Langage C : Chaînes de caractères

Sébastien Jean

IUT de Valence
Département Informatique

v1.0, 12 décembre 2025

IUT Valence
Université Grenoble Alpes

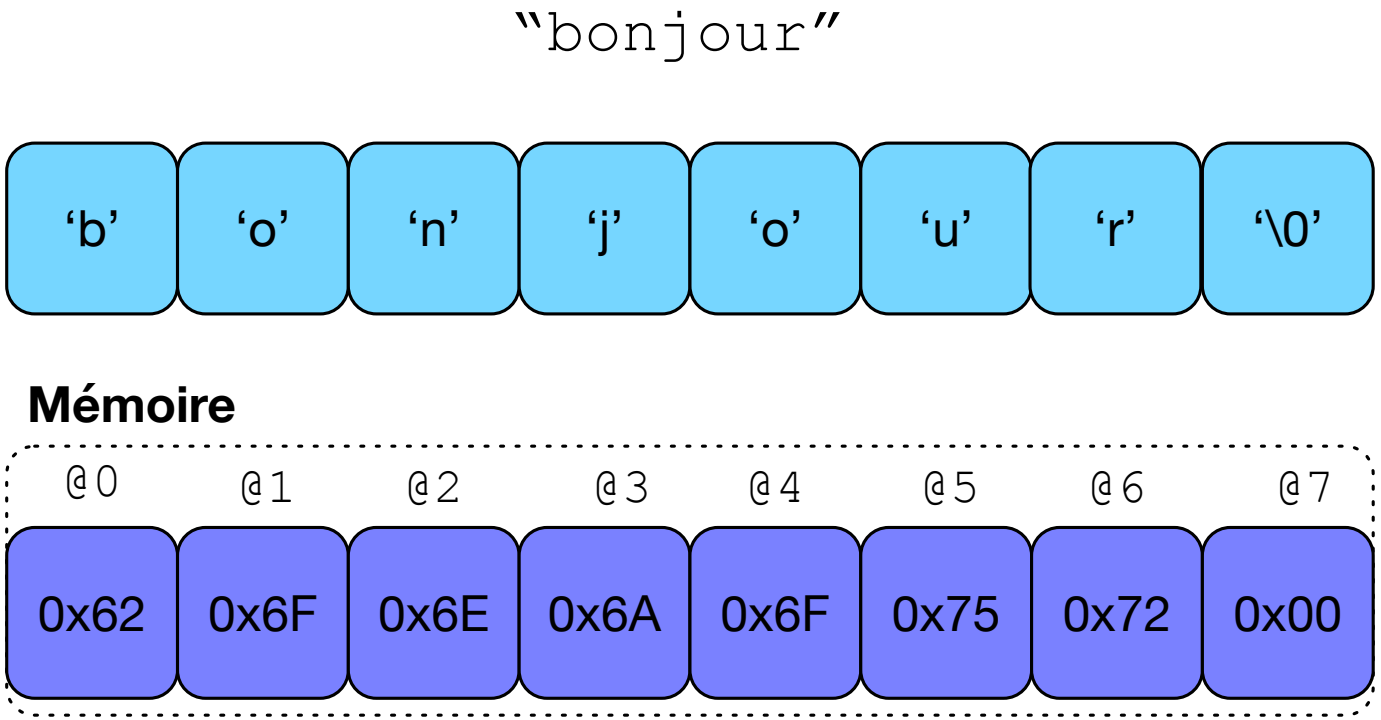
Interlude : Création/clonage d'un projet Gitlab

- **Créer un projet** `ChainesC` sur Gitlab (avec un `README`)
- **Cloner** le projet depuis *VsCode* et ouvrir le dépôt
- **Modifier** le fichier `README.md` pour indiquer à quoi sert ce projet
- N.B. : pour chacun des exemples/exercices `ExX` suivant :
  - **Ecrire le programme** dans `ExX/src/main.c` et le compiler dans `ExX/build/ExX`
  - `Rédiger (si nécessaire) un jeu d'essai` dans un fichier `ExerciceX/Essai`

IUT Valence Université Grenoble Alpes

Représentation des chaines de caractères

- Une chaine de caractères en C est une suite de valeurs de type `char` (1 octet) terminée par `'\0'`

