Université Lille Nord de France
Pôle de Recherche et d'Enseignement Supérieur

Université de Valenciennes et du Hainaut-Cambrésis

ALGORITHMIQUE ET PROGRAMMATION C NIVEAU 2

UVHC – ISTV – Licence 2

Rabie Ben Atitallah
rabie.benatitallah@univ-valenciennes.fr
http://www.lifl.fr/~benatita/pages/Teaching

CHAP II PROGRAMMATION ET ALGORITHMES

I INTRODUCTION

La programmation : Structurée , Modulaire,
Itérative ou récursive

Les objets : Représentation interne, Type et types scalaires, Types structurés (chaîne, tableau,
structures, fichiers), Types énumérés

Les L.P. papier , graphique : arbre programmatique,

L.P. impératifs: BASIC, PASCAL, C, ADA, JAVA

L.P. fonctionnels/déclaratifs: LISP,PROLOG,CAML

Les algorithmes:de base, numériques, non numériques

II RAPPELS DE PROGRAMMATION

1 La programmation structurée

Enchainement des actions de base : Séquence, Alternative, Répétition, pas de GOTO

Analyse descendante

2 La programmation modulaire

Modules = sous-programmes = fonctions

Fonction avec 1 seul résultat dont on donne le type

Fonction sans résultat : *void* fonction

Passage des paramètres par valeur

Fonction sans paramètre : mettre les ( ) sans paramètre

- 3 Itérative ou Récursive
- itérative: boucles TQ, POUR ou FAIRE TQ REPETER JQA condition_arret FAIRE TQ **non** condition_arret
- récursive si appel à la fonction elle-même

```
int fact ( int n )
{  int i , f ;
   f = 1 ;
   for ( i =1; i<n; i =i+1)
                  f = f * i ;
   return f ;
}
```

```
int fact ( int n )
{       if ( n = = 0 )
                  return 1 ;
                  else
   return n* fact ( n-1 );
}
```

4 Objets manipulés

Constantes ou variables

Types **Scalaires** (Ensemble ordonné de valeurs non structurées)

Structurés = non Scalaires (composition et structure)

Standard : entier , réel , caractère

Défini par l'utilisateur: Type énuméré, Tableau, Chaîne, Structure, Fichier, *Pointeur*

Représentation de structures de données

Matrices, Vecteur, Polynômes

Listes, Piles, Files d'attente, Arbre

5 Algorithmes

de base en Informatique : Permutation , Recherche séquentielle, Recherche associative, Enumération, ...

sur les types structurés : Chaînes de caractères, Fichiers

sur les structures de données : Polynômes, Vecteurs, Matrices, *Listes, Piles, etc...*

Classiques : Recherche, Tri , etc ...

Numériques : $f(x) = 0$ $y' = K(x)$

$A.X = B$ A^{-1} $|A|$

III ALGORITHMES DE BASE

1 Permutation

```
z = x ;   x = y ;   y = z ;
```

ou encore

```
z = y ;   y = x ;   x = z ;
```

```
x = x+y ;   y = x-y ;   x = x-y ;
```

ou encore

```
y = x+y ;   x = y-x ;   y = y-x ;
```

ou encore

```
y += x ;    x = y-x ;    y -= x ;
```

2 Enumération séquentielle

POUR I de 1 à N FAIRE début ... FIN

```
for ( i = 1 ; i<= n ; i = i+1 ) { ... }
```

TQ non condition_arret FAIRE début ... FIN

```
while ( !condition_arret ) { ... }
```

FAIRE ... TQ non condition_arret

```
do ... while ( !condition_arret )
```

TQ non condition_arret et non trouve FAIRE DEB FIN

```
while ( !condition_arret  && !trouve )  { ...  }
```

FAIRE ... TQ non condition_arret et non trouve

```
do  ...  while (!condition_arret  &&  !trouve )
```

```
int existe ( int t [ ] ,  int n ,  int x )
            {   int i = 0 ;
                while ( i<n  &&  t [ i ]  != x )
                            i=i+1 ;
                return  ( i<n ) ;
            }
```

```
int existe ( int t [ ] ,  int n ,  int x )
        {    int i  ;
             for ( i=0;  i<n  &&  t [ i ]  != x; i++ )
                                ;
             return  ( i<n ) ;
             }
```

```
int existe ( int t [ ] ,  int n ,  int x )
     {    int i = 0 ;
          while ( i<n   )
                    if (t [ i ] != x)      i = i+1 ;
                                           else return 1;
          return  0 ;
     }
```

4 Recherche dichotomique

Suite ordonnée et à chaque coup on divise l'intervalle de recherche en 2 (dichotomie): on part de l'intervalle [D, F] et on compare la valeur recherchée x avec la valeur au milieu. Suivant le résultat soit on a trouvé soit on réduit l'intervalle.

```
int existe_dicho (int t [ ] ,  int n ,  int x )
{    int  d = 0 ,  f = n-1 ,  m = (d+f) /2 ;
     while  ( t [ m ] != x    &&    f > d )
               {    if  ( x< t [ m ] )          f =m - 1;
                                          else  d=m + 1;
                    m = ( d + f ) / 2 ;
               }
     return  ( t [ m ] == x ) ;
}
```

IV ALGORITHMES

1 Algorithmes numériques

Résolution d'équations: $P_n(x) = 0$ $f(x) = 0$

Méthode dichotomique, Méthode de Newton

Calcul d'intégrales définies: Méthode des rectangles,

Méthode des trapèzes, Méthode de Simpson

Résolution d'équations différentielles $y' = f(x,y)$

Méthode d'Euler, Méthode de Runge-Kutta

Algèbre linéaire : Matrices, Systèmes linéaires

Eq diff d'ordre >1 ou Système d'éq diff. du 1er ordre

Interpolation polynomiale ou expon, Approximation

2 Algorithmique non numérique

Chaînes de caractères

Fichiers séquentiels

Enumération séquentielle , Rech. associative

Insertion , Suppression

Algorithmes de TRI:

tournoi, bulle, insertion, extraction

tri rapide (quicksort), tri fusion

Algorithmes sur Structures de données:

Listes, Piles, Files d'attente, Listes chaînées

Arbres, Graphes