UNIVERSITÉ LIBANAISE
FACULTÉ DES SCIENCES
SECTION I

*CONTENU DES COURS*
*DU DIPLÔME DE*

# MAITRISE

# D'INFORMATIQUE

## I. MP11: ANALYSE MATHMATIQUE (120h)

*Corps des réels. Suites numériques. Fonctions numériques d'une variable réelle. Dérivées des fonctions numériques. Fonctions logarithmes et fonctions associées. Développements limités. Fonctions de plusieurs variables. Intégrales de Riemann. Série entière. Séries numériques. Equations différentielles.*

## II. MP12: ANALYSE VECTORIELLE (36h de cours + 36h de T.D)

*Fonctions de plusieurs variables réelles. Géométrie des surfaces. Intégrales doubles et triples. Intégrales curvilignes. Intégrales de surfaces. Théorie des champs.*

## III. MP13: ALGEBRE I (36h de cours + 36h de T.D)

*Notions de logique. Ensembles. Relation d'équivalence. Applications et propriétés des applications. Lois de composition interne. Monoïde. Groupes. Anneaux et corps.*

## IV. MP14: ALGEBRE LINEAIRE (36h de cours + 36h de T.D)

*Corps des nombres complexes. Espaces vectoriel. Matrices. Déterminant. Système d'équations linéaires. Polynômes. Réduction des matrices.*

## V. MP15: PHYSIQUES I (Mécanique et onde) (96h)

*Mécanique: Cinématique du point matériel. Système de particules.*
*Oscillations et ondes: Oscillateur harmonique simple. Les ondes dans les milieux élastiques. Ondes sonores.*
*Optique géométrique: Lois générales de l'optique géométrique.*

## VI. MP16: PHYSIQUES II (Electrostatique, électricité) (72h)

*Champ électrostatique. Flux du champ électrostatique. Conducteur en équilibre électrostatique. Electrocinétique.*

## VII. MP17: CHIMIE GENERALE (60h)

*L'atome. Electrons dans les atomes. Classification périodique des élémenis. Liaisons chimiques. Liaisons dans le modèle ondulatoire. Liaisons intermoléculaires. Etat gazeux. Etat liquide. Etat solide. Les solutions. Thermodynamique chimique. Equilibre chimique.*

## I. M21: CALCUL DIFFERENTIEL (60h de cours + 60h de T.D)

*Suites et séries de fonctions. Séries trigonométriques. Les transformations (de Laplace, de Fourier en z). Equations différentielles du premier ordre. Equations différentielles du second ordre. Systèmes d'équations différentielles.*
*Fonction de $R^n \rightarrow R^m$ Fonctions complexes. Fonctions élémentaires dans C. Transformations conformes. Intégrales complexes. Série complexes. Théorème des résidus.*

## II. M22: TOPOLOGIE METRIQUE (60h de cours + 60h de T.D)

*Nombres réels. Espaces métriques. Espaces métriques compact. Espaces normés. Espaces de Hilbert. Espace des fonctions. Intégrale de Riemann.*

## III. M23: ALGEBRE II (60h de cours + 60h de T.D)

*Arithmétiques. Groupes. Groupes de permutations. Anneaux. Division dans un anneau. Anneaux principaux. Anneau de polynômes. Théorie du rang. Réduction des endomorphismes et des matrices. Espace dual. Algèbre bilinéaire. Espaces euclidiens.*

## IV. M24: PROBABILITES ET STATISTIQUES (144h)

***Probabilité** (96h).*
*Théorie des ensembles. Analyse combinatoire. Espace de probabilité. Probabilité conditionnelle et indépendance. Variables aléatoires. Vecteurs aléatoires. Espérance mathématique et moments. Lois de probabilité. Convergences et théorèmes limites.*
***Statistique descriptive, inférence statistique:** (48h).*
*Généralités. Etude d'une série à une variable. Etude d'une série à deux variables. Droites de régression.*
*Estimation, test d'hypothèse. Ajustement d'une distribution.*

## V. M25: INFORMATIQUE (120h).

*Algorithmique et programmation (72h). Informatique générale (48). (voir détail plus loin).*

# ALGORITHMIQUE & PROGRAMMATION – M25 (72H)

## *Objectifs*

L'objectif de ce module est de former des élèves capables de procéder avec méthode afin de construire les algorithmes de façon systématique. Parmi toutes les méthodes disponibles, la méthode de décomposition (approche descendante) constitue la véritable essence de la programmation sur ordinateur et la base de l'application de mécanismes relativement simples à des problèmes d'une énorme complexité. La structuration des instructions sous-jacentes aux langages de haut niveau tels que le langage C pour implanter les algorithmes donne au texte algorithmique un haut degré de clarté et d'ordre. Nos objectifs se résument ainsi :

- Maîtriser les mécanismes d'abstraction afin de pouvoir analyser un problème et concevoir, de façon systématique, des algorithmes corrects et adéquats.
- Implémenter moyennant un langage de programmation de haut niveau les algorithmes solutionnant un problème particulier.

*ALGORITHMIQUE*

### I. Concepts et éléments de base (2h)

A. Définitions et exemples
   1. *Notion d'acteur*
   2. *Situations initiale et finale*
   3. *Algorithme*

B. Cas particulier d'algorithme : le programme

C. Actions élémentaires
   1. *Définition*
   2. *Exemples*

D. Structure de contrôle
   1. *La séquence*
   2. *La sélection*
   3. *La répétition*

E. Exemples d'algorithmes

### II. Méthode de développement par raffinages successifs (2h+2hTD)

A. Approche descendante
   1. *Définition*
   2. *Principe de raffinage*

B. Analogie avec d'autres domaines

C. Exemple : recherche d'une chaîne de caractères dans une autre

D. Avantages et inconvénients

E. Exercices

### III. La machine algorithmique (5h+3hTD)

A. Description de la machine

B. Le langage algorithmique
   1. *Représentation des données*
      a) Type : domaine, opérations
      b) Variables et constantes
      c) Expressions
   2. *Instructions élémentaires*
      a) Affectation
      b) Lecture

SITÉ LIBAN

c) Ecriture
3. *Composition d'instructions*
a) Séquence
b) Sélection : simple et multiple
c) Répétition : tant que, répéter, pour
4. *Structure d'un algorithme*
a) Déclaration et définition des données
b) Corps
c) Exemples et exercices

C. Analyse d'un algorithme
1. *Etat d'un algorithme*
2. *Notion de situation*
a) Situations initiale, intermédiaires, finale
3. *Tableau de situations*

## PROGRAMMATION EN LANGAGE C

### IV. Les aspects classiques (2h+4hTD+6hTP)

A. Les éléments de base
B. Les déclarations
C. Les opérateurs
D. Les expressions
E. Les instructions
F. Les entrées/sorties élémentaires
G. Données structurées
1. *Les tableaux : unidimensionnels et bidimensionnels*
a) Déclaration
b) Manipulation : accès à un élément
c) Exercices
(1) Tri d'un tableau
(2) Recherche d'un élément dans un tableau

### V. Les sous-programmes (3h+3hTD+8hTP)

A. Définition d'une fonction
1. *L'entête de la fonction*
2. *Le corps de la fonction*
3. *Exemples*
B. Appel de fonction
C. Passage de paramètres
D. Les variables locales
1. *La validité des variables locales*
E. Les variables globales
F. La récursivité
G. Exemples.

