LU1IN002 : Éléments de programmation 2
Cours 4 - partie 2
Algorithmes

Cours : Jean-Lou Desbarbieux, François Bouchet,
Mathilde Carpentier
et toute l'équipe de l'UE
LU1IN002 Sorbonne Université 2022/2023

Problème 1

Problème : étant donné un tableau de valeurs entières triées par ordre croissant `xs` et une valeur `x`, trouver un indice `i` dans `xs` tel que `xs[i]== x`

Problème 1

Problème : étant donné un tableau de valeurs entières triées par ordre croissant `xs` et une valeur `x`, trouver un indice `i` dans `xs` tel que `xs[i]== x`

Questions à se poser pour résoudre un problème et écrire une fonction

- Quels sont les entrées (arguments) ?
- Quels sont les sorties (valeur de retour ou arguments) ?
- Quelle méthode employer (algorithme) ?
- Tous les cas sont ils bien prévus ?
- Comment tester la fonction ?

Questions à se poser

- Quels sont les entrées (arguments) ?

Questions à se poser

- Quels sont les entrées (arguments) ?
  Le tableau, sa taille et la valeur cherchée.

Questions à se poser

- Quels sont les entrées (arguments) ?
  Le tableau, sa taille et la valeur cherchée.
- Quels sont les sorties (valeur de retour ou arguments) ?

Questions à se poser

- Quels sont les entrées (arguments) ?
  Le tableau, sa taille et la valeur cherchée.
- Quels sont les sorties (valeur de retour ou arguments) ?
  Valeur de retour : l'indice dans le tableau de la valeur cherchée ou -1 si elle n'est pas présente.

Questions à se poser

- Quels sont les entrées (arguments) ?
  Le tableau, sa taille et la valeur cherchée.
- Quels sont les sorties (valeur de retour ou arguments) ?
  Valeur de retour : l'indice dans le tableau de la valeur cherchée ou -1 si elle n'est pas présente.
- Quelle méthode employer (algorithme) ?

Questions à se poser

- Quels sont les entrées (arguments) ?
  Le tableau, sa taille et la valeur cherchée.
- Quels sont les sorties (valeur de retour ou arguments) ?
  Valeur de retour : l'indice dans le tableau de la valeur cherchée ou -1 si elle n'est pas présente.
- Quelle méthode employer (algorithme) ?
  Parcourir tout le tableau en comparant les valeurs à la valeur cherchée. S'arrêter si on trouve la valeur. retourner -1 si aucun valeur du tableau ne correspond à la valeur cherchée.

Questions à se poser

- Quels sont les entrées (arguments) ?
  Le tableau, sa taille et la valeur cherchée.
- Quels sont les sorties (valeur de retour ou arguments) ?
  Valeur de retour : l'indice dans le tableau de la valeur cherchée ou -1 si elle n'est pas présente.
- Quelle méthode employer (algorithme) ?
  Parcourir tout le tableau en comparant les valeurs à la valeur cherchée. S'arrêter si on trouve la valeur. retourner -1 si aucun valeur du tableau ne correspond à la valeur cherchée.
- Tous les cas sont ils bien prévus ?

Questions à se poser

- Quels sont les entrées (arguments) ?
  Le tableau, sa taille et la valeur cherchée.
- Quels sont les sorties (valeur de retour ou arguments) ?
  Valeur de retour : l'indice dans le tableau de la valeur cherchée ou -1 si elle n'est pas présente.
- Quelle méthode employer (algorithme) ?
  Parcourir tout le tableau en comparant les valeurs à la valeur cherchée. S'arrêter si on trouve la valeur. retourner -1 si aucun valeur du tableau ne correspond à la valeur cherchée.
- Tous les cas sont ils bien prévus ?
  2 cas : la valeur est présente, la valeur est absente. Ils sont bien prévus.

Questions à se poser

- Quels sont les entrées (arguments) ?
  Le tableau, sa taille et la valeur cherchée.
- Quels sont les sorties (valeur de retour ou arguments) ?
  Valeur de retour : l'indice dans le tableau de la valeur cherchée ou -1 si elle n'est pas présente.
- Quelle méthode employer (algorithme) ?
  Parcourir tout le tableau en comparant les valeurs à la valeur cherchée. S'arrêter si on trouve la valeur. retourner -1 si aucun valeur du tableau ne correspond à la valeur cherchée.
- Tous les cas sont ils bien prévus ?
  2 cas : la valeur est présente, la valeur est absente. Ils sont bien prévus.
- Comment tester la fonction ?

