TD : autres chaînes de caractères

Objectif(s)

- ✯ comprendre et maîtriser les chaines de caractères
- ✯ première initiation à l'utilisation d'une pile.

Exercices obligatoires

Exercice 1 – les chaines de caractères

Question 1

Écrivez une fonction `invertstring` qui prend une chaîne de caractères en paramètre et l'inverse (en place).

Question 2

Écrivez une fonction `substring` qui prend deux chaîne de caractères `src` et `dest` ainsi qu'un indice de début d et un indice de fin f. La fonction écrit dans la chaine de caractères `dest` la sous-chaîne de d (inclu) à f (exclu) de `src`. On n'oubliera pas de terminer la chaîne par `'\0'`. Elle retourne la taille de la chaîne `dest` obtenue.

Question 3

Écrivez une fonction `strcmp` qui compare deux chaînes de caractères en minuscule : elle retourne -1 si la première chaîne passée en paramètre est lexicographiquement plus petite que la deuxième, 1 si la premère est plus grande, 0 si les deux sont égales.

Exercice 2

L'objectif de cet exercice est de vérifier qu'une expression mathématique est correctement parenthésée.
Les expressions $((a+b)*(c+d))$, $((a)(()))$ et $()$ sont correctement parenthésées.
Les expressions suivantes ne le sont pas : $((a+b)*(c+d)$, $(a+b))$, $)($.
Soit une chaine de caractères contenant une expression arithmétique.

Question 1

Une première technique consiste à utiliser un compteur que l'on incrémente de 1 lorsque l'on rencontre une (et que l'on décrémente de 1 lorsque l'on rencontre une). Si en parcourant la chaine de caractères le compteur devient inférieur à 0 ou si à la fin du parcours le compteur est différent de 0, il y a une erreur de parenthésage.

Écrivez une fonction `verifie_expression_1` qui renvoie `true` si une expression est correctement paranthèsée, `false` sinon.

Question 2

Une autre technique consiste à utiliser une pile. Lorsque l'on rencontre une (on la met dans la pile, et lorsque l'on rencontre une) on dépile. Si en parcourant la chaine de caractères, on ne peut pas dépiler car la pile est vide ou si à la fin du parcours la pile n'est pas vide, il y a une erreur de parenthésage.

Pour gérer une pile, on dispose d'un type `pile_s` et de 4 fonctions :

- `init(pile_s *p)` : initialise à zero la pile p.
- `void empile(pile_s *p, char e)` : dépose au sommet de la pile l'élément e.
- `char depile(pile_s *p)` : retire et retourne l'élément du sommet de la pile.
- `int est_vide(pile_s p)` : renvoie vrai si la pile est vide sinon faux.

Écrivez une fonction `verifie_expression_2` qui renvoie `true` si une expression est correctement paranthèsée, `false` sinon.

Question 3

Nous allons maintenant complexifier un peu le problème en ajoutant des [et des] dans les expressions.

Les expressions $[(a+b)*[(c+d)]]$ et $((a)([])$ sont correctement parenthésées. Les expressions suivantes ne le sont pas : $[(a+b)*(c+d)$, $(a+b)]$, $([(])$.

Quelle technique permet de vérifier une expression avec des parenthèses et des crochets ?

Écrivez une fonction `verifie_expression_3` qui renvoie `true` si une expression est correcte avec des parenthèses et des crochets, `false` sinon.

Renforcement

Exercice 3

Question 1

Écrivez une fonction `strcpy` qui copie le contenu de la deuxième chaîne de caractères (nommé `src`), passée par adresse, dans la première (nommé `dest`), aussi passée par adresse.

Question 2

Écrivez une fonction `match` qui prend deux chaînes de caractères en parametre et retourne vrai si la première est contenue dans la deuxième. Combien de comparaison de letttres avez vous fait ?