsigned int val=100,char paraUnite='m');

- temporisation

Fonctionnalité : Réalise la temporisation.

Signature : void temporisation();

	Lycée de l'Hyrôme - Chemillé	2012 - 2013
	<b>Classes de service</b>	<b>Langage C++</b>
<b>BTS IRIS</b>		<b>TPx</b>

## 2.4. Travail à effectuer

Réaliser la classe et la tester.  
Le programme de test sera un objet.

## 3. Classe de nombres aléatoires

### 3.1. Introduction

**Fonctionnalité :** Cette classe permettra de générer des nombres aléatoires qui seront éventuellement sauvegardés dans un tableau. On pourra également indiquer si les nombres sont uniques.

**Nom de la classe :** Random

**Nom des fichiers :** Random

**Remarque :**

On devra utiliser les fonctions en C random et rand.

### 3.2. Les attributs

- tabRandom :

Type : pointeur d'entier.

Rôle : tableau de sauvegarde des nombres aléatoire.

- unique :

Type : booléen.

Rôle : indique si les données sont uniques.

- nbRandom :

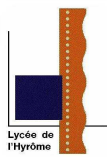
Type : entier non signé.

Rôle : nombre de données aléatoires

- maxi :

Type : entier non signé..

Rôle : valeur maxi des nombres aléatoires.

	<b>Lycée de l'Hyrôme - Chemillé</b>	2012 - 2013
	<b>Classes de service</b>	<b>Langage C++</b>
<b>BTS IRIS</b>		<b>TPx</b>

- mini :

Type : entier non signé.

Rôle : valeur mini des nombres aléatoires.

### 3.3. Les méthodes

Les méthodes "calculValeur" et "rechercheValeur" sont privées les autres sont publiques.

- Random :

Fonctionnalité : Ce constructeur permet d'initialiser les attributs (taille par défaut du tableau 1 et unique à faux).

Signature : Random(int paraMini,int paraMaxi,unsigned int paraNbRandom=1,bool paraUnique=false);

- ~Random :

Fonctionnalité : Libération de la mémoire.

Signature : ~Random();

- rechercheValeur :

Fonctionnalité : indique si la valeur spécifiée est déjà dans le tableau.

Signature : bool rechercheValeur(int donnee);

- calculValeur :

Fonctionnalité : retourne une valeur aléatoire comprise dans la fourchette.

Signature : int calculValeur();

- initTab :

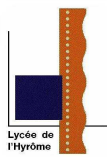
Fonctionnalité : Rempli le tableau avec des valeurs aléatoires (unique ou pas selon le cas).

Signature : void initTab();

- getNbRandom :

Fonctionnalité : Indique la taille du tableau.

Signature : unsigned int getNbRandom();

	Lycée de l'Hyrôme - Chemillé	2012 - 2013
	<b>Classes de service</b>	<b>Langage C++</b>
<b>BTS IRIS</b>		<b>TPx</b>

- setNbRandom :

Fonctionnalité : Modifie la taille du tableau et le remplit.

Signature : void setNbRandom(unsigned int nbVal);

- getMaxi :

Fonctionnalité : Indique la valeur aléatoire maxi possible.

Signature : unsigned int getMaxi();

- getMini :

Fonctionnalité : Indique la valeur aléatoire mini possible.

Signature : unsigned int getMini();

- setMaxi :

Modifie la valeur aléatoire maxi possible.

Signature : void setMaxi(int val);

- setMini :

Fonctionnalité : Modifie la valeur aléatoire mini possible.

Signature : void setMini(int val);

- int getDonnee(unsigned int indice);

Fonctionnalité : Indique la valeur aléatoire de la case spécifiée.

Si l'indice n'est pas valide, on retournera une valeur égale à mini-1.

Signature : int getDonnee(unsigned int indice);

- operator[] :

Fonctionnalité : Indique la valeur aléatoire de la case spécifiée.

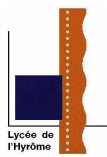
Si l'indice n'est pas valide, on retournera une valeur égale à mini-1.

Signature : int operator[](unsigned int indice);

- Modifie :

Fonctionnalité : Modifie la donnée aléatoire de la case spécifiée (si l'indice est valide) et la retourne (sinon retourne -1).

Signature : int modifie(unsigned int indice=0);

 Lycée de l'Hyrôme	<b>Lycée de l'Hyrôme - Chemillé</b>	2012 - 2013
	<b>Classes de service</b>	<b>Langage C++</b>
<b>BTS IRIS</b>		<b>TPx</b>

### 3.4. Travail à effectuer.

Réaliser la classe et la tester.