

Corrigé du TD n°2

Exercice 1

Les dessins de l'état de la mémoire ont été faits en TD.

Exercice 2

On peut remarquer que les cases vers lesquelles `t` pointe sont consécutives, donc il s'agit d'un tableau. On peut aussi remarquer que la dernière de ces cases a une valeur non déterminée.

Le tableau peut être dynamique ou statique, on va ici faire un tableau dynamique.

```
int* t = malloc(3*sizeof(int));
t[0] = 5;
t[1] = 4;
int x = 12;
int* q = &x;
int** p = &q;
```

Exercice 3

On propose le programme suivant pour échanger les valeurs de deux variables :

```
void swap(int x, int y){
    int tmp = x;
    x = y;
    y = tmp;
}

int main(){
    int x = 3;
    int y = 5;
    swap(x,y);
}
```

1. C fonctionne par passage par valeurs donc les variables `x` et `y` dans la fonction `swap` n'ont rien à voir avec les variables `x` et `y` dans le `main`, à part qu'elles ont initialement la même valeur.

```
2. void swap_bis(int* a, int* b){
    int tmp = *a;
    *a = *b;
    *b = tmp;
}
```

```
3. void minimax(int* a, int* b){
    if (*a>*b){swap_bis(a,b);}
}
```

Exercice 4

On considère la fonction suivante :

```
void mystere(int* x, int* y) {
    *x = *x - *y;
    *y = *x + *y;
    *x = *y - *x;
}
```

1. On conjecture que la fonction échange les valeurs des variables `*x` et `*y`. Pour faire la preuve complète d'un coup, on la fera après la question 4
2. On étudie les valeurs successives des `x` et `y`, qu'on indexe selon la ligne.

On a alors : $x_0 = 3, y_0 = 3; x_1 = x_0 - y_0 = 0, y_1 = 0$ car $x = y$ entraîne $*y = *x; x_2 = 0, y_2 = x_1 + y_1 = 0$ et $x_3 = y_2 - x_2 = 0, y_3 = 0$.

Ici, le fait d'avoir $x = y$ casse notre conjecture, puisque modifier l'un modifie l'autre.

En fait la fonction échange les valeurs de `*x` et `*y` si x et y ne pointent pas au même endroit et met 0 dans `*x` et `*y` sinon.

En effet dans le premier cas on a successivement : $x_1 = x_0 - y_0, y_1 = y_0; x_2 = x_0 - y_0, y_2 = x_1 + y_1 = x_0$ et $x_3 = y_2 - x_2 = x_0 - (x_0 - y_0) = y_0, y_3 = x_0$.

Dans le deuxième cas on a $x_1 = x_0 - y_0 = 0, y_1 = 0$ car $x = y$ entraîne $*y = *x; x_2 = 0, y_2 = x_1 + y_1 = 0$ et $x_3 = y_2 - x_2 = 0, y_3 = 0$.

Exercice 5

1.

```
int div_euclidienne(int a, int b, int* r){
    *r = a%b; //Reste de la division euclidienne
    return a/b; //Renvoi du quotient
}
```

2. Plusieurs choix : on peut renvoyer un tableau de deux éléments, ou renvoyer le nombre de fois et mettre l'indice de première occurrence dans un pointeur.

```
int occurrences(int* t, int n, int x, int* premiere_occ){
    int nb = 0;
    *premiere_occ = -1; //-1 indique qu'on a jamais rencontré d'occurrence de x
    for(int i=0; i<n;i+=1){
        if(t[i]==x){
            if (*premiere_occ==-1){*premiere_occ = i;}
            nb+=1;
        }
    }
    return nb;
}
```