

## First Sequences: Strings

### Parcourir une chaîne

#### ALGO

```
procedure print_string(chaine s)
  variables
    entier i
debut
  i ← 1
  n = longueur(s)
  tant que i <= n faire
    ecrire(s[i], '\n')
    i ← i + 1
  fin tant que
fin
```

```
procedure print_string(chaine s)
  variables
    entier i
debut
  pour i ← 1 jusqu'a longueur(s) faire
    ecrire(s[i], '\n')
  fin pour
fin
```

---

## 1 Des classiques

### Exercice 1.1 (Recherches)

1. Écrire une fonction qui recherche si un caractère donné appartient à une chaîne. La fonction devra retourner la position du premier caractère trouvée, la valeur -1 si celui-ci n'est pas présent.
2. Écrire une fonction qui cherche si une chaîne est sous-chaîne d'une autre. La fonction devra retourner la position du premier caractère de la sous-chaîne si le test est positif, la valeur -1 sinon.

---

### Exercice 1.2 (Palindrome)

Écrire une fonction qui détermine si une chaîne est un palindrome.

*Quelques palindromes :*

- Engage le jeu que je le gagne!
- Never odd or even.
- Nice hat, Bob Tahecin.
- God! A red nugget! A fat egg under a dog!

Trois niveaux :

**level 0 :** La chaîne ne contient que des lettres minuscules non accentuées (pas d'espaces).

Ex : "engagelejeujelegagne".

**level 1 :** La chaîne contient uniquement des lettres minuscules non accentuées et des espaces. Le premier et le dernier caractères ne peuvent pas être des espaces, et il ne peut y avoir deux espaces qui se suivent.

Ex : "nice hat bob tahecin".

**level + :** La chaîne contient tout type de caractères : accentués, majuscules, ponctuation...

Ex : "Tu l'as trop écrasé César, ce port salut."

## 2 Un peu d'archi et ...

### Exercice 2.1 (Conversions)

1. Écrire une fonction qui convertit un entier (relatif)  $n$  en son équivalent en binaire "complément à 2" sur  $p$  bits (représenté par une chaîne).

*Exemples de sortie :*

```
1 >>> integer_to_twoscomp(-42, 8)
2 '11010110'
3 >>> integer_to_twoscomp(42, 8)
4 '00101010'
```

2. Écrire la fonction qui fait la conversion inverse :

```
1 >>> twoscomp_to_integer("11010110", 8)
2 -42
3 >>> twoscomp_to_integer("00101010", 8)
4 42
```

### Exercice 2.2 (Fréquence)

1. Écrire une fonction qui retourne le caractère le plus fréquent d'une chaîne, ainsi que son nombre d'occurrences.
2. On donne les fonctions suivantes :

```
1 >>> help(ord)
2 ord(c) -> integer
3 Return the integer ordinal of a one-character string.
4
5 >>> ord('A')
6 65
7
8 >>> help(chr)
9 chr(i) -> Unicode character
10 Return a Unicode string of one character with ordinal i...
11
12 >>> chr(65)
13 'A'
```

On suppose que la chaîne ne contient que des caractères "classiques" (codés entre 0 et 255).

Écrire une version plus optimale de la fonction de la question précédente.

3. Écrire une fonction qui compte le nombre de caractères différents dans une chaîne de caractères.