### VI. Structures (1h+3hTD+2hTP)

A. Déclaration de structures
B. Initialisation de structures
C. Accès à un champ
D. Le surnommage de type (typedef)
E. Utilisation des structures
F. Exemples

### VII. Pointeurs et tableaux (4h+5hTD+7hTP)

A. Pointeurs et adresses
B. Opérations sur les pointeurs
C. Gestion dynamique de la mémoire
D. Pointeurs et tableaux
E. Passage de paramètres
F. Tableaux de pointeurs
G. Pointeurs de fonctions
H. Exemples

## VIII. Les entrées/sorties (2h+3hTD+5hTP)

A. Les entrées/sorties standards
B. La manipulation des fichiers
   1. *Ouverture/fermeture d'un fichier*
   2. *Les entrées/sorties de caractères*
   3. *Les entrées/sorties formatées*
   4. *Les entrées/sorties de lignes de caractères*
   5. *Les entrées/sorties de blocs d'informations*

# INFORMATIQUE GENERALE – M25 (48H)

## I. Histoire de l'ordinateur (2h)

A. Introduction
   1. *Développement historique et conceptuel*
   2. *Naissance de l'ordinateur*
   3. *Naissance de l'industrie informatique.*

## II. Présentation générale (6h)

A. Ordinateur et informatique
B. Principaux éléments d'un ordinateur
C. Utilisation des ordinateurs
D. Principe de fonctionnement.

## III. Représentation interne des informations (3C+3TD)

A. Données non numériques
B. Données numériques
   1. *Entiers positifs ou nuls*
   2. *Entiers négatifs*
   3. *Nombres fractionnaires*
   4. *Décimaux codés en binaire*
C. Codes détecteurs et correcteurs d'erreurs.

## IV. Algèbre de Boole (5C+5TD)

A. Définitions
B. Propriétés et théorèmes de base
C. Dualité
D. Fonctions booléennes
E. Ecriture d'une fonction sous forme de somme de produits
F. Ecriture d'une fonction sous forme de produit de sommes
G. Passage d'une forme à l'autre
H. Réduction des fonctions booléennes de n variables
   1. *Méthode de Karnaugh*
   2. *Méthode de Tison*
I. Fonctions incomplètes.

## V. Circuits logiques (8h)

A. Notion de circuit logique
B. Circuits combinatoires
   1. *Additionneur*
   2. *Soustracteur*
   3. *Multiplicateur par addition-décallage*
   4. *Multiplexeurs – Démultiplexeurs*
   5. *Décodeurs – Codeurs*
C. Circuits séquentiels
   1. *Bistable R-S*
   2. *Bistables J-K*
   3. *Registres.*

## VI. Mémoires (4h)

A. Généralités et définitions
B. Hiérarchie des mémoires
C. Organisation des informations

D. Mémoire centrale
   1. *Mémoire cache*
   2. *Mémoires auxiliaires.*

**VII. Unité centrale de traitement** (6h)

A. Architecture
B. Unité de commande
C. Registres du CPU
D. Adressage des opérandes
E. Taille de l'adresse et taille de la mémoire
F. Unité arithmétique et logique.

**VIII. Entrées / Sorties** (6h)

A. Evolution
B. Terminaux Interactifs (Moyen d'interaction avec l'écran, Tube cathodique, Ecrans alphanumérique, Ecrans graphiques)
C. Imprimantes (Imprimantes avec impact, Imprimantes sans impact, Traceurs)
D. Scanners
E. Architectures et procédures d'entrée/sortie
   1. *Accès direct à la mémoire (DMA)*
   2. *Canaux d'entrée/sortie*
   3. *Périphériques*

**Références**


1. *Architecture et technologie des ordinateurs* – Paolo Zanella, Yves Ligier, Dunod, Paris, 1998.
2. *Concepts fondamentaux de l'informatique* – Aho A., Ullman J.F. Dunod, Paris, 1993.
3. *Architecture de l'ordinateur* – Tanenbaum A., InterEditions, Paris, 1996.
4. *Le monde Internet, Guide & resources* - Editions O'Reilly International, Thomson, Paris, 1995.
5. *Informatique Générale* - Souheil A. Yactine, Issam Abdul Kader, 1983.

# ALGORITHME NUMERIQUE - I201 (96H)

## ANALYSE NUMERIQUE (48H) :

### Partie I - Analyse matricielle

#### I. Rappels

A. Introduction
B. Notion et classification des erreurs
C. Rappel sur l'analyse vectorielle
   1. *Espace Vectorielle (sur R) : dimension, normes et normes équivalentes.*
   2. *Applications linéaire:isomorphismes, automorphismes et Endomorphisme.*
   3. *Espaces des endomorphismes, relation avec l'espace des matrices carrées et normes matricielles induites.*

#### II. Systèmes linéaires

A. Les méthodes directes.
   1. *Méthode d'élimination de Gauss.*
   2. *Méthode UL*
   3. *Méthode de Choleski*
   4. *Méthode de Jordan : inversion d'une matrice*
B. Les méthodes itératives.
   1. *Introduction:vue générale*
   2. *Méthode de Jordan*
   3. *Méthode de Gauss-Seidel*
   4. *Méthode de relaxation*
   5. *Méthode de gradient*
   6. *Accélération de la convergence*
   7. *Utilisation de MATLAB*

#### III. Valeurs propres

A. Introduction: définition et vue générale
B. Méthode de Jacobi
C. Méthode de Givens Householder
D. Vue sur autres Méthodes

### Partie 2 - Approximation

#### IV. Interpolation

A. Définition
B. Existence et unicité
C. Estimation d'erreur

#### V. Différences finies

A. Opérateurs différences finies
B. Formules de Newton
C. Tables différences finies
D. Différences finies et interpolation
E. Utilisation de MAPLE

#### VI. Intégration et dérivation numériques

A. Intégration numérique

1. *Méthodes avec différences finies.*
2. *Méthodes de Gauss.*

B. Dérivation numérique

**VII. Equations différentielles ordinaires**

A. Introduction: pourquoi résolution numérique
B. Problème théorique et théorème d'existence et d'unicité
C. Méthode d'Euler-Cauchy
D. Formulation générale d'une méthode à un pas
E. Convergence d'une méthode à un pas
F. Ordre d'une méthode à un pas
G. Méthodes de Taylor et Méthode de Runge et Kutta
H. Méthodes à pas multiples: Adams, Moulton et Nistrom

**VIII. Equations aux dérivées partielles**

A. Equations hyperboliques
B. Equations paraboliques
C. Résolution numérique par différences finies.

**IX. Graphique: Définition Numérique des courbes et surfaces paramétriques polynomiales**

A. Fonctions splines
B. Fonctions Bezier

## LOGIQUE (24H) :

**Objectifs :**

- Donner les bases théoriques pour l'étude et la modélisation du raisonnement.
- Se familiariser avec les paradigmes de la programmation logique, étudier ses caractéristiques et connaître ses principes fondamentaux.
- Acquérir une expérience pratique avec la programmation logique (Prolog) en utilisant des bonnes pratiques de programmation : abstraction, généralité, modularité, fiabilité, performance, etc.

