

Introduction aux algorithmes IV

Algorithmes et structures de données, 2025-2026

P. Albuquerque (B410) et O. Malaspinas (A401), ISC, HEPIA
2025-10-07

En partie inspiré des supports de cours de P. Albuquerque

Le tri par sélection

Tri par sélection (1/3)

Problématique

Trier un tableau par ordre croissant.

Idée d'algorithme

```
ind = 0
tant que (ind < SIZE-1)
    Trouver le minimum du tableau, tab_min = min([ind:SIZE-1]).
    Échanger tab_min avec tab[ind]
    ind += 1
```

Tri par sélection (2/3)

Quel est l'algorithme du tri par sélection ?

Quel est l'algorithme du tri par sélection ?

1. Soit un tableau d'entiers, `tab[0:SIZE-1]` et `i = 0`.
2. Trouver l'indice, `j`, de `tab[i:SIZE-1]` où la valeur est minimale.
3. Échanger `tab[i]` et `tab[j]`.
4. `i += 1` et revenir à 2, tant que `i < SIZE-1`.

Implémentation par groupe de 3

- Initialiser aléatoirement un tableau de double de taille 10 ;
- Afficher le tableau ;
- Trier par sélection le tableau ;
- Afficher le résultat trié ;
- Vérifier algorithmiquement que le résultat est bien trié.

Les chaînes de caractères

Un type de tableau particulier

Les chaînes de caractères

```
string = tableau + char + magie noire
```


Le type `char`

- Le type `char` est utilisé pour représenter un caractère.
- C'est un entier 8 bits signé.
- En particulier :

- Écrire

```
char c = 'A';
```

- Est équivalent à :

```
char c = 65;
```

- Les fonctions d'affichage interprètent le nombre comme sa valeur ASCII.

Chaînes de caractères (strings)

- Chaîne de caractère == tableau de caractères **terminé par la valeur** `'\0'` ou `0`.

Exemple

```
char *str = "HELLO !";  
char str[] = "HELLO !";
```

Est représenté par

char	H	E	L	L	O		!	\0
ASCII	72	69	76	76	79	32	33	0

Chaînes de caractères (strings)

- Chaîne de caractère == tableau de caractères **terminé par la valeur** `'\0'` ou `0`.

Exemple

```
char *str = "HELLO !";  
char str[] = "HELLO !";
```

Est représenté par

char	H	E	L	L	O		!	\0
ASCII	72	69	76	76	79	32	33	0

A quoi sert le `\0` ?

Chaînes de caractères (strings)

- Chaîne de caractère == tableau de caractères **terminé par la valeur** `'\0'` ou `0`.

Exemple

```
char *str = "HELLO !";  
char str[] = "HELLO !";
```

Est représenté par

char	H	E	L	L	O		!	\0
ASCII	72	69	76	76	79	32	33	0

A quoi sert le `\0` ?

Permet de connaître la fin de la chaîne de caractères (pas le cas des autres sortes de tableaux).

Syntaxe

```
char name[5];  
name[0] = 'P'; // = 70;  
name[1] = 'a'; // = 97;  
name[2] = 'u'; // = 117;  
name[3] = 'l'; // = 108;  
name[4] = '\0'; // = 0;  
char name[] = {'P', 'a', 'u', 'l', '\0'};  
char name[5] = "Paul";  
char name[] = "Paul";  
char name[100] = "Paul is not 100 characters long.";
```

Fonctions

- Il existe une grande quantité de fonction pour la manipulation de chaînes de caractères dans `string.h`.
- Fonctions principales :

```
// longueur de la chaîne (sans le \0)
size_t strlen(char *str);
// copie jusqu'à un \0
char *strcpy(char *dest, const char *src);
// copie len char
char *strncpy(char *dest, const char *src, size_t len);
// compare len chars
int strncmp(char *str1, char *str2, size_t len);
// compare jusqu'à un \0
int strcmp(char *str1, char *str2);
```

- Pour avoir la liste complète : `man 3 string`.

Fonctions

- Il existe une grande quantité de fonction pour la manipulation de chaînes de caractères dans `string.h`.
- Fonctions principales :

```
// longueur de la chaîne (sans le \0)  
size_t strlen(char *str);  
// copie jusqu'à un \0  
char *strcpy(char *dest, const char *src);  
    // copie len char  
char *strncpy(char *dest, const char *src, size_t len);  
// compare len chars  
int strncmp(char *str1, char *str2, size_t len);  
// compare jusqu'à un \0  
int strcmp(char *str1, char *str2);
```

- Pour avoir la liste complète : man 3 string.

Quel problème peut se produire avec `strlen`, `strcpy`, `strcmp` ?

Fonctions

- Il existe une grande quantité de fonction pour la manipulation de chaînes de caractères dans `string.h`.
- Fonctions principales :

```
// longueur de la chaîne (sans le \0)  
size_t strlen(char *str);  
// copie jusqu'à un \0  
char *strcpy(char *dest, const char *src);  
    // copie len char  
char *strncpy(char *dest, const char *src, size_t len);  
// compare len chars  
int strncmp(char *str1, char *str2, size_t len);  
// compare jusqu'à un \0  
int strcmp(char *str1, char *str2);
```

- Pour avoir la liste complète : man 3 string.

Quel problème peut se produire avec `strlen`, `strcpy`, `strcmp` ?

- Si `\0` est absent... on a un comportement indéfini.

Les anagrammes

Les anagrammes

Définition

Deux mots sont des anagrammes l'un de l'autre quand ils contiennent les mêmes lettres mais dans un ordre différent.

Exemple

t	u	t	u	t	\0
t	u	t	t	u	\0

Problème : Trouvez un algorithme pour déterminer si deux mots sont des anagrammes.

Les anagrammes

Il suffit de :

1. Trier les deux mots.
2. Vérifier s'ils contiennent les mêmes lettres.

Implémentation ensemble

```
int main() { // pseudo C
    tri(mot1);
    tri(mot2);
    if egalite(mot1, mot2) {
        // anagrammes
    } else {
        // pas anagrammes
    }
}
```

Les palindromes

Les palindromes

Mot qui se lit pareil de droite à gauche que de gauche à droite :

Les palindromes

Mot qui se lit pareil de droite à gauche que de gauche à droite :

- rotor, kayak, ressasser, ...

Problème : proposer un algorithme pour détecter un palindrome

Les palindromes

Mot qui se lit pareil de droite à gauche que de gauche à droite :

- rotor, kayak, ressasser, ...

Problème : proposer un algorithme pour détecter un palindrome

Solution 1

```
while (first_index < last_index) {  
    if (mot[first_index] != mot [last_index]) {  
        return false;  
    }  
    first_index += 1;  
    last_index -= 1;  
}  
return true;
```

Les palindromes

Mot qui se lit pareil de droite à gauche que de gauche à droite :

- rotor, kayak, ressasser, ...

Problème : proposer un algorithme pour détecter un palindrome

Solution 1

```
while (first_index < last_index) {  
    if (mot[first_index] != mot [last_index]) {  
        return false;  
    }  
    first_index += 1;  
    last_index -= 1;  
}  
return true;
```

Solution 2

```
mot_tmp = revert(mot);  
return mot == mot_tmp;
```


Le crible d'Ératosthène

Crible d'Ératosthène

Algorithme de génération de nombres premiers.

Exercice

- À l'aide d'un tableau de booléens,
- Générer les nombres premiers plus petits qu'un nombre N

Pseudo-code

- Par groupe de trois, réfléchir à un algorithme.

Programme en C

- Implémenter l'algorithme et le poster sur le salon `Element`.