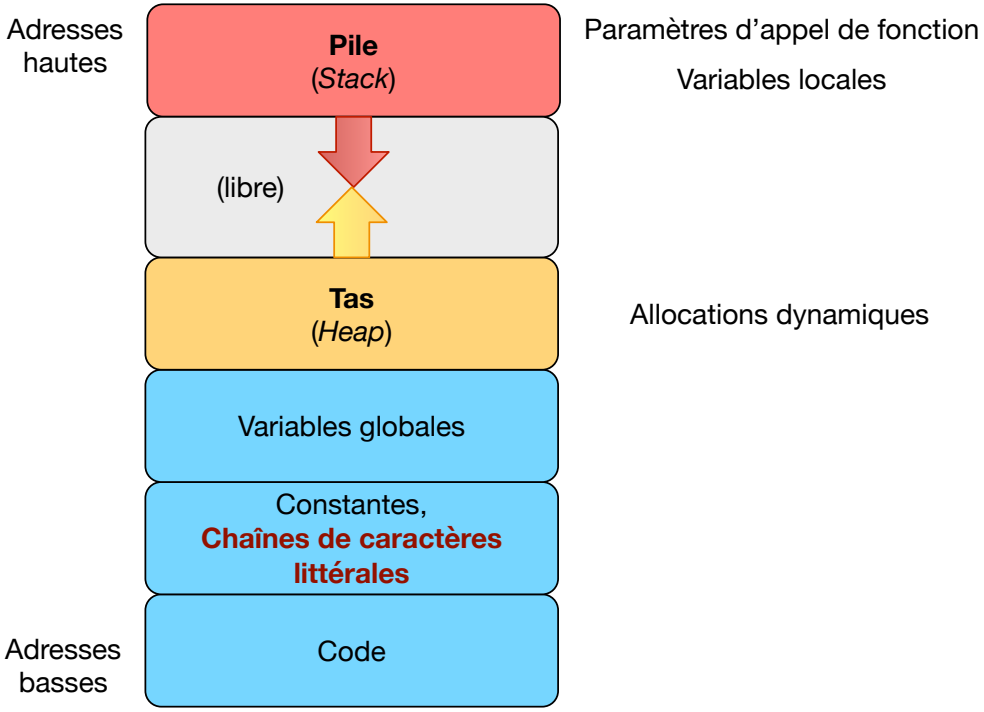


- Les valeurs de type `char` sont interprétés en fonction du jeu de caractères local lors de la compilation et l'exécution

IUT Valence Université Grenoble Alpes

Chaines de caractères constantes

- Les chaines de caractères constantes (littérales) sont exprimées entre guillemets (raccourci de syntaxe)
  - Les caractères spéciaux sont échappés avec `'\'`
    - cf. `https://en.wikipedia.org/wiki/Escape_sequences_in_C`
- Elles sont stockées dans une zone dédiée, en lecture seule, hors de la pile et du tas.



Variables de type chaîne de caractères

- Une chaîne peut être manipulée à travers 2 types de variables :
  - un tableau de caractères → `char[]`
    - un tableau de `n` caractères permet de représenter une chaîne de `n-1` caractères (+ `'\0'`)
  - un pointeur vers un caractère (le premier de la chaine) → `char *`

Code C

```c
char ch1[8] = "bonjour";
// Le tableau est alloué dans la pile,
// les caractères de la chaine y sont recopiés
// (le caractère '\0' est inclus)

char *ch2 = "bonjour";
// La chaine littérale est allouée dans la zone dédiée
// (le caractère '\0' est inclus),
// ch2 contient l'adresse de la chaine
```

Variables de type chaîne de caractères

Code C

```c
char ch1[8] = "bonjour";
ch1[0] = 'B';
// Modification autorisée car allocation dans la pile

char *ch2 = "bonjour";
ch2[0] = 'B';
// Modification interdite car allocation dans la zone
    dédiée
```

IUT Valence
Université Grenoble Alpes

Fonction de manipulation de chaînes

- Inclusion de `string.h` nécessaire

Signature C

```c
/**
* Obtention de la longueur d'une chaîne.
* (caractère '\0' non inclus)
*/
int strlen(char *s);

/**
* Comparaison lexicographique de 2 chaînes.
* s1 : la première chaine
* s2 : la seconde chaine
* retour : positif si s1 > s2, négatif si s1 < s2,
*          nul si s1 = s2
*/
int strcmp(const char *s1, const char *s2);
```

Fonction de manipulation de chaînes

Signature C

```c
/**
* Concaténation de 2 chaînes.
* dest : la chaine destination
* append : la chaine à ajouter en fin de chaîne
* retour : dest (dest est modifiée en place)
*/
char *strcat(char *dest, const char *append);

/**
* Variante avec un nombre maximum
* de caractères à ajouter.
*/
char *strncat(char *dest, const char *append, int n);
```