**I. La logique propositionnelle**

A. Introduction
B. Vocabulaire
   1. *Proposition*
   2. *Connecteur*
C. Syntaxe
D. Sémantique
   1. *Interprétation*
   2. *Tables des vérités des connecteurs*
   3. *Equivalence*
   4. *Validité et inconsistance*
E. Etude de validité
   1. *Arbre sémantique*
   2. *Réduction*
      a) Algorithme de réduction

UNIVERSITÉ LIBANAISE

b) Approche algébrique
3. *Clauses et formes normales*
   a) Définitions
   b) Algorithme de normalisation
4. *Algorithme de Quine*
5. *Algorithme de Davis et Putnam*
6. *Principe de résolution*
7. *Clauses de Horn*

## II. La logique des prédicats

A. Introduction
B. Vocabulaire
C. Syntaxe
D. Propriétés des variables
E. Sémantique
   1. *Interprétation*
   2. *Validité et inconsistance*
F. Règles d'interférence
   1. *Définition*
   2. *Quelques règles : modus ponens, modus tonens, spécialisation universelle*
   3. *Validité d'un ensemble de règles d'inférence*
   4. *Complétude d'un ensemble de règles d'inférence*
G. Principe de résolution
   1. *Définitions*
   2. *Forme normale prénexe d'une formule*
   3. *Forme clausale d'une formule*
   4. *Application du principe de résolution à des clauses concrètes*
H. Unification
   1. *Définition*
   2. *Substitution et instanciation*
   3. *Notion d'unificateur*
   4. *Algorithme d'unification*
   5. *Application du principe de résolution à des clauses quelconques*

## LE LANGAGE PROLOG (24H) :

# RECHERCHE OPERATIONNELLE - I203
# (38C + 38TD + 20TP)

## *Objectifs*

La recherche opérationnelle englobe un ensemble de méthodes d'analyses scientifiques des problèmes posés, méthodes particulièrement tournées vers la recherche de la meilleure façon d'appréhender les faits et de prendre des décisions permettant de déboucher aux meilleurs résultats.
La recherche opérationnelle intègre les données réelles de l'entreprise, les outils mathématiques et l'informatique.
La recherche opérationnelle permet de modéliser des situations concrètes et réelles de types combinatoires où il y a des difficultés d'énumérer toutes les solutions possibles à partir d'un certain seuil et sous certaines conditions, par exemple quand il s'agit de prendre des décisions dans l'incertain ou de faire face à diverses stratégies de l'adversaire.
On peut citer quelques exemples d'applications dans plusieurs disciplines:

- Les problèmes de productions dans l'entreprise
- Les problèmes d'affectations des ressources
- Les problèmes des flots dans les réseaux de transports
- Les problèmes linéaires d'optimisation des coûts et des profits
- Les problèmes des cheminements dans les graphes etc.

L'informatique contribue fortement à la progression et au développement de ces outils en permettant de traiter d'importantes masses de données à l'aide des algorithmes complexes.

## Partie I – Recherche opérationnelle

### X. Généralités sur la programmation linéaire

A. La programmation linéaire
B. Historique
C. Domaines d'applications
D. Exemples - Formulation d'un modèle de programmation linéaire
E. Résolution graphique d'un problème pour préciser les propriétés importantes de la programmation linéaire: connexité du domaine, programmes intéressantes, dégénérescence, Interprétation des variables d'écarts, stabilité de l'optimum.

### XI. Méthode et critère de DANTZIG

A. Introduction et rappels
B. Algorithme Matricielle du Simplexe
C. Cas particulières
D. Applications
E. Tableau de Charne-Cooper-Henderson : Tableau du simplexe - Algorithme
   1. *Cas d'inégalités- Exemple traité*
   2. *Autre cas- Exemple traité*
   3. *Interprétation des quantités du tableau*

### XII. Paramétrage des coûts

A. Intérêt
B. Méthode
C. Exemple traité- Discussion d'après les valeurs du paramètre

## XIII. La Dualité

A. Le problème Dual d'un primal particulier
B. Tableau des coefficients isolés de Tucker
C. Théorème de Dualité
D. Dual d'un système primal général. Exemple traité (Résolution de deux programmes duaux- intérêt).
E. Interprétation économique de la Dualité
F. Paramétrage des seconds membres - Intérêt
G. Exemple traité

## XIV. Théorie des jeux

A. Introduction: jeu sur un tableau rectangulaire
B. Choix d'un critère : critère de Neumann, critère de Laplace, critère de Hurwcz, critère de Savage
C. Point d'équilibre- théorème fondamental Exemple d'application
D. Stratégies pures et stratégies mixtes- propositions de Von Neumann - Remarques
E. Propriétés diverses: Invariance de solutions optimales, Réduction par domination, Notion d'équilibre
F. Algorithme de réduction de l'ordre de la matrice du jeu- Exemple d'application
G. Forme développé- Exemples

## XV. Programmation linéaire en nombres entiers

A. Méthode Booléenne ou les variables sont binaires- Formulation et transformation du système en équation booléenne unique- Application
B. Méthode de résolution- Représentation Matricielle - Application
C. Méthode Booléenne applicable aux programmes linéaire en nombre entiers- Inconvénient
D. Méthode de résolution d'un programme linéaire en nombre entiers
E. Méthode Arborescente- Exemple d'application

# Partie II – Théorie des graphes

## XVI. Définitions et concepts de base

A. Concepts orientés et mon orientés
B. Successeurs et prédécesseur- Degré
C. Sous graphe, graphe partiel, opérations
D. Graphes particuliers
E. Chemins et chaînes - circuit et cycle
F. Matrices associées à un graphe

## XVII. Nombre cyclomatique et cocyclomatique

A. Graphe convexe - composantes connexes d'un graphe
B. Nombre cyclomatique: cycles et cocycles
C. Ensemble de connectants minimal: propriétés caractéristique d'un cycle élémentaire et d'un cocycle élémentaire
D. Lemme de Minuty
E. Base des cycles et cocycles - structure algébrique de l'ensemble $\Phi$ des cycles et de l'ensemble H des cocycles
$IR^m = \Phi + H$, exemple d'application
F. Arbre- théorème caractéristique
G. Caractérisation d'un graphe connexe - Algorithme de construction des

bases des cycles

## XVIII. Graphe quasi fortement connexe, semi-fortement connexe et fortement connexe

A. Définitions et exemples
B. Propriétés caractéristiques d'un graphe quasi fortement connexe
C. Composantes fortement connexes - Algorithmes: principe de cheminement concentrique ou en profondeur directe
D. Algorithmes de constructions des composantes connexes et des composantes fortement connexes
E. Graphe quotient - propriétés de cheminement dans le graphe réduit caractérisation pour que Gr soit quasi fortement connexe
F. Caractérisation des graphes fortement connexes
G. Arborescence - théorème caractéristique Algorithme de construction des circuits dans un graphe fortement connexe.

