Universtité Nice Sophia Antipolis
Polytech'Nice Sophia - SI 3

2016 – 2017

NOM: BEROUK
PRÉNOM: Zaki
GROUPE: 4

Programmation Procédurale

18 octobre 2016

Documents non autorisés. **Durée: 1h30**

Vous pouvez omettre les directives `#include` *dans vos réponses. Par ailleurs, vous apporterez un très grand soin à* ***la présentation*** *car elle* ***interviendra dans la notation.***

Question 1:

Ecrire la fonction C:

```
int strspn(const char s[ ], const char accept[ ]);
```

Le résultat de cette fonction est la longueur de la plus longue chaîne préfixe de `s` formée uniquement de caractères de `accept`.

Pour simplifier l'écriture de cette fonction, vous pouvez vous définir auparavant la fonction `isin(s, c)` qui vous indiquera si le caractère `c` apparaît dans la chaîne `s`.

Exemples:

```
strspn("essai", "es")     → 3
strspn("essai", "se")     → 3
strspn("essai", "asze")   → 4
strspn("essai", "xyz")    → 0
```

```
int strspn (const char s[], const char accept[]){
    int cpt=0;
    int tmp=0;
//  int k=0;
    int i=0;
    while (s[i] != '\0'){
        for( int j=0, int k=i; accept[j] != '\0' ;j++){
            if( s[k] == accept[j]){
                tmp++;
                k++;
            }
        }
        if (tmp > cpt){
            cpt = tmp;
        }
        i++;
    }
    return cpt;
}
```

Question 2:

Écrire la fonction `imprime` qui permet d'imprimer une série de paramètres de longueur impaire. Les paramètres d'indice impair sont des chaînes de caractères et les paramètres d'indice pair sont de type entier. La série de paramètres se termine par la valeur `NULL`. Cette fonction permet d'imprimer une série d'entiers où chaque entier est précédé par une chaîne le décrivant. Ainsi par exemple l'appel

```
imprimer("Valeur de x=", x, " Valeur de y=", y, " => x+y=", x+y, NULL);
```

affichera

```
Valeur de x=1 Valeur de y=2 => x+y=3
```

```
void imprimer(char text, int val,...){
    va_list ap;
    va_start(ap, text);
    printf("%s", va_start(ap, text));

    while (ap != NULL){
        printf("%d", va_list(ap, char));
        printf("%s", va_list(ap, int));
    }
    va_end(ap, int);
}
```

Question 3:

Écrire une version du programme `cat` (appelée `lower`) qui convertit en minuscule les caractères du fichier standard d'entrée sur le fichier standard de sortie. Un exemple d'utilisation de ce programme est donné ci-dessous:

```
$ cat < fichier
Un fichier avec
du TEXTE
$ lower < fichier
un fichier avec
du texte
$
```

Non ce c'est pas ce qui est demandé !

```
Void lower (char s){
    int c;
    int i=0;
    while ((c=getchar()) != EOF){
        if (c >= 'A' && c <= 'Z'){
            s[i] = c + ('A' - 'a');
        }
        i++;
    }
}
```

1 BP

où est l'appel ?

Main ?

Relisez le sujet !!

Question 4:

On a un programme constitué des fichiers suivants: `A.c`, `A.h`, `B.c`, `B.h`, `C.c`, `C.h` et `commun.h`. Le fichier `commun.h` contient des déclarations communes et est inclus par tous les fichiers `'.c'`. Par ailleurs, tous les fichiers `'.c'` incluent le `'.h'` qui leur correspond. Ce programme est ecrit en C99 et doit être compilé par gcc avec les options `-Wall -std=c99`.

Ecrire un fichier `Makefile` permettant de "gérer" ce programme. Vous pouvez/devez utiliser ici le plus possible les règles implicites connues de la commande **make**.

```
CCFLAGS (gcc -Wall -std=c99)
OBJ (A.c, A.h, B.c, B.h, C.c, C.h, commun.h)

commun $(OBJ)
    gcc $CCFLAGS -o commun.o -c $OBJ;

A.c : A.h, commun.h
    gcc $CCFLAGS -c A.c A.h commun.h

B.c : B.h, commun.h
    $CCFLAGS -c B.c B.h commun.h

C.c : C.h, commun.h
    $CCFLAGS -c C.c C.h commun.h

clean:
    rm -rf *.o  & + executable
```

1

Question 5:

Une chaîne de caractères est un *palindrome* si la succession des caractères qui la composent est la même quand on la parcourt de gauche à droite ou de droite à gauche. Un palindrome peut se composer d'un seul mot, comme "non" ou "kayak" ou ce peut être une phrase, comme "karine alla en irak" (il faut alors ignorer le caractère espace).

- Écrire la fonction palindrome qui vérifie renvoie 1 si le mot qui lui est passé en paramètre est un palindrome et 0 sinon (on ne gère pas les espaces ici)

```
int palindrome (char s1[]){
    char s2 = s1;
    int i=0;
    int j = strlen(s1);
    while ( s1[i] != '\0'){
        if (s1[i] != s2[j]){
            return 0;
        }
        i++;
        j--;
    }
    return 1;
}
```

?? 1 1/2 BP

Pourquoi utiliser une copie ? On peut tout faire avec la variable originale

- Écrire la fonction palindrome qui renvoie 1 si la phrase qui lui est passée en paramètre est un palindrome et 0 sinon (dans ce cas, on ignorera, les espaces).

```
int palindrome_v2 (char s1){
    char s2 = s1;
    int i=0;
    int j = strlen(s2);
    while (s1[i] != '\0'){
        if ( s1[i] == ' ' ){
            i++;
        }
        if (s2[j] == ' '){
            j--;
        }
        if (s1[i] != s2[j]){
            return 0;
        }
        i++;
        j--;
    }
    return 1;
}
```

3

2/3

Attention on peut être tous les deux