INFO505 - TP1 – 2024-2025

Exercice I : Pointeurs et chaine de caractères

Sans utiliser les bibliothèques standard string.h et ctype.h, il vous est demandé de réécrire certaines des fonctions standard.

- int printchar(const char *str)
- size_t stringLength (const char * str) ;
- int stringCompare (const char * str1 , const char * str2) ; // compare les chaînes C str1 et str2. Elle renvoie un nombre négatif, 0, ou un nombre positif si str1 apparaît avant, est égal à, ou apparaît après str2 dans l'ordre lexicographique.
- const char * findFirst (const char * str , char c) qui renvoient un pointeur sur la première occurrence d'un caractère donné c dans la chaîne str ou NULL si c n'apparaît pas
- **(optionel)** const char * searchString (const char * haystack , const char * needle , int csensitive) ; // qui renvoie un pointeur sur la première occurrence de la chaîne C needle dans la chaîne C haystack, ou NULL si la sous-chaîne n'est pas trouvée.

Exercice II : Vigenère cipher

Le « chiffre de Vigenère » est une méthode de chiffrement et de déchiffrement d'un texte alphabétique, appelé message, à l'aide d'un autre texte alphabétique, appelé clé. Il fonctionne de la manière suivante :

1. Tout d'abord, chaque lettre du message est associée à une lettre de la clé. La clé est répétée si nécessaire. Les caractères spéciaux sont ignorés. Par exemple, pour crypter le message « L'escargot se promène avec sa maison » en utilisant la clé “perdu”, on associe les lettres comme ci-dessous.

2. Le message crypté est ensuite dérivé de la table dite de Vigenère (tableau 1). Chaque paire de lettres donne une lettre du message crypté. Ligne L, colonne P, nous avons un A ; ligne E, colonne E, nous avons un I, et ainsi de suite. Ainsi, le message crypté est « a'ijfugkfm my tirgtrv dptg jd gpmjrh ».

L’ESCARGOT SE PROMENE AVEC SA MAISON
P ERDUPERD UP ERDUPER DUPE RD UPERDU
> A’IJFUGKFW MT TIRGTRV DPTG JD GPMJRH

3. Bien entendu, le processus inverse permet de décrypter le message. Dans la colonne P, la lettre A est à la ligne L ; dans la colonne E, la lettre I à la ligne E ; etc.

A’IJFUGKFW MT TIRGTRV DPTG JD GPMJRH
P ERDUPERD UP ERDUPER DUPE RD UPERDU
> L’ESCARGOT SE PROMENE AVEC SA MAISON

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
B	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a
C	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b
D	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c
E	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d
F	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
G	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f
H	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g
I	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h
J	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i
K	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j
L	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k
M	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l
N	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m
O	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n
P	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
Q	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
R	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
S	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r
T	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s
U	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t
V	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u
W	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
X	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
Y	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x
Z	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y

Figure 1 : Table de vigenere

Complétez le programme fourni plus bas de façon à ce que :

(a) la table de Vigenère soit initialisée correctement après avoir été définie ;

(b) le message soit crypté, puis décrypté, en utilisant la clé fournie.

Vérifiez qu'il est correctement transformé, puis qu'il revient à son contenu d'origine.

```
int main () {
        char message [] = « L'escargot ␣se␣ promene ␣ avec ␣sa␣ maison » ;
        const char * key = « perdu » ;
        char vigenereTable1 [26][26] ;
        ...
        encrypt_1 ( message , key , vigenereTable1 ) ;
        printf ( » encrypt1 :␣%s\n », message ) ;
        decrypt_1 ( message , key , vigenereTable1 ) ;
        printf («  decrypt1 :␣%s\n », message ) ;
        return 0 ;
}
```

2. Même question, sauf que la table de Vigenère doit maintenant être allouée dynamiquement.

```
int main () {
        char message [] = « L'escargot ␣se␣ promene ␣ avec ␣sa␣ maison » ;
        const char * key = « perdu » ;
        char ** vigenereTable2 = NULL ;
        ...
        encrypt_2 ( message , key , vigenereTable2 ) ;
        printf ( » encrypt2 :␣%s\n », message ) ;
        decrypt_2 ( message , key , vigenereTable2 ) ;
        printf («  decrypt2 :␣%s\n », message ) ;
        ...
        retour 0 ;
}
```

3. Essayez maintenant de crypter et de décrypter le message donné sans initialiser et stocker explicitement la table

```
int main () {
        char message [] = « L'escargot ␣se␣ promene ␣ avec ␣sa␣ maison » ; const char * key = « perdu » ;
        encrypt_3 ( message , key ) ;
        printf ( » encrypt3 :␣%s\n », message ) ;
        decrypt_3 ( message , key ) ;
        printf («  decrypt3 :␣%s\n », message ) ;
        retour 0 ;
}
```

Exercice III : jeu du pendu et autres exercices du TD1 et TD2