## XIX. Problèmes de flots

A. Définitions- Exemples- Relations avec les cycles, programme linéaire associé
B. Problème du flot maximum- Algorithme de Ford et Fulkerson - Amélioration du vecteur flot - Théorème de Ford- Fulkerson
C. Structure algébrique dans l'ensemble des flots- théorème caractéristique pour qu'un vecteur Ø soit un flot
D. Problème du flot compatible- condition d'existence d'un vecteur flot. Algorithmes pour les problèmes du flot compatible et théorème du flot compatible - Application

## XX. Problème de tension

A. Définition- Relation avec les cocycles programme linéaire associé - Domaines d'applications.
B. Théorèmes caractéristiques pour qu'un vecteur Ø soit une tension
C. Orthogonalités de l'ensemble des flots et des tensions

## XXI. Problèmes de cheminement: algorithmes de recherche de plus courts chemin

A. Plus court chemin dans un réseau. Position du problème
B. Plus court chemins et plus courtes distances
C. Algorithme de Bell (réseau sans circuit)
D. Algorithme de Dijkstra (longueurs positives ou nulles)
E. Algorithme général
F. Algorithme de Dantzig. Applications

## XXII. Problème de transport

A. Définitions et formulation analytique : programme linéaire- Interprétation en terme de théorie des graphes
B. Algorithmes permettant d'obtenir une solution de bases (méthode de la différence maximale, méthode du coin Nord Ouest, méthode du minimum en ligne puis en colonne) -applications
C. Dégénérescence- passage d'une solution dégénérée à une solution de base- calcul des coûts marginaux
D. Optimisation : tension sur les arcs du graphe G = (x,u) chaînes de substitutions méthode d'optimisation- Applications
E. Problème d'affectation : formulation, interprétation en terme théorie des graphes, méthode Hongroise Applications

## XXIII. Problème d'ordonnancement

A. Introduction
B. Les contraintes potentielles, disjonctives, cumulatives
C. Historique
D. Construction d'un réseau PERT- formulation en terme de programmation linéaire
E. Recherche de niveaux dans un graphe sans circuits - fonction ordinale
F. Dates au plus tôt - dates au plus tard d'un événement tâches critiques - chemins critiques, intervalle de flottement, marge total, marge libre, marge certain
G. Méthode de potentielle, Applications

# ALGORITHMIQUE & STRUCTURES DE DONNEES – I205 (72H)

## *Objectifs*

L'objectif de ce module est de former des étudiants capables de développer des applications dont la mise en œuvre complète requiert une réelle compétence dans le domaine de la programmation structurée.
Nos objectifs se résument ainsi :

- Spécifier, concevoir et développer des modules implémentant les structures fondamentales de données.
- Etre en mesure de choisir la structure de données la plus adéquate pour représenter un problème donné et permettre de le résoudre en mettant en œuvre des traitements efficaces et optimaux.

**I. Rappels** **(2hC+6hTD)**

A. Pointeurs
B. Tableaux
C. Fonctions

**II. La récursivité** **(2hC+2hTD+2hTP)**

A. Exemples classiques (factorielle, Fibonacci, ...)
B. Intérêts
C. Application
   1. *Les tours de Hanoi*

**III. Complexité des algorithmes** **(2hC+1hTD)**

A. Méthodes d'évaluation des algorithmes
   1. *Mesure de la complexité en temps*
   2. *Mesure de la complexité en espace mémoire*
   3. *Autres critères d'évaluation*
B. Comparaison de deux algorithmes

**IV. Les algorithmes de tri** **(2hC+2hTP)**

A. Tri par sélection
B. Tri par insertion

**V. Concept de type abstrait** **(6hC)**

A. Importance du choix de la structure de données dans le processus de développement du logiciel
   1. *Exemple : le jeu de la vie*
      a) Représentation matricielle
      b) Représentation sous forme de deux listes (cellules nées et cellules mortes)
B. Définition
C. Encapsulation
D. Signature
E. Description des propriétés d'un type abstrait
F. Exemple : le type ensemble

## VI. Le type abstrait liste linéaire (2hC+4hTD+4hTP)

A. Définition
B. Opérations
C. Implémentation
   1. *Liste simplement chaînée*
   2. *Liste ordonnée*
   3. *Liste doublement chaînée*
   4. *Liste circulaire*
D. Applications
   1. *Les polynômes*
   2. *Les polygones*

## VII. Le type abstrait File (2hC+2hTD+4hTP)

A. Définition
B. Opérations
C. Implémentation contiguë (statique)
D. Implémentation chaînée (dynamique)
E. Application
   1. *Simulation d'une file d'attente*

## VIII. Le type abstrait Pile (2hC+2hTD+4hTP)

A. Définition
B. Opérations
C. Implémentation contiguë
D. Implémentation chaînée
E. Applications
   1. *Analyse syntaxique d'une expression infixée*
   2. *Evaluation d'une expression polonaise inversée*

## IX. Les structures arborescentes (2hC+4hTD+6hTP)

A. Les arbres binaires
   1. *Définition*
   2. *Propriétés fondamentales*
   3. *Représentation*
   4. *Parcours : infixé, préfixé, postfixé*
B. Les arbres binaires de recherche
   1. *Définition*
   2. *Recherche d'un élément*
   3. *Adjonction d'un élément*
   4. *Suppression d'un élément*
   5. *Analyse de la complexité de la recherche d'un élément*
   6. *Applications*
      a) Tri rapide
      b) Tri par tas
C. Introduction aux autres types d'arbres
   1. *Arbres équilibrés*
   2. *Arbres planaires généraux*

## X. Les méthodes de hachage (2hC+2hTD+4hTP)

A. Principes du hachage

B. Fonctions de hachage
C. Résolution de collision
D. Applications

# BASES DE DONNEES - I211 (72C+18TD +48TP)

## *Objectifs*

Ce cours forme les étudiants à la conception et à la mise en place et usage de bases de données classiques, de type relationnel. Il donne la connaissance nécessaire pour :

- La théorie et la pratique de systèmes de gestion de base de données
  - Exprimer le besoin en information d'applications avec un formalisme simple et rigoureux,
  - Concevoir une base de données avec la démarche d'ingénieur,
- Expérience pratique avec un SGBD commercial majeur, et
  - Implanter une base de données sur un système de gestion de base de données classique
- Utiliser les bases de données à travers des langages de manipulation offerts par un SGBD classique .
- Elaborer une application de base de données de la conception à la mise en oeuvre.