Fonction de manipulation de chaînes

Signature C

```c
/**
* Recopie d'une chaîne.
* src : la chaine source
* dest : la chaine destination
* retour : dest (dest est modifiée en place)
*/
char *strcpy(char *dest, const char *src);

/**
* Variante avec un nombre maximum
* de caractères à recopier.
*/
char *strncpy(char *dest, const char *src, int n);
```

Fonction de manipulation de chaînes

Signature C

```c
/**
* Duplication d'une chaîne.
* s : la chaine à dupliquer
* retour : une nouvelle chaine, copie de s
*/
char *strdup(char *s);

/**
* Variante avec un nombre maximum
* de caractères à copier.
*/
char *strndup(char *s, int n);
```

IUT Valence Université Grenoble Alpes

Fonction de manipulation de chaînes

Signature C

```c
/**
* Recherche de la première occurrence
* d'un caractère dans une chaîne.
* s : la chaine où chercher
* c : le caractère
* retour : un pointeur sur le caractère trouvé
* ou NULL ni non trouvé
*/
char *strchr(const char *s, int c);

/**
* Variante pour la dernière occurrence.
*/
char *strrchr(const char *s, int c);
```

Fonction de manipulation de chaînes

Signature C

```c
/**
* Recherche de la première occurrence
* d'une sous-chaîne dans une chaîne.
* s1 : la chaîne où chercher
* s2 : la sous-chaîne à chercher
* retour : un pointeur sur le début de la
* sous-chaîne ou NULL ni non trouvé
*/
char *strstr(const char *s1, const char *s2);
/**
* Copie d'octets.
* dest : destination
* src : source
* n : nombre d'octets à copier
* retour : dest (dest modifé en place)
*/
void *memcpy(void *dest, void *src, int n);
```

Exercice

- Ecrire un module `morestring` (`.h`/`.c`) contenant les fonctions décrites dans la suite
  - Les pré-conditions seront supposées satisfaites
  - Chaque fonction sera validée par une application mettant en œuvre des jeux de tests pertinents

IUT Valence
Université Grenoble Alpes

Exercice

Signature C

```c
/**
* Recherche de la première occurrence
* d'un caractère dans une chaine.
* s : la chaîne où chercher
* c : le caractère à chercher
* pré-conditions : aucune
* post-condition : la valeur retournée est soit
* l'indice trouvé (à partir de 0) , soit -1 sinon
*/
int indice_de(const char *s, char c);
```

Exercice

Signature C

```c
/**
* Extraction d'une sous-chaine entre 2 indices.
* s : la chaîne où extraire
* b : indice de début
* e : indice de fin
* pré-conditions :
* - s n'est pas vide
* - 0 <= e >= b < longueur(s)
* post-condition : une nouvelle chaîne contenant
* les caractères e à b (inclus) de s est retournée
*/
char *sous_chaine(const char *s, int b, int e);
```

IUT Valence Université Grenoble Alpes

Exercice

Signature C

```c
/**
* Inversion (de l'ordre) des caractères d'une chaîne.
* s : la chaîne à inverser
* pré-conditions : aucune
* post-condition : une nouvelle chaîne contenant
* les caractères de s dans l'ordre inverse est
* retournée
*/
char *inverse(const char *s);
```

IUT Valence Université Grenoble Alpes

Exercice

Signature C

```c
/**
* Test pour savoir si une chaîne est un palyndrome.
* s : la chaîne à tester
* pré-conditions : aucune
* post-condition : retourne vrai si s est un
* palyndrome, faux sinon
*/
bool est_palyndrome(const char *s);
```

IUT Valence Université Grenoble Alpes

Exercice

Signature C

```c
/**
* Mélange (aléatoire) des caractères d'une chaîne.
* s : la chaîne à mélanger
* pré-conditions : aucune
* post-condition : une nouvelle chaîne contenant les
* caractères de s dans un ordre quelconque est
* retournée
*/
char *melange(const char *s);
```

IUT Valence Université Grenoble Alpes

Exercice

Signature C

```c
/**
* Chiffrement de César d'une chaine
* (décalage alphabétique de n positions)
* s : la chaîne à chiffrer
* n : le décalage
* pré-conditions :
* - s est composée de caractères dont les codes
* ASCII sont dans [32,126]
* - 0 <= n <= 94
* - 0 <= e >= b < longueur(s)
* post-condition : une nouvelle chaînes chiffrée avec
* le code de cesar de décalage n est retournée
*/
char *cesar(const char *s, int n);
```

Fin !

IUT Valence
Université Grenoble Alpes