TS2 –ELECTRONIQUESUJET : SCIENCES INFORMATIQuES

# sciences informatiques

# (30 H)

## Compétences

­­­– Programmer en C++

## Objectifs

L’étudiant doit être capable de :

– Analyser un problème donné

– Faire un algorithme de résolution

– Faire un programme en C++

– Rédiger un compte rendu

Utiliser la programmation C++ dans le domaine de spécialité

## Evaluation

L’étudiant doit être testé sur sa capacité à atteindre les compétences demandées lors d’un examen pratique et lors des séances des TP en salle informatique.

L’évaluation sera en fonction des critères suivants :

– Compte rendu à la fin du TP comportant :

\* L’analyse du problème à résoudre

\* L’algorithme

\* Le listing du programme

\* Le programme exécutable sur disquette

– Autonomie de l’étudiant

– Clarté de la présentation

– Argumentation de la démarche suivie

– Exactitude des résultats obtenus

– Capacité à exécuter une tache donnée

## Contenu

Chapitre 1  
introduction a la programmation C++ (4 H)

1.1 Ecriture sur la sortie standard avec **cout**

1.2 Lecture sur l’entrée standard avec **cin**

1.3 Exemples sur la sortie et entrée standard

1.4 Commentaire de fin de ligne

1.5 Déclaration et initialisations en C++

1.6 Notion de référence

1.6.1 Notion générale de référence en C++

1.6.2 Initialisation de référence

1.6.2 Transmission d’arguments par référence

1.6.3 Transmission par référence de la valeur de retour d’une fonction

1.6.4 Exemples

1.7 Les arguments par défaut

1.8 Sur définition de fonctions

1.8.1 Mise en œuvre de la sudéfinition de fonctions

1.8.2 Règles de recherche d’une fonction surdéfinie

1.8.3 Exercices

1.9 Les opérateurs **new** et **delete**

1.10 La spécification **inline**

Chapitre 2   
Classes et objets (10 H)

2.1 Introduction à la programmation orientée objet

2.1.1 La problématique de la programmation structurée

2.1.2 Introduction à la notion de classe et d’objet

2.1.3 Introduction à la notion d’encapsulation

2.1.4 Introduction à la notion de polymorphismes

2.1.5 Introduction à la notion de l’héritage

2.2 Les classe

2.2.1 Notion de classes

2.2.2 Déclaration de classe comportant des fonctions membre

2.2.3 Définition des fonctions membre d’une classe

2.2.4 Utilisation d’une classe comportant des fonctions membre

2.2.5 Exemples récapitulatif

2.3 Notion d’objet

2.4 Affectation d’objets

2.5 Notion de constructeur et de destructeur

2.6 Les objets automatiques et statiques (Durée de vie et appel de constructeurs et des destructeurs)

2.7 Les objets dynamiques (pointeur à un objet)

2.8 Les membres donnés statique

2.9 Objet membre d’une classe

2.10 Le tableau d’objets

2.11 Exploitation d’une classe (Notion de composant logiciel)

Chapitre 3   
Les propriétés des fonctions membre (6 H)

3.1 Surdéfinition (surchangement) des fonctions membre

3.2 Argument par défaut

3.3 Les fonctions membre **inline**

3.4 Objets transmis en argument d’une fonction membre

3.5 Mode de transmission des objets en argument (transmission par adresse d’un objet et transmission par référence problème de la transmission par valeur)

3.6 Objet comme valeur de retour d’une fonction

3.7 Le mot clés **this**

3.8 Exercices

Chapitre 4   
Constructeur de Recopie (3 H)

4.1 Présentation du constructeur de recopie

4.2 Emploi du constructeur de recopie par défaut

4.3 Définition d’un constructeur de recopie

4.4 Exemples d’utilisation de constructeurs de recopie (Objet transmis par valeur à une fonction, objet transmis en valeur de retour d’une fonction, initialisation d’un objet par un autre lors de la déclaration)

4.5 Exercices

Chapitre 5   
Les fonctions amies (3 H)

5.1 Fonction indépendante amie d’une classe

5.2 Fonction membre d’une classe, amie d’une autre classe

5.3 Fonction amie de plusieurs classes

5.4 Toutes les fonctions d’une classe sont amies d’une autre classe

5.5 Exercices

Chapitre 6   
La surdéfinition d’opérateurs (4 H)

6.1 Mécanisme de surdéfinition d’opérateur.

6.2 Surdéfinition d’opérateur avec une fonction amie.

6.3 Surdéfinition d’opérateur avec une fonction membre.

6.4 Opérateurs de transmission par référence.

6.5 Les limites et les possibilités de surdéfinition d’opérateurs (il faut se limiter aux opérateurs existants, étude du cas des opérateurs ++ et ––).

6.6 Les conversions.

6.7 Exemple de surdéfinition de l’opérateur =.

6.8 Notion de forme canonique d’une classe.

6.9 Exemple de surdéfinition de l’opérateur [ ].

6.10 Surdéfinition de l’opérateur ( ).

6.11 Surdéfinition de l’opérateur new et delete.

6.12 Exercices de Surdéfinition de quelques opérateurs.