## I. INTRODUCTION : LES CONCEPTS DE BASE [6h]

A. Etude du cas: l'introduction :
   1. *Monde réel*
   2. *Modèle de données (base de données)*
   3. *Historique : Système de gestion des fichiers, Modèles : Hiérarchique, réseaux, relationnel et orienté objet.*
   4. *Langage de requête (interface utilisateur): Langage de programmation haut niveau*

B. Définitions
   1. *Base de données, système de gestion de base de données, Les différentes couches*
   2. *Modèle des Données, stockage de données, accès aux données*

C. Conception des données: étapes principales et stratégie
   1. *Diagramme entités association*
   2. *Processus de normalisation*

TP (5PTS): système de Gestion des fichiers (C/C++/Java)

D. Conception de base de données (diagramme Entités association) [6h]
   1. *Entité: attribue, ensemble d'entités (instances), etc.*
   2. *Association: attribue, cardinalité, ensembles d'association (instances), etc.,*
   3. *Diagramme E/A, validation du diagramme E/A*

E. Modèle relationnel [6h]

*Définitions [4h]*
   1. *Définition de base des données relationnelle*
   2. *Relation (Table): attribut, domaine, clef,*
   3. *Contrainte d'Intégrité: contraintes de clef, clef étrangère, de domaine, etc.*

*Elaboration du schéma de base des données relationnelle [10h]*
   4. *Diagramme E/A vers le Modèle relationnel ; Règles et démarche*
   5. *Processus de normalisation (formes normales): 1NF, 2NF, 3NF, BCNF, etc.*

*Langages [22h]*
   1. *Langages Formels [7h]*
      a) Algèbre Relationneile
      b) Calcul Relationnel
   2. *Langage de définition de donnée (SQL: LDD) [3h]*
      a) Création de base de données, tables

b) Implémenter les Contraintes d'Intégrités

3. *Langages de requête [12h]*
   a) SQL: LMD
      select-from-where (group by, order by, having) sous-requête, vue, curseur, trigger, procédure stockée, Embedded SQL, etc.
   b) [4h] autres Langages: 2D QBE langage, interface utilisateur (VB via ODBC/ADO, ...)

F. Introduction aux thèmes avancés [6h]

1. *Architecture de SGBD*
2. *Stockage de donnée et Indexation*
   a) Stocker les données: Disques et fichiers
   b) Indexes, création et utilisation d'index en SQL
3. *Gestion des sessions*
4. *Evaluation de requête*
   a) Optimisation (arbre de requête)
   b) Introduction: Evaluation des opérateurs relationnels
5. *Gestion des transactions*
   a) Introduction (création et usage de transaction en SQL)
   b) Concurrence, Control et gestion de crash

G. TPs (20 PTS); MINI PROJET

H. TP : démo [6h], TPs [18h]

**MATERIELs & REFERENCES:**

**Software:** AMC*Designor, MSSQLSERVER (7/2000) et Oracle 7/8 desktop editions.

**Livres:** Database Management Systems 2nd edition, R. Ramakriham/j.Gehrke, ISBN:0-07-232206-3, McGraw-Hill, http://www.mhhe.com.

Fundamentals of database systems 3rd edition, ELMASRI / NAVATHE, ISBN:0-201-54263-3,ADDISION-WESLEY, http://www.awlonline.com.

**Pré requis :**

**Rien**

**Préparation pour Bases de données Avancées**

# PROGRAMMATION AVANCEE (ORIENTEE OBJET) - I209 (60H)

## *Objectifs*

L'objectif de ce module est de former des élèves capables de développer des logiciels modernes et conviviaux. Actuellement, les principes et les mécanismes de la conception et la programmation orientée objet semblent être la technique la plus en adéquation avec les besoins de ce type de logiciels. En effet, ces logiciels sont, en général, d'une énorme complexité ; l'orientation objet facilite leur analyse et leur programmation car le modèle objet permet une modélisation directe des objets du monde réel, le suivi de leur cycle de vie depuis leur création à leur disparition et la création de modèle plus compact grâce aux facilités de réutilisation. La programmation est également facilitée car l'orientation objet permet la génération de squelettes de modules applicatifs à partir du modèle graphique résultant de l'analyse, la réutilisation de composants existants ainsi que le prototypage rapide permettant de visualiser des scénarios de fonctionnement (Utilisation d'outils tels que Rational Rose). Ainsi nous fixons-nous les objectifs suivants :

- Maîtriser les bases méthodologiques essentielles de la conception orientée objet résultant des derniers développements et en particulier celles concernant la méthode UML.
- Maîtriser les concepts de la programmation objet.
- Maîtriser un langage de programmation orienté objet.
- Etre en mesure de résoudre des problèmes complexes à partir de spécifications données en concevant et en élaborant des solutions orientées objet.

## Partie I : Initiation à la programmation objet

### I. Introduction (↓)

A. Origine de la programmation objet
B. Pourquoi la programmation objet ?
C. Les langages à objets

### II. Les concepts (7hC)

A. Les classes
   1. *Attributs*
   2. *Méthodes : l'interface*
B. L'instanciation
C. L'envoi de messages
D. L'héritage
   1. *Techniques de spécialisation*
   2. *Héritage et sous-typage*
   3. *Interprétations de l'héritage*
   4. *Héritage multiple et conflits d'héritage*
   5. *Polymorphisme*
E. Définition de classe par composition

### III. La méthode UML (Unified Modeling Language) (8hC+10TD)

A. Le modèle fonctionnel
   1. *Identification des acteurs*
   2. *Identification des cas d'utilisation*
   3. *Réalisation du diagramme de cas d'utilisation*
B. Le modèle statique

1. *Diagrammes de classes*
   a) Classes
   b) Interfaces de classe
   c) Classes paramétrables
   d) Associations
   e) Contraintes
   f) Agrégations
   g) Généralisation.
   h) Classes association

## Partie II : Application : le langage C++

### IV. Les apports de C++ (2hC)

A. Les déclarations
B. Le type référence
C. Les types union, struct et enum
D. Les conversions de type
E. Les fonctions
F. Edition de liens
G. Allocation mémoire
H. Les entrées / sorties

### V. Définition d'une classe simple (2hC+4hTD+5hTP)

A. Syntaxe
   1. *Spécification d'une classe*
   2. *Définition des membres*
   3. *Le mot clé this*
   4. *Variables et méthodes de classes*
   5. *Opérateur de résolution de portée*
B. Construction et destruction d'objets
   1. *Composition d'objet*
   2. *Construction par copie*
C. Instanciation
D. Opérateurs
   1. *Opérateurs arithmétiques et logiques*
   2. *Affectation*
   3. *Opérateur d'indice*
   4. *Conversions de type*
   5. *Opérateur d'appel fonctionnel*
   6. *Déréférenciation*
   7. *Allocation mémoire*

### VI. La généricité (1hC+1hTD+2hTP)

A. Fonctions génériques
B. Classes génériques

### VII. Les exceptions (1hC+1hTD+2hTP)

A. Levée d'une exception
B. Le bloc try-catch
C. Spécification de fonction
D. Exceptions et constructeurs
E. Exceptions non traitées et inattendues

### VIII. L'héritage (4hC+4hTD+6hTP)

A. Introduction

1. *Héritage de spécialisation*
2. *Héritage d'implantation*

B. Les classes dérivées
1. *Interface d'une classe dérivée*
2. *Utilisations de la dérivation*
3. *Constructeurs et destructeurs*
4. *Conflits d'héritage*

