Chapitre V
Chaînes de Caractères & Enregistrements

Sommaire

Cours Elearning :
https://elearning.univ-bejaia.dz/course/view.php?id=2749

Page facebook :
https://www.facebook.com/InitiationAlgoProgrammation/

La chaîne Youtube :
https://www.youtube.com/c/AlgoProgrammation1èreAnnéeTechnologie

La playlist sur le langage C :
https://youtube.com/playlist?list=PLwHHAvorm5F-tL9EXDEH0miOKmAj7iUTU

Adapté par : Redouane OUZEGGANE
rouzeggane@gmail.com - redouane.ouzeggane@univ-bejaia.dz

Chapitre V : Chaîne de caractères & Enregistrements

V.1. Introduction

Dans ce chapitre, nous allons voir brièvement, le type chaîne de caractère, ainsi que le type enregistrement en algorithmique et en langage C. Le premier est nécessaire pour présenter tout type d'information textuelle, comme le nom, prénom, filière, adresse, *etc.*Le second est important pour regrouper une ensemble de variables (informations) de différents types et de différentes désignation dans une structure dite « enregistrement », ceci permet de représenté facilement des objets complexes du problème traité.

V.2. Chaîne de caractères

Le type chaîne de caractères nous permet de stocker et de manipuler toute information textuelle. Dans cette section, nous allons voir comment déclarer, lire, afficher et réaliser des quelques traitements sur les chaînes de caractères, et ceci en algorithmique ainsi que en langage C.

V.2.1. Déclaration d'une chaîne de caractère

Pour déclarer une variable de type chaîne de caractère, on suit la syntaxe suivante :

ALGORITHME

```
<id_str> : String[<taille>];
```

LANGAGE C

```
char <id_str>[<taille>]
```

Tel-que :

<id_str> : le nom de la chaîne de caractères

<taille> : taille maximale de la chaîne de caractère.

Par exemple :

ALGORITHME

```
nom, prenom : String[25];
```

LANGAGE C

```
char nom[25], prenom[25];
```

Dans la **RAM,** une variable chaîne de caractères est représentée comme un tableau de caractères. Ainsi, on peut la parcourir comme un tableau normal avec un indice afin d'accéder aux caractères de cette chaîne. Ce tableau se termine par le caractère '\0' indiquant la fin de chaîne de caractères.

En langage C, on peut déclarer et initialiser une chaîne de caractères, comme suit :

LANGAGE C
```
char str[100] = "Cours d'Algorithmique";
```

Donc, pour la variable *str* elle sera représentée comme suit :

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	.....	99
C	o	u	r	s		d	'	A	l	g	o	r	i	t	h	m	i	q	u	e	\0	...	

Ainsi, on peut modifier un caractère en utilisant son indice dans la chaîne de caractères, par exemple : str[5] = '_'; , la chaîne de caractère deviendra :

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	.....	99
C	o	u	r	s	_	d	'	A	l	g	o	r	i	t	h	m	i	q	u	e	\0	...	

Sin on affiche cette chaîne de caractères, sur l'écran : `Cours d'Algorithmique`

V.2.3. Lecture et affichage d'une chaîne de caractère

Une chaîne de caractère est lue et affichée comme toute autre variable de type numérique : entier et réel. La syntaxe de lecture d'une chaîne de caractères est commet suit :

ALGORITHME
```
Lire(<id_str>);
```

LANGAGE C
```
scanf("%s", <id_str>);
```

Et la syntaxe d'écriture (d'affichage) :

ALGORITHME
```
Écrire(<id_str>);
```

LANGAGE C
```
printf("%s", <id_str>);
```

Tel-que : **<id_str>** est un identifiant d'une chaîne de caractère.

Dans l'exemple sur le lien suivant : https://onlinegdb.com/YLM07wPsl, vous aurez un exemple d'une lecture et d'affichage d'une chaîne de caractères.

Remarque :

1- Pour lire une chaîne de caractères avec des espaces, ce qui n'est pas possible avec le format :

%s, nous devons utilise le format : **%[^\n]** . Voir l'exemple sur ce lien : https://onlinegdb.com/y17M3Q1kN8 .

