IUT "A" Paul SABATIER
Dpt Informatique

AP => Compléments du langage C : Pgc2

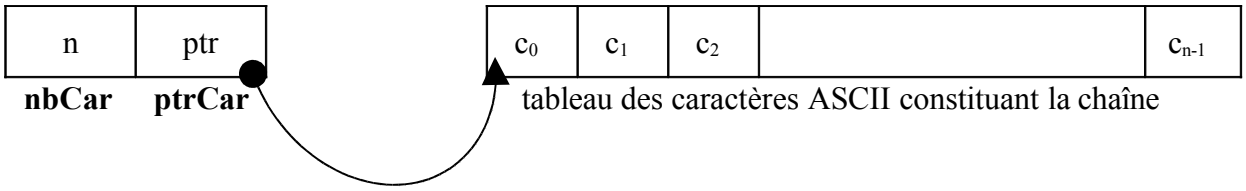
TP : chaînes de caractères dynamiques

La manipulation des chaînes de caractères en langage C est une source d'erreurs importante lors de l'exécution de programmes. Ceci est dû au mode de représentation interne de ces chaînes et aux sous-programmes de traitement proposés par la bibliothèque standard du langage :

- les chaînes dont la taille peut varier, sont mémorisées dans des tableaux d'octets **statiques** dont la taille est fixée au moment de la compilation.
- les sous-programmes de traitement des chaînes ne contrôlent pas les **débordements** de tableaux qui peuvent se produire lors de l'exécution du programme.

On se propose de définir une nouvelle représentation des chaînes de caractères et une bibliothèque de sous-programmes de traitement qui ne présentent pas les inconvénients cités ci-dessus.

Une chaîne de caractères sera représentée par un **enregistrement statique** et par **un tableau dynamique de caractères** dont le modèle est le suivant :



- Le champ **nbCar** contient un entier naturel binaire, égal au nombre de caractères de la chaîne.

- Le champ **ptrCar** contient l'adresse binaire du tableau de caractères où est stockée la suite de caractères ASCII constituant la chaîne.

  Dans une mise en œuvre en langage C, seule l'allocation en zone **mémoire dynamique** du tableau de caractères, permet de faire varier sa taille, pour l'adapter à la taille de la chaîne qu'il mémorise. Les variables dynamiques créées pendant l'exécution du programme n'ont pas de nom, puisqu'elles ne sont pas déclarées dans le programme. Elles ne sont accessibles que par le biais d'une variable pointeur qui contiendra leur adresse.

A cette représentation, on associera un type et une bibliothèque de sous-programmes de traitement, dont la spécification est la suivante :

```
typedef  char *  PointeurCar ;

typedef  struct
{
  int  nbCar ;                    /* nombre de caractères de la chaîne */
  PointeurCar  ptrCar ;           /* pointeur sur le tableau de caractères contenant la suite de
                                     caractères ASCII constituant la chaîne */
  } ChaineDyn ;
```

Remarques :

- Cette bibliothèque de sous-programmes sera programmée en C avec le compilateur **gcc** de GNU sous **Linux.**
- Le traitement des exceptions dans les S/P concernés sera mis en œuvre par la technique des points de reprises.

1)- /* Cette procédure initialise *ch* à chaîne vide. Elle devra **obligatoirement** être appelée pour toutes les variables du type **ChaineDyn** déclarées dans le programme C, avant leur première utilisation. */

void creerChaineVide (**ChaineDyn** * *ch*) ;

2)- /* Cette procédure affecte les caractères de *ch2* qui est une chaîne du langage C, à la chaîne *ch1* qui est une chaîne de type **ChaineDyn.**
Elle lève l'exception ″ERR_ALLOC″ en cas d'erreur d'allocation dynamique.
Si une exception est levée la chaîne *ch1* ne sera pas modifiée. */

void convertirChaine (**ChaineDyn** * *ch1* , **const char** * *ch2* , **jmp_buf** *ptRep*) ;

3)- /* Cette fonction retourne un entier égal à la longueur de la chaîne *ch*. */

int longueurChaine (**const ChaineDyn** *ch*) ;

4)- /* Cette procédure affecte la chaîne *ch2* à la chaîne *ch1*.
Elle lève l'exception ″ERR_ALLOC″ en cas d'erreur d'allocation.
Si une exception est levée la chaîne *ch1* ne sera pas modifiée. */

void copierChaine (**ChaineDyn** * *ch1*, **const ChaineDyn** *ch2* , **jmp_buf** *ptRep*) ;

5)- /* Cette procédure affecte les *n* premiers caractères (**au plus**) de la chaîne *ch2* à la chaîne *ch1*.
Elle lève l'exception ″ERR_PARAM″ si $n < 0$.
Elle lève l'exception ″ERR_ALLOC″ en cas d'erreur d'allocation dynamique.
Si une exception est levée la chaîne *ch1* ne sera pas modifiée. */

void copierSouschaine (**ChaineDyn** * *ch1*, **const ChaineDyn** *ch2* , **const int** *n* ,
jmp_buf *ptRep*) ;