C. Edition de lien dynamique
1. *Méthodes virtuelles*
2. *Constructeurs et destructeurs virtuels*
3. *Méthodes virtuelles et extensibilité*

# ARCHITECTURE LOGICIELLE DES SYSTEMES INFORMATIQUES I - I213 (96H)

## I. Description fonctionnelle d'un ordinateur séquentiel (Von Neumann) (4h)

A. Schéma général d'un ordinateur
   1. *La mémoire*
      a) La cellule
      b) La notion d'adresse
   2. *Le bus*
      a) Le bus d'adresses
      b) Le bus de données
   3. *L'unité centrale*
      a) L'unité arithmétique et logique
      b) L'unité de commande
      c) Les registres
         (1) Registre d'instructions
         (2) Compteur ordinal
         (3) Pointeur de pile
         (4) Accumulateurs
         (5) Registre d'index
         (6) Frame pointer
         (7) Registres d'état
   4. *Les périphériques d'entrée/sortie*
   5. *Les mémoires secondaires*

B. Le cycle d'exécution d'une instruction
   1. *Etapes d'un cycle*
      a) Recherche d'instruction
      b) Décodage
      c) Exécution
   2. *Exemple d'exécution*

## II. La couche Assembleur (14h)

A. Le langage d'assemblage : jeu d'instructions
   1. *Le format d'instruction*
      a) Code opération
      b) Mode d'adressage
      c) Opérandes
   2. *Les structures de contrôle*
      a) Choix
      b) Boucle

B. Assemblage
   1. *Définition*
   2. *Principe*
   3. *L'assembleur*
      a) Table des symboles
      b) Génération de code
   4. *Edition de liens*
      a) Définition
      b) Intérêt
      c) L'éditeur de liens
      d) Méthodes d'édition
         (1) Statique
         (2) Dynamique

5. *Chargement*

C. Les piles et les sous-programmes
   1. *Gestion des appels*
   2. *Gestion des retours*
   3. *Passage des paramètres*
   4. *Variables locales*
   5. *Fonctions récursives*

## III. Les entrées/sorties (14h)

A. Introduction
   1. *Organisation*
   2. *Périphériques d'entrée/sortie*
   3. *Contrôleurs d'entrée/sortie*

B. Les ports d'entrée/sortie
   1. *Définition*
   2. *Intérêt*

C. Le clavier
   1. *Tampon d'entrée*
   2. *Mécanisme de lecture : mode programmé*
      a) Registre d'état
      b) Boucle de test

D. L'écran
   1. *Pixel*
   2. *Mode alphanumérique et mode graphique*
   3. *Mécanisme d'affichage*

E. L'entrée avec écho

F. Les interruptions
   1. *Concept d'interruption*
   2. *Notion de priorité*
   3. *Masquage et désarmement d'interruption*

G. L'accès direct à la mémoire

## IV. La couche système (4h)

A. Généralités sur les systèmes d'exploitation
   1. *La machine physique (Rappels)*
      a) Description
      b) Mode d'exploitation
         (1) Langage machine
      c) Limites et inconvénients
   2. *Fonctionnalités*
      a) Gestion des ressources
         (1) Gestion de l'unité centrale
         (2) Gestion de la mémoire centrale
         (3) Gestion de la mémoire secondaire
      b) Machine virtuelle

B. Evolution des systèmes d'exploitation
   1. *Systèmes monoprogrammés*
   2. *Systèmes multiprogrammés*
   3. *Le temps partagé*
   4. *Systèmes distribués*

## V. Les éléments de base (6h)

A. Processus
   1. *Définition*
   2. *Etat d'un processus*

3. *Représentation interne*
   a) Bloc de contrôle (descripteur de processus)
   b) Exemples : Unix,...
4. *Opérations sur les processus*
   a) Créer
   b) Détruire
   c) Mettre en attente
   d) Réveiller
   e) Suspendre
   f) Reprendre
5. *Graphe d'états*

B. Ressources
1. *Définition*
2. *Classes de ressources*
   a) Définition
   b) Opérations communes
      (1) Demander / allouer
      (2) Utiliser
      (3) Libérer
3. *File d'attente de blocs de contrôle de processus*

## VI. La gestion de l'unité centrale (8h)

A. Commutation de mot d'état
1. *Notion de mot d'état (PSW)*
2. *Commutation*

B. Interruptions
1. *Types*
   a) Externe
   b) Déroutement
   c) Appel au système
2. *Priorité*
3. *Masquage et désarmement d'interruption*

C. Entrée / sortie

D. Ordonnancement
1. *Files d'attente*
2. *Types d'ordonnanceurs*
3. *Algorithmes d'ordonnancement*

## VII. Généralités sur la mémoire (6h)

A. Type de mémoires
1. *Mémoire volatile*
   a) Mémoire cache : temps d'accès, taille, coût
   b) Mémoire centrale (RAM) : temps d'accès, taille, coût
2. *Mémoire permanente*
   a) Mémoires secondaires : variantes, temps d'accès, taille, coût

B. Hiérarchie de mémoires

C. Gestion de la mémoire
1. *Problèmes*
   a) L'organisation
      (1) Découpage en zones : taille fixe ou variable
   b) L'allocation
   c) La libération
2. *Méthodes d'allocation*
   a) Le modèle d'allocation contiguë
      (1) Principe
      (2) Inconvénients

(i) *Fragmentation interne*
(ii) *Fragmentation externe*
(3) La fusion et le compactage
(i) *Principe*
(ii) *Inconvénients*
(4) Stratégies de placement
(i) *Première zone libre*
(ii) *Meilleur ajustement*
(iii) *Plus grand résidu*
(iv) *Comparaison : critères d'appréciation*

b) Le modèle d'allocation non contiguë
(1) Principe
(2) Avantages
(3) Gestion de liens
(i) *Le chaînage*
(ii) *L'indexation*

## VIII. Gestion de la mémoire centrale (14h)

A. Allocation contiguë en mémoire centrale
1. *Mécanismes de traduction et de liaison*
a) Traduction
b) Liaison
2. *Système monoprogrammé*
a) Système mono-utilisateur
(1) Organisation
(2) Gestion
(3) Mémoire virtuelle
(i) *Recouvrement*
(ii) *Chargement dynamique*
b) Système à temps partagé
(1) Echanges : swap in, swap out
3. *Système multiprogrammé*
a) Partition fixe
(1) Organisation
(2) Gestion et protection des zones
b) Partition variable
(1) Organisation
(2) Gestion et protection des zones

B. Allocation non contiguë de la mémoire centrale
1. *Généralités*
a) Principe : systèmes paginés / segmentés
b) Mécanisme : conversion d'adresse
c) Mémoire virtuelle
2. *Systèmes paginés*
a) Conversion d'adresse
b) Description de la table de pages
(1) Table en mémoire centrale
(i) *Inconvénients*
(2) Table partielle en mémoire associative
(i) *Gestion des tables : principes de localité*
c) Partage et protection des pages
3. *Systèmes segmentés*
a) Conversion d'adresse
b) Description de la table de segments
c) Partage et protection des segments
4. *Principe des systèmes combinés*
5. *Gestion des pages dans les systèmes monoprogrammés*
a) Méthode d'allocation
(1) Stratégie de chargement

