Programmation en C - TP 3

Exercice 1 (Affichage)

```c
#include <stdio.h>

int main(void) {

    char myvar = 'Z';

    printf("%d %x %c %p\n", myvar, myvar, myvar, &myvar);

    char diff = 'a'-'A';
    // la variable diff correspond à la diffé décimale entre le petit et le grand caractère

    char myvar_diff = myvar + diff;


    printf("%d %x %c %p\n", myvar_diff, myvar_diff, myvar_diff, &myvar_diff);

    return 0;

}
```

Affichage :

```
90 5a Z 000000147f3ffdae122
7a z 000000147f3ffdad
```

Exercice 2 (Alphabet)

```c
#include <stdio.h>

int main(void) {

// 1) Majusule puis minuscule
    // Majuscule
    printf("de A à Z : ");
    for(int i = 'A'; i <= 'Z'; i++) {
        printf("%c ", i);
    }
    printf("\n");
    // Minuscule
    printf("de a à z : ");
    for(int i = 'a'; i <= 'z'; i++) {
        printf("%c ", i);
    }
    printf("\n");

// 2) Majuscule puis minuscule en décimal
    // Majuscule
    printf("de A à Z : ");
    for(int i = 'A'; i <= 'Z'; i++) {
        printf("%d ", i);
    }
    printf("\n");
    // Minuscule
    printf("de a à z : ");
    for(int i = 'a'; i <= 'z'; i++) {
        printf("%d ", i);
    }
    printf("\n");

// 3) Majuscule puis minuscule en hexadécimal
    // Majuscule
    printf("de A à Z : ");
    for(int i = 'A'; i <= 'Z'; i++) {
        printf("%x ", i);
    }
    printf("\n");
    // Minuscule
    printf("de a à z : ");
    for(int i = 'a'; i <= 'z'; i++) {
        printf("%x ", i);
    }
    printf("\n");
    // Valeurs vérifiées et bonnes

// 4) ASCII de (32)10 à (126)10
```

```
    printf("de 32 à 126 : ");
    for(int i = 32; i <= 126; i++) {
        printf("%c ", i);
    }
    printf("\n");

    return 0;
}
```

Affichage :

```
de A á 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 Z
de a à 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 z
de A á Z : 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89
90
de a á z : 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115
116 117 118 119 120 121 122
de A á Z : 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e 4f 50 51 52 53 54 55 56 57 58 59
5a
de a á z : 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 77 78 79
7a
de 32 á 126 :   ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ 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 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 { | } ~
```

4. Nous ne démarrons pas la valeur à 0 puisque entre 0 et 32 se trouve des caractères de contrôle permettant d'effectuer certaines actions précise sur l'ordinateur

5. D'après le code ASCII, le caractère décimal 7 correspond à un caractère de contrôle qui permet de faire un son de cloche (une goutte sur Linux).

Exercice 3 (Minuscule)

```c
#include <stdio.h>
#include <string.h>

int main(void) {

    char s[] = "Bonjour, voici Une Chaine avec Des MaJusCules !!";
    const unsigned short taille = strlen(s)+1;
    char d[taille];

    strcpy(d, s);

    printf("Original: %s\n", s);

    int i = 0;
    while(d[i] != '\0') {
        if(d[i] <= 'Z' && d[i] >= 'A')
            d[i] += 32;
        i++;
    }

    printf("Minuscule: %s\n", d);



    return 0;
}
```

Affichage :

```
Original: Bonjour, voici Une Chaine avec Des MaJusCules !!
Minuscule: bonjour, voici une chaine avec des majuscules !!
```

Exercice 4 (String concat)

```c
#include <stdio.h>
#include <string.h>

int main(void) {

    char tab1[] = "ceci";
    char tab2[] = "est une phrase";
    char tab3[30];

    printf("%s %s\n", tab1, tab2);

    strcat(tab3, tab1);
    strcat(tab3, " ");
    strcat(tab3, tab2);
    strcat(tab3, "\0");

    printf("%s\n", tab3);

    return 0;
}
```

Affichage :

```
ceci est une phrase
ceci est une phrase
```

2. L'affichage de `tab3` au moment de l'exécution du programme peut poser problème car lorsque nous souhaitons l'afficher, il y a de fortes chance que les différentes adresses entre tab[0] et tab[29] ne possèdent pas le caractère nul (`\0`) et en conséquence, va afficher la mémoire de l'ordinateur (`Buffer Overflow`)

3. Il n'est pas possible de concatener les valeurs de deux tableau en C car chaque tableau est représenté par un **pointeur** et une **taille**, or le programme ne connait pas directement la taille du tableau. La concaténation par l'opérateur `+` n'est donc pas possible.

Exercice 5 (String copy)

```c
#include <stdio.h>
#include <string.h>

int main(void) {

    char tab1[] = "ceci";
    char tab2[] = "est une phrase";
    char tab3[30];

    printf("%s %s\n", tab1, tab2);

    strcpy(tab3, strcat(tab1, tab2));
    strcat(tab3, "\0");

    printf("%s\n", tab3);
    return 0;
}
```

Affichage :

```
ceci est une phrase
ceciest une phrase
```

Exercice 6 (Select sort)

```c
#include <stdio.h>
#include <limits.h>

int main(void) {

    int tab[] = {10,4,45,453,2,4356,23,98,23,1};

    const unsigned long size = sizeof(tab)/sizeof(tab[0]);

    printf("Tableau non trié:\n[");
    for(int i = 0; i < size; i++) {
        printf((i == size-1) ? "%d" : "%d, ", tab[i]);
    }
    printf("]\n");

    printf("Tableau trié:\n[");
    for(int i = 0; i < size; i++) {
        int m = tab[i], a = i;
        for(int j = i; j < size; j++) {
            if(m > tab[j]) {
                m = tab[j];
                a = j;
            }
        }

        int v = tab[i];
        tab[i] = m;
        tab[a] = v;
        printf((i == size-1) ? "%d" : "%d, ", m);
    }
    printf("]\n");

    return 0;
}
```

Affichage :

```
Tableau non trié:
[10, 4, 45, 453, 2, 4356, 23, 98, 23, 1]
Tableau trié:
[1, 2, 4, 10, 23, 23, 45, 98, 453, 4356]
```