Questions à se poser

- Quels sont les entrées (arguments) ?
  Le tableau, sa taille et la valeur cherchée.
- Quels sont les sorties (valeur de retour ou arguments) ?
  Valeur de retour : l'indice dans le tableau de la valeur cherchée ou -1 si elle n'est pas présente.
- Quelle méthode employer (algorithme) ?
  Parcourir tout le tableau en comparant les valeurs à la valeur cherchée. S'arrêter si on trouve la valeur. retourner -1 si aucun valeur du tableau ne correspond à la valeur cherchée.
- Tous les cas sont ils bien prévus ?
  2 cas : la valeur est présente, la valeur est absente. Ils sont bien prévus.
- Comment tester la fonction ?
  Tester si on trouve bien une valeur en première case, en dernière case, dans une case quelconque, et si on retourne bien -1 si la valeur est absente.

Code en C

```c
/*...*/
int Rechercher_val(int tab[], int nbEl, float val){
  int i;
  for(i=0;i<nbEl;i++){
    if(tab[i]==val){
      return i;
  }}
  return -1;
}
int main() {
      int tab[6] ={1, 5, 7, 8, 10, 19 };
      printf("La valeur %d est à l'indice %d\n", 1,
          Rechercher_val(tab, 6, 1));
      printf("La valeur %d est à l'indice %d\n", 19,
          Rechercher_val(tab, 6, 19));
      printf("La valeur %d est à l'indice %d\n", 7,
          Rechercher_val(tab, 6, 7));
      printf("La valeur %d est à l'indice %d\n", 99,
          Rechercher_val(tab, 6, 99));
      return 0;
}
```

Problème 1

Problème : étant donné un tableau de valeurs triées par ordre croissant `xs` et une valeur `x`, trouver un indice `i` dans `xs` tel que `xs[i]== x`.

Rechercher une valeur coutera `n` comparaisons dans le pire des cas, `n` étant la taille du tableau.

Ne pourrait-on pas être plus efficace ?
Et utiliser le fait que le tableau est trié ?

Problème 1

Problème : étant donné un tableau de valeurs triées par ordre croissant `xs` et une valeur `x`, trouver un indice `i` dans `xs` tel que `xs[i]== x`.

Hypothèse : les valeurs sont rangées par ordre croissant dans le tableau `xs`.

Question : que faire si `x` n'a pas d'occurence dans `xs` ?
Réponse : renvoyer `-1`

Méthode : procéder à une *recherche dichotomique*

Recherche dichotomique

Idée :

- regarder la valeur au milieu du tableau :
- si c'est la valeur cherchée, donner l'indice
- sinon,
  - si la valeur cherchée est plus petite que la valeur au milieu du tableau alors chercher dans la partie gauche du tableau
  - sinon, chercher dans la partie droite du tableau

Recherche dichotomique

Idée :

- regarder la valeur au milieu du tableau :
- si c'est la valeur cherchée, donner l'indice
- sinon,
  - si la valeur cherchée est plus petite que la valeur au milieu du tableau alors chercher dans la partie gauche du tableau
  - sinon, chercher dans la partie droite du tableau

Question : quand est-ce qu'on s'arrête ?

- quand on a trouvé la valeur,
- quand le (sous)tableau est vide.

Mise en œuvre

Question : qu'est-ce qu'un *sous-tableau* du tableau `xs` ?
$\Longrightarrow$ les valeurs de **xs** comprises entre deux indices `ig` et `id` en supposant `ig` $\leq$ `id`[1]

Question : quelle est la valeur ≪au milieu≫ du sous tableau de `xs` délimité par `ig` et `id` ?
$\Longrightarrow$ la valeur de `xs` à l'indice `(ig+id)/2`
En effet : `(ig+id)/2` est égal à `ig + (id-ig)/2`

Question : quand le sous tableau délimité par `ig` et `id` est-il vide ?
$\Longrightarrow$ lorsque `id` $<$ `ig`

[1] En python on écrirait `xs[ig:id+1]` mais ça n'est pas permis en C

Code C

```c
/*...*/
int Recherche_dicho(int tab[], int nbEl, int val) {
  int ig = 0,  id = nbEl-1, i;
  while ( ig <= id ) {
    i = (ig + id) / 2;
    if ( tab[i] == val) {
      return i;
    }
    if (val < tab[i]) {
      id = i-1;
    }
    else{
      ig = i+1;
   }
  }
 return -1;
}
```

Questions à se poser

- Quels sont les entrées (arguments) ?
  Le tableau, sa taille et la valeur cherchée.
- Quels sont les sorties (valeur de retour ou arguments) ?
  Valeur de retour : l'indice dans le tableau de la valeur cherchée ou -1 si elle n'est pas présente.
- Quelle méthode employer (algorithme) ?
  Parcourir tout le tableau en comparant les valeurs à la valeur cherchée. S'arrêter si on trouve la valeur. retourner -1 si aucun valeur du tableau ne correspond à la valeur cherchée.
- Tous les cas sont ils bien prévus ?
  2 cas : la valeur est présente, la valeur est absente. Ils sont bien prévus.
- Comment tester la fonction ?