(2) Stratégie de placement
(3) Stratégie de remplacement

6. *Gestion des pages dans les systèmes multiprogrammés*

## IX. Introduction à la programmation sous Unix (24hTP)

A. Les processus
   1. *Création*
   2. *Destruction*

B. Les tubes de communication

C. Les signaux

UNIVERSITE LIBANAISE

## X. La couche Physique
- A. Introduction à la transmission de données
- B. Les supports de transmission

## XI. La couche liaison de données
- A. Les services de la couche liaison de données
- B. La notion de la trame
- C. Détection et correction d'erreurs
- D. Polynôme générateur
- E. Conception de protocoles : Protocoles à fenêtres d'anticipation
- F. Spécification et vérification de protocoles
  1. *Automates finis*
  2. *Réseaux de petri*
- G. Exemples de protocoles
  1. *PPP*
  2. *HDLC*

## XII. Contrôle d'accès au canal
- A. Les protocoles d'accès
- B. Les LAN
  1. *Les réseaux Ethernet – IEEE 802.3*
  2. *Bus à jeton - IEEE 802.4*
  3. *Anneau à jeton - IEEE 802.5*

## XIII. Exemples de protocoles La couche réseau
- A. Les services de la couche réseau
- B. Algorithmes de routage
- C. Contrôle de congestion
- D. Interconnexion de réseau
- E. Exemples de protocoles
  1. *La couche IP*

## XIV. La couche transport
- A. Les services de la couche transport
- B. L'adressage
- C. Gestion de connexion
  1. *Etablissement d'une connexion*
  2. *Libération d'une connexion*
- D. Contrôle de flux
- E. Reprise après incident
- F. Exemples de protocoles (UDP, TCP)

## XV. La couche application
- A. La sécurité dans les réseaux
- B. DNS (Domain Name System)
- C. SNMP
- D. SMTP
- E. WWW

# PROGRAMMATION WEB - I244

## *Objectif*

Initier les étudiants aux principes, méthodes et langages de programmation utilisés dans le World Wide Web.

### I. Introduction au World Wide Web

A. Histoire du World Wide Web
B. HTML - HyperText Markup Language
C. SGML - Standard Generalize Markup Language
D. HTML Avancée
E. URI - Universal Resource Identifier

### II. Interaction Client - serveur - HTTP, CGI

A. HTTP - HyperText Transfer Protocol
B. HTTP/1.0
C. HTTP/1.1
D. HTTP-NG
E. Secure HTTP
F. CGI - Common Gateway Interface
G. Créer des applications CGI

### III. Programmation côté client

A. JavaScript
B. Utilisation de JavaScript avec des formulaires
C. VBScript

### IV. Dynamic HTML

A. Cascading Style Sheets (CSS)
B. Cascading Style Sheets, niveau 2
C. Feuilles de style pour la lecture

### V. Programmation côté serveur ASP, PERL

A. PERL - Practical Extraction Report Language
B. Variables d'environnement CGI
C. Formulaires HTML et CGI
D. Complément sur les formulaires HTML
E. Manipulation de fichiers Perle
F. Variables Spéciales Perle
G. SSI - server-side includes
H. Traitement des chaînes de caractères et expressions régulières en Perle

### VI. Programmation côté serveur client - Java

A. Java
B. Applets et Applications
C. Le langage Java
D. Applets
E. Evénements Java

F. Threads Java, animations
G. Complément sur les applets

## VII. Extensible Markup Language (XML)

A. Extensible Markup Language (XML) – L'après HTML
B. XML – C'est quoi?
C. Extensible HTML (XHTML)
D. Applications XML
E. Extensible Style Language (XSL)
F. XML et Java

# MULTIMEDIA ET NOUVELLES TECHNOLOGIES
## I236 (60H)

I. **Introduction :** multimédia et hypertexte, normalisation, internet

II. **Son :** caractéristiques, analyse en fréquences, transformée de Fourier, synthèse musicale Transaction management

III. **La numérisation :** *signaux analogiques ou numériques, processus de numérisation filtrage, échantillonnage, qualification*

IV. **Image :** système visuel humain, espaces de couleurs type d'images, analyse élémentaire d'images

V. **Signal Vidéo analogique et numérique**

VI. **Traitement d'images :** filtrage, détection des contours, morphologie mathématique, restauration, recalage, reconnaissance de formes

VII. **Eléments de la théorie de l'information :** principe de la compression, mesure de l'information, notion d'entropie, codage

VIII. **La compression :** compression sans ou avec perte, compression de données textuelles, compression audio, compression image, compression video, algorithmes de compression

IX. **Formats et conversion, JPEG, MPEG,....**

X. **Application et logiciels multimédias, langages de programmation et de synchronisation**

XI. **Réseaux et multimédia**

XII. **Matériel informatique multimédia**

UNIVERS...

# PROGRAMMATION RESEAUX I260

## I. Introduction

A. Rappel sur le modèle TCP/IP
B. La notion de port.
C. Exemple d'une application : le protocole HTTP

## II. Introduction à JAVA

D. Les éléments de bases du langage
E. Les entrés/sorties en JAVA
F. Introduction au thread en JAVA

## III. les Sockets

G. Introduction à la communication entre processus (Inter-Process Communication IPC)
H. Rappels sur les protocoles TCP/UDP
I. Programmation sockets avec TCP
J. Programmation sockets avec UDP
K. Des applications utilisant les sockets (chat, un simple serveur http, etc.)

## IV. Middleware

L. Définition
M. Modèles d'architectures client serveur (deux tiers, trois tiers, N tiers)
N. Evolution des middlewares
O. Communication synchrone/asynchrone
P. Passage de paramètres

## V. JAVA RMI : Remote Method Invocation

Q. Les objets distribués
R. Les applications clients/serveur en RMI
S. Le service annuaire: RMI Registry
T. La communication asynchrone : Callback

## VI. CORBA Common Object Broker Architecture

U. L'architecture CORBA
V. Les services CORBA
W. Le langage IDL (Interface Definition Language)
X. Projection du langage IDL dans des autres langages de programmation.
Y. Les applications Client/serveur en CORBA

## VII. Les Services web

A. Introduction
B. Le protocole SOAP (Simple Object Access Protocol)

C. Le langage de description des services web (WSDL : Web Service Description Language).
D. UDDI : Universal Description, Discovery, and Integration (UDDI)

# THEORIE DE LANGAGE ET COMPILATION - I214
# (25C + 25TD + 12TP)

## *Objectifs*

L'écriture des compilateurs est passée du stade artisanal au stade industriel. Les techniques relatives à l'analyse lexicale, syntaxique et à l'optimisation du code se sont de plus en plus affinées.
En parallèle à cette évaluation les besoins de langages plus restreints, mieux adaptés à l'utilisateur et à ses problèmes se sont imposés: langages graphiques, langages orientés objets, langage de simulations, machines abstraites etc.... ces besoins nécessitent souvent l'écriture de compilateurs et des interpréteurs adaptés. L'objectif est de présenter les principes, les techniques et outils de la compilation et en particulier:

- Analyse lexicale
- Analyse syntaxique
- Traduction dirigée par la syntaxe
- Contrôle de type
- Production de code intermédiaire
- Production de code objet et optimisation

L'étudiant peut profiter des idées et techniques développées dans ce cours pour la conception d'autres logiciels d'usage général par exemple: les grammaires non contextuelles et des définitions dirigées par la syntaxe peuvent être utilisées pour implanter de nombreux " petits langages ", parmi lesquels des systèmes de compositions de textes et de dessin des figures. L'accent est mis sur la résolution de problèmes rencontrés au cours de la conception de tout traducteur quel que soit le langage source et la machine cible.