2- Lorsqu'on veut lire une liste de chaîne de caractères, on peut avoir un problème de lectures suivantes qui ne peuvent pas être effectuée : la première lecture se passe correctement, par contre la deuxième ne peut être effectuée, voir l'exemple suivant :

LANGAGE C

```
char str[100];
do{
      printf("Donner une chaîne de caractères : ");
      scanf("%[^\n]", str);
      printf("Vous avez introduit : '%s'.\n\n", str);

while (true);
```

Dan l'exemple précédent (voir le lien : https://onlinegdb.com/xyQuxY6kL), après la première lecture, on aura une boucle infinie (faites cltr+c pour terminer le programme), puisque le deuxième `scanf` n'est pas bloquant. Pour remédier à ce problème, il faut vider le *buffer de lecture* après chaque lecture de chaîne de caractères, avec l'instruction : `scanf("%c", &temp);` tel-que temp est une variable temporaire de type caractère :

LANGAGE C

```
char str[100];
char temp;
do{
      printf("Donner une chaîne de caractères : ");
      scanf("%[^\n]", str);
      scanf("%c", &temp); //Vider le tampon (buffer) de lecture
      printf("Vous avez introduit : '%s'.\n\n", str);

while (true);
```

Cette fois-ci, on aura pas de problème et à chaque itération on peut faire une lecture (voir le code suivant : https://onlinegdb.com/eSrJ2xkpT).

V.2.4. Manipulation des chaînes de caractères : Fonctions C

Langage C nous propose un ensemble de fonctions nous permettant de manipuler les chaînes de caractères. Parmi ces fonctions, nous avons : sprintf (dans la bibliothèque : stdio.h), strlen, strcpy, strcat, strcmp, strchr, strpbrk, strstr (dans la bibliothèques : string.h).

a- sprintf

Cette fonction, qui se trouve dans le ficher entête : stdio.h, ressemble à la fonction printf : au lieu d'écrire sur l'écran, elle écrit sur une chaîne de caractère (s dans sprintf pour string). Ceci nous facilite de créer une chaîne de caractère en combinant plusieurs variables avec leurs formats. Par exemple :

Langage C
```
sprintf(str, "Votre nom est : %s et votre age est : %d ans.", nom, age);
```

Si nom = "Mohammed" et age=19, on aura la valeur str :

```
Votre nom est : Mohammed et votre age est : 19 ans.
```

Voir l'exemple sur le lien suivant : https://onlinegdb.com/AaaRsk4kE

b- strcpy

Cette fonction, qui se trouve dans le ficher entête : string.h, permet de copier une chaîne de caractère (source) vers une autre chaîne de caractère (destination). La syntaxe de cette fonction est comme suit :

Langage C
```
stcpy(str_dest, str_src);
```

Par exemple : `stcpy(str, "Algorithmique ... Programmation");` permet de copier le deuxième paramètre vers le premier paramètre. (Voir : https://onlinegdb.com/Fkgla8CI7)

c- strcat

Cette fonction, qui se trouve dans le ficher entête : string.h, permet de concaténer deux chaînes de caractères. La syntaxe :

Langage C
```
stcat(str_1, str_2);
```

Cette syntaxe permet concaténer str_1 et str_2 et mettre le résultat dans str1. Voir l'exemple suivant : https://onlinegdb.com/Npc--VF5BR

d- strlen

Cette fonction, qui se trouve dans le ficher entête : string.h, permet de retourner la longueur (nombre de caractères) d'une chaîne de caractère.

Voir l'exemple sur le lien : https://onlinegdb.com/q2RMVPOep

e- strcmp

Cette fonction permet de comparer deux chaînes de caractères, et elle retourne un entier : 0 pour deux chaînes identiques et différent de 0 si elle sont différente :

```
char str1[50], str2[50];
do{
      printf("Donner la 1ère chaîne de caractères : ");
      scanf("%[^\n]", str1);

      printf("Donner la 2ème chaîne de caractères : ");
      scanf("%[^\n]", str2);

      if (strcmp(str1, str2) == 0)
      {
            printf("Les deux chaînes sont identiques.");
      }
      else
      {
            printf("Les deux chaînes sont différentes.");
      }

while (true);
```

Pour les autres fonctions de chaîne de caractères (et aussi d'autres thèmes sur les chaînes de caractères), vous pouvez consulter cette page web :

https://yard.onl/sitelycee/cours/c/Fonctionsdemanipulationdeschanes.html

V.3. Enregistrement : Structure

Une structure (ou enregistrement) est un regroupement de plusieurs variables (objets), éventuellement de types différents, afin de représenter un objet complexe du monde réel : Voiture, Vehicule, Filière, Etudiant, Matière, … Ça permet de simplifier la programmation et de rendre les programmes de plus haut niveau.

Les enregistrements (Structures) sont lié aux fichiers (structurées) pour la sauvegarde de données dans les mémoires secondaires.