Tests

Considérer :

- des cas où l'on trouve et des cas où l'on de trouve pas
- des tableaux de longueur paire et des tableaux de longueur impaire (division entière par 2)
- des tableaux *limites* (longueur 1, longueur 2)

Tests : implémentation

Pouvoir comparer la valeur obtenue avec la valeur attendue :

```
int main() {

  double tab1[] = { 2 };
  printf("%d == 0\n", Recherche_dicho(tab1,1, 2));
  printf("%d == -1\n", Recherche_dicho(tab1,1, 1));
  printf("%d == -1\n", Recherche_dicho(tab1,1, 3));
  double tab2[] = { 2, 4 };
  printf("%d == 0\n", Recherche_dicho(tab2,2, 2));
  printf("%d == 1\n", Recherche_dicho(tab2,2, 4));
  printf("%d == -1\n", Recherche_dicho(tab2,2, 1));
  printf("%d == -1\n", Recherche_dicho(tab2,2, 3));
  printf("%d == -1\n", Recherche_dicho(tab2,2, 6));
  double tab3[] = { 2, 4, 6 };
  printf("%d == 0\n", Recherche_dicho(tab3,3, 2));
  printf("%d == 1\n", Recherche_dicho(tab3,3, 4));
  printf("%d == 2\n", Recherche_dicho(tab3,3, 6));
  printf("%d == -1\n", Recherche_dicho(tab3,3, 1));
  printf("%d == -1\n", Recherche_dicho(tab3,3, 3));
  printf("%d == -1\n", Recherche_dicho(tab3,3, 5));
  printf("%d == -1\n", Recherche_dicho(tab3,3, 7));

}
```

Problème 2

Problème : calculer un tableau trié à partir d'un tableau quelconque
Contrainte : ordonner le tableau **en place**, par permutation de ses éléments.
Méthode : *Tri par sélection*

1. Trouver le minimum
2. L'échanger avec la première valeur
3. Recommencer avec le sous-tableau commençant à la deuxième case

On s'arrête quand le tableau n'a plus qu'un élément.

Questions à se poser

- Quels sont les entrées (arguments) ?
- Quels sont les sorties (valeur de retour ou arguments) ?
- Quelle méthode employer (algorithme) ?
- Tous les cas sont ils bien prévus ?
- Comment tester la fonction ?

Questions à se poser

- Quels sont les entrées (arguments) ?
  le tableau et son nombre d'éléments.
- Quels sont les sorties (valeur de retour ou arguments) ?
  le tableau (trié), attention c'est le même tableau
- Quelle méthode employer (algorithme) ?
  cf. diapo précédente.
- Tous les cas sont ils bien prévus ?
  Tableau de taille 0 ? 1 ? n ?
- Comment tester la fonction ?

Algorithme itératif

- Tant que len > 1
  - Trouver l'indice du minimum
  - Echanger le minimum avec la première valeur.
  - Faire débuter le tableau à la seconde case et décrémenter len

Code C

```c
/*Trier un tableau par selection.
  tab: le tableau
  nbEl: le nombre de valeurs dans le tableau
Modifie tab pour qu'il soit trie*/
void select_sort_it(int *tab, int len) {
  int i;
  while (len > 1) {
        i = min_pos(tab,len);
        swap(&tab[0],&tab[i]);
        tab = &tab[1];
        len = len-1;
  }
}
```

Comment tester ?

```c
/*Verifier qu'un tableau est trie.
  tab: le tableau
  len: le nombre de valeurs dans le tableau
Retourne 1 si le tableau est trie, 0 sinon*/
int Tab_trie(int tab[], int len) {
  int i;
  for(i=0; i < len-1; i++) {
    if (tab[i] > tab[i+1]) {
      return 0;
    }
  }
  return 1;
}
int main(){
  int tab[6]={4, 8, 2, 1, 9, -6};
  int tab2[6]={4, 8, 2, 1, 9, -6};

  select_sort_rec(tab, 6);
  if( Tab_trie(tab, 6)==0){
    printf("Le tableau tab n'est pas trié\n");
  }else{
    printf("Le tableau tab est trié\n");}
  select_sort_it(tab2, 6);
  if( Tab_trie(tab2, 6)==0){
    printf("Le tableau tab2 n'est pas trié\n");
  }else{
    printf("Le tableau tab2 est trié\n");}
  return 0;
}
```