## I. Généralités et concepts de bases

A. Qu'est ce qu'un compilateur
B. Qu'est ce qu'un programme objet
C. Environnement du compilateur
D. Analyse du programme source: Analyse lexicale, Analyse syntaxique et Analyse sémantique.
E. Rôle de l'analyseur lexical-Attributs des unités lexicales
F. Table des symboles: définition, Interface et implantation d'une table des symboles.
G. Phases d'analyse
H. Partie frontale et partie finale d'un compilateur

## II. Automate d'états finis- conception d'un générateur d'analyseurs lexicaux

A. Définition et représentation d'un automate d'état finis-Exemples
B. Construction d'un automate fini non-déterministe (AFN) à partir d'une expression régulière-algorithme de Thompson
C. Automate fini non-déterministe avec ε- mouvement, définition et propriétés
D. Algorithme de construction d'un automate déterministe DFA à

partir d'un automate NFA- Application

E. Minimisation du nombre d'états d'un automate fini déterministe DFA- Algorithme et application
F. Conception d'un générateur d'analyseurs lexicaux et réalisation d'un analyseur pour un langage évolué donné
G. Méthode d'implantation efficace d'un analyseur lexical

**III. Analyse syntaxique**

A. Introduction et rôle de l'analyseur syntaxique
B. Portée des identificateurs, traitement des erreurs syntaxiques, contrôle de types et de contexte
C. Analyse prédictive non récursive: calcul PREMIER (X) et SUIVANT (A). Analyseur syntaxique prédictif- Exemples
D. Analyseur LR et Algorithme d'Earley: construction des ensembles d'items-Applications
E. Table d'analyse SLR- Algorithme - Applications
F. Construction des tables canonique d'analyse LR - Algorithme
G. Table d'analyse LALR- Algorithme de commande d'un analyseur syntaxique LR
H. Récupération sur erreur en analyse LR - Applications

**IV. Système d'écriture de traducteurs-traduction dirigée par la syntaxe**

A. Grammaire attribuée: attributs hérités et synthétisés. Définition du domaine et règle sémantique. Exemple (attributs hérités et synthétisés)
B. Graphe de dépendance – exemples - Arbre Abstrait. Définition dirigée par la syntaxe pour construire des arbres abstraits. Graphes orientés acycliques pour les expressions. Applications
C. Ordre d'évolution en profondeur pour les attributs d'un arbre syntaxique-Schémas de traduction et exemple-traduction descendante. Application, Algorithme d'un traducteur prédictif dirigé par la syntaxe. Applications
D. Construction d'un arbre syntaxique par descente récursive

**V. Production de code intermédiaire**

A. Introduction et représentations d'une expression.
B. Code à trois adresses: différentes sortes de code à trois adresses. Implantation d'instructions à trois adresses
C. Traduction dirigée par la syntaxe pour la production de code à trois adresses - Instruction de flot de contrôle -Exemples- adressage des éléments de tableaux exemples.
D. Expression booléenne. Applications
E. Traduction de l'instruction de choix multiple. Appels de procédures

**VI. Production de code**

A. Introduction: problèmes posés et machine cible
B. Gestion de la mémoire à l'exécution
C. Blocs de base et graphes de flot de contrôle. Algorithme de partition d'un programme écrit sous forme d'instructions de

UNIVERSITÉ LIBANAISE

code à trois adresses en blocs de base. Applications

D. Informations sur les utilisations ultérieures
E. Un générateur de code simple
F. Allocation et assignation les registres. Exemples
G. Représentation des blocs de base sous forme d'un graphe orienté

Acyclique (DAG).Algorithme de construction d'un DAG. Applications-production de code à partir des DAG.Exemples

H. Construction des générateurs de code

## VII.Optimisation de code

A. Introduction
B. Optimisation des blocs de base
C. Boucles dans le graphe de flot de contrôle
D. Transformation pour améliorer le code
E. Analyse de flot de données dans les graphes structurés
F. Algorithmes de flot de données efficaces

# Architecture logicielle des systèmes informatiques II(1215)

## I. La synchronisation de processus (26h)

A. Introduction
   1. *Exemple simple*
   2. *Instruction atomique (indivisible)*

B. Problème de la section critique
   1. *Définition du problème*
   2. *Exclusion mutuelle*
   3. *Notion de Protocole*
      a) Prologue (section d'entrée)
      b) Epilogue (section de sortie)
   4. *Solutions matérielles*
      a) Désarmement des interruptions
      b) Instruction test and set (TAS)
   5. *Solutions logicielles : les algorithmes parallèles*

C. Communication entre processus et problèmes de synchronisation
   1. *Les sémaphores*
      a) Définition
      b) Synchronisation
      c) Inconvénients
   2. *Les moniteurs de Hoare*
      a) Définition
      b) Variable de type condition
      c) Synchronisation
   3. *Modèles classiques de synchronisation*
      a) Modèle de producteur / consommateur
      b) Modèle de lecteur / rédacteur

## II. Gestion de la mémoire secondaire : le système de gestion des fichiers (16h)

A. Structure d'un SGF (modèle Unix)
   1. *Bootstrap, Super bloc, zone inodes, zone des données*

B. Gestion d'un espace disque
   1. *Gestion de l'espace libre*
   2. *Allocation / libération de zones (blocs)*

C. Structure d'un fichier régulier : descripteur de fichier (inode)

D. Répertoires

E. Opérations sur les fichiers
   1. *Création*
   2. *Ouverture*
   3. *Lecture*
   4. *Ecriture*
   5. *Fermeture*

## III. Programmation sous Unix (18hTP)

A. Les sémaphores

## B. Les sockets

# SYNTHESE D'IMAGE - I232 (60H)

I. **Introduction**

II. **Dispositifs d'affichage**

III. **Génération des tracés de base**

IV. **Transformations graphiques en 2D et 3D**

V. **Transformation de visualisation en 2D et 3D**

VI. **Formes géométriques et modèles**

VII. **Render et structure de représentation**

Elimination des parties cachées, coloriage, ombrage, éclairage,
- BSP, Brep, ...

Application (Open GL/L)