V.3.1. Déclaration d'un enregistrement

Pour utiliser les enregistrements, une bonne pratique est de déclarer un nouveau type de l'enregistrement, et par la suite, déclarer des variables de ce nouveau type. Pour réaliser ça, nous suivons la syntaxique suivante :

ALGORITHME

```
Type
<type_enreg> = Enregistrement
   <champ_1> : <type_1>;
   <champ_2> : <type_2>;
   .......
   <champ_n> : <type_n>;
Fin;

Variable
  <var_enreg> : <type_enreg>;
```

LANGAGE C

```
typedef struct
{
   <type_1> <champ_1>;
   <type_2> <champ_2>;
   .......
   <type_n> <champ_n>;
} <type_enreg>;


<type_enreg> <var_obj> ;
```

Donc, il faut tout d'abord déclarer le type enregistrement, en définissant ses champs (attributs ou propriétés), par la suite utiliser ce nouveau type pour créer des variables. Pour bien illustrer ça, voici un exemple de déclaration de types TVoiture définie par son matricule, marque, modèle, puissance, couleur :

ALGORITHME

```
Type
TVoiture = Enregistrement
   matricule : chaîne[25];
   marque : chaîne[50];
   modele : chaîne[30];
   puissance : entier;
   couleur : chaîne[20];
Fin;

Variable
  V1, V2 : TVoiture;
```

LANGAGE C

```
typedef struct
{
   char matricule[25];
   char marque[50];
   char modele[30];
   int puissance;
   char couleur[20];
} TVoiture;


TVoiture V1, V2;
```

Pour cet exemple de déclaration, vous pouvez consulter ce lien : https://onlinegdb.com/nc9yIEYIb

V.3.2. Lecture & initialisation de variable enregistrement

Une variable de type enregistrement peut être lue, par clavier, en lisant tous ou une partie de ses champs (attributs), et on peut aussi initialiser ses champs.

a- Lecture

On lit une variable enregistrement, en lisant tous ces champs. Ceci se fait en indiquant la variable de la structure ainsi que le nom du champs en les séparant par un point, comme indiqué ci-dessous :

ALGORITHME

```
Écrire ("Informations du véhicule :");
Écrire("  Matricule : ");
Lire(V1.matricule);
Écrire("  Marque : ");
Lire(V1.marque);
Écrire("  Nombre de chevaux: ");
Lire(V1.puissnce);
```

LANGAGE C

```
printf("Informations du véhicule : \n");
printf("  Matricule : ");
scanf("%s", V1.matricule);
printf("  Marque : ");
scanf("%s", V1.marque);
printf("  Nombre de chevaux : ");
scanf("%d", &V1.chevaux);
```

b- Initialisation

On peut initialiser un objet de type enregistrement de deux façon différente : initialisation séquentielle ou initialisation sélective.

- Initialisation séquentielle

Permet d'initialiser une partie ou la totalité des attributs (champs) d'un enregistrement, à condition de respecter l'ordre de déclaration des champs, comme par exemple, l'enregistrement TVoiture défini précédemment :

LANGAGE C

```
TVoiture voiture1 = {"12544-06-19", "Volkswagen", "Golf", 220, "Rouge"};
TVoiture voiture2 = {"08043-06-14", "Volkswagen", "Polo"};
```

- Initialisation sélective

Permet d'initialiser les champs sélectivement en indiquant leur nom, sans, obligatoirement, respecter l'ordre des déclarations des champs :

LANGAGE C

```
TVoiture voiture3 = {.modele="ID.3", .puissance=310, .couleur="Grise"};
TVoiture voiture4 = {.marque="Peugeot", .matricule="7854-16-16"};
```

Le programme sur le lien suivant : https://onlinegdb.com/vcKwsuPBG illustre les deux initialisation vue précédemment.

V.3.3. Affichage des enregistrements

Comme la lecture des enregistrement, l'affichage de ce type d'objet se fait par l'écriture champs par champs de l'enregistrement :

ALGORITHME

```
Écrire(<var_enreg>.<champ_1>);
Écrire(<var_enreg>.<champ_2>);
.......
Écrire(<var_enreg>.<champ_n>);
```

LANGAGE C

```
printf("%ftc", <var_enreg>.<champ_1>);
printf("%ftc", <var_enreg>.<champ_2>);
........
printf("%ftc", <var_enreg>.<champ_n>);
```

Tel-que : `%ft` est le format du champs correspondant de l'enregistrement.

V.3.4. Modéliser, enregistrements & tableaux

Souvent, pour représenter un monde réel, on utilise les enregistrement afin de modéliser un concept (concret ou abstrait). Par exemple, on veut réaliser un programme pour dessiner des formes géométriques, donc, nous aurons besoins des structures (enregistrements) suivantes : TPoint , TDroite, TRectangle , TCarree, TCercle , TEllipse , … tel-que chaque type d'objet possède des champs qui le caractérisent. On sait que TPoint possède deux champs : x et y. Et TDroit possède deux points (deux champs de type TPoint), … *etc.*

En combinant ces structures avec les tableaux, on peut créer une liste d'objets : liste de lignes droites, liste de rectangles, listes des points, … Et comme vous le saviez, dans le *chapitre IV*, les tableaux sont statiques : On réserve un espace fixe (et important) au début du programme : Ce qui représente les structures de données statiques.

Voir ce petit programme qui indique comment déclarer ces structures : https://onlinegdb.com/kJBI7wBKJI