6)- /* Cette procédure concatène la chaîne *ch2* à la chaîne *ch1*.
Elle lève l'exception ″ERR_ALLOC″ en cas d'erreur d'allocation dynamique.
Si une exception est levée la chaîne *ch1* ne sera pas modifiée. */

void collerChaine (**ChaineDyn** * *ch1*, **const ChaineDyn** *ch2* , **jmp_buf** *ptRep*) ;

7)- /* Cette procédure concatène les *n* premiers caractères (**au plus**) de la chaîne *ch2* à la chaîne *ch1*.
Elle lève l'exception ″ERR_PARAM″ si $n < 0$.
Elle lève l'exception ″ERR_ALLOC″ en cas d'erreur d'allocation dynamique.
Si une exception est levée la chaîne *ch1* ne sera pas modifiée. */

void collerSouschaine (**ChaineDyn** * *ch1*, **const ChaineDyn** *ch2* , **const int** *n* ,
jmp_buf *ptRep*) ;

8)- /* Cette procédure extrait de la chaîne *ch2*, la sous-chaîne de *n* caractères (**au plus**) commençant à la position *p* et l'affecte à la chaîne *ch1*.
Elle lève l'exception ″ERR_PARAM″ si $p \leq 0$ ou $p >$ longueur(*ch2*) ou $n < 0$.
Elle lève l'exception ″ERR_ALLOC″ en cas d'erreur d'allocation dynamique.
Si une exception est levée la chaîne *ch1* ne sera pas modifiée. */

void extraireSousChaine (**ChaineDyn** * *ch1*, **const ChaineDyn** *ch2*,
const int *p*, **const int** *n* , **jmp_buf** *ptRep*) ;

9)- /* Cette fonction effectue la comparaison **lexicograhique** des 2 chaînes et retourne :

0 si *ch1* = *ch2*
+1 si *ch1* > *ch2*
- 1 si *ch1* < *ch2* */

int comparerChaine (**const ChaineDyn** *ch1*, **const ChaineDyn** *ch2*) ;

10)- /* Cette fonction effectue la comparaison **lexicograhique** des *n* premiers caractères (**au plus**) des 2 chaînes et retourne les mêmes résultats que comparerChaine.
Elle lève l'exception ″ERR_PARAM″ si $n < 0$. */

int comparerSousChaine (**const ChaineDyn** *ch1*, **const ChaineDyn** *ch2*, **const int** *n* ,
jmp_buf *ptRep*) ;

11)- /* Cette procédure lit en séquence des caractères dans le fichier *f* à la position courante et les affecte à la chaîne *ch*.
La lecture s'arrête dès que *n* caractères sont lus ou que le caractère fin de ligne (**\n**) est rencontré ou que la fin de fichier est rencontrée.
L'ouverture du fichier devra avoir été faite dans le programme appelant.
Elle lève l'exception ″ERR_PARAM″ si $n < 0$.
Elle lève l'exception ″ERR_FIC″ en cas d'erreur de lecture du fichier
Elle lève l'exception ″ERR_ALLOC″ en cas d'erreur d'allocation dynamique.
Si une exception est levée la chaîne *ch* ne sera pas modifiée. */

void entrerChaine (**ChaineDyn** * *ch*, **FILE** * *f*, **const int** *n*, **jmp_buf** *ptRep*) ;

12)- /* Cette procédure écrit en séquence les caractères de la chaîne *ch* dans le fichier *f* à la position courante.
L'ouverture du fichier devra avoir été faite dans le programme appelant.
Elle lève l'exception ″ERR_FIC″ en cas d'erreur d'écriture du fichier. */

void sortirChaine (**const ChaineDyn** *ch*, **FILE** * *f* , **jmp_buf** *ptRep*) ;

13)- /* Cette procédure libère le bloc d'octets alloué à la chaîne *ch* en mémoire dynamique et réinitialise *ch* en chaîne vide. */

void liberer Chaine(**ChaineDyn** * *ch) ;*

TRAVAIL DEMANDÉ

1ère partie: exercice préliminaire

Le répertoire **/usr/local/public/Pgc2** contient les fichiers suivants :

- **creerChaineVide.c**, **convertirChaine.c**, **longueurChaine.c**, **copierChaine.c** et **libererChaine.c** dans lesquels sont codés en C, les S/P (sous-programmes) correspondants.
- **chaine.h** dans lequel sont codées en C, les déclarations (prototypes) de ces S/P et du type <ChaineDyn>.
- **testChaine.c** dans lequel est codé en C, un programme de test des S/P ci-dessus.
- **makefile** contenant les directives pour générer automatiquement l'application avec l'utilitaire **make**.

1°) Se connecter sur le poste de travail sous Linux.
Créer un sous-répertoire **Pgc2** et copier dans ce sous-répertoire tous les fichiers catalogués dans le répertoire **/usr/local/public/Pgc2.**

2°) Après avoir analysé le fichier **makefile**, créer le programme exécutable **testChaine** avec l'utilitaire **make.**

3°) Exécuter le programme **testChaine** et vérifier le bon fonctionnement des S/P qui y sont testés.

2ème partie: programmation

Reprendre l'application fournie en exemple dans la 1ère partie, et la compléter en y intégrant le développement et le test des S/P suivants :

- **sortirChaine**
- **entrerChaine**
- **extraireSousChaine**
- **comparerChaine**

Contrôle des programmes: pendant les séances de T.P.