Les tableaux statiques et dynamiques

1. Existe-t-il une différence entre les lignes ci-dessous :

```
int *tab1 = malloc(3 * sizeof(int));
int tab2[3] ;
```

2. Ecrire le code permettant de placer dans tab1 et tab2 les valeurs 1,2,3.
3. Ecrire le code de la fonction init_tab(int t[], int taille) qui initialise le tableau t avec les valeurs 1,2,..., jusqu'à taille -1. Puis écrire le code du main permettant d'appeler cette fonction pour tab1 et tab2.

Les fonctions avec pointeurs

4. Ecrire le code la fonction division qui prend 2 valeurs entières et doit renvoyer le quotient et le reste de la division, puis écrire l'appel de la fonction.
5. Ecrire le code de la fonction resolution_eq qui prend 3 paramètres a,b,c ($ax^2+bx+c=0$) d'une équation du second degré et doit renvoyer x1,x2 et un code permettant de préciser si l'équation possède des solutions ou pas.
6. Ecrire le code de la fonction

Les chaines de caractères

7. Ecrire le code de la fonction mirroir(char* dst, char *src) qui prend la chaine src et la recopie dans dst mais à l'envers.
8. Ecrire le code de la fonction maj(char* dst, char *src) qui prend la chaine src et la recopie dans dst mais en majuscules (attention il faudra tester chaque caractères de la chaine avant de le mettre en majuscule puisque 0 & W ne doivent pas être mis en majuscule)
9.
10. QAppeler la fonction produit dans le main :

```
int produit(int x, int y) {
    return (x * y);
}
```

11. Appeler la fonction produit1 dans le main :

```
int produit1(int x, int y, int* erreur) {
    if (x * y > 1000)
        *erreur = -1;
    else
        *erreur = 0;
    return (x * y);
}
```

12. Appeler la fonction Somm1 dans le main, on crée un tableau de 4 éléments dans le main.

```
int Somme1(int tab[], int nb) {
    int somme = 0;
    tab[2] = 10;
    for (int i = 0; i < nb; i++)
        somme += tab[i];//+tab[i]
    return somme;
}
```

13. Appeler la fonction Somm1 dans le main, on crée un tableau de 4 éléments dans le main.

```
int Somme2(int *tab, int nb) {
```

```
    int somme = 0;
    tab[2] = 10;
    for (int i = 0; i < nb; i++)
        somme += *(tab+i);//+tab[i]
    return somme;
}
```

14. Que fait le programme ci-dessous

```
#include <stdio.h>

int main() {
    int a = 5;
    int *p = &a;

    printf("%d\n", *p);
    printf("%x\n", p);
    return 0;
}
```

15. Que fait le programme ci-dessous

```
void f(int x) {
    x = 10;
}
main(){
      int x=2;
      f(x);
      printf("%d",x);
}
```

16. Que fait le programme ci-dessous

```
void f(int *x) {
    *x = 10;
}
main(){
      int x=2;
      f(&x);
      printf("%d",x);
}
```

17. Que fait le programme ci-dessous

```
int f(int x){
      return x*2;
}
main(){
      int x=2;
      x=f(x);
      printf("%d",x);
}
```

18. Ecrire le code de la fonction echange et l'appel de la fonction. Quel sera le résultat affiché ?

```
void echange(int *a, int *b) {
    // à compléter
}
main(){
      int x=2,y=3;
      // appel de echange

      printf("%d %d",x,y);
}
```

19. Ecrire le code de la fonction incrementer

```
void incrementer(int *tab, int taille);
```

20. Que fait le code ci-dessous ?

```
int *tab = malloc(10 * sizeof(int));
```

21. Que fait le code ci-dessous ?

```
int tab[3] = {1,2,3};
int *p = tab;

printf("%d\n", *(p+1));
```

22. Ecrire le code de la fonction incrementer

```
int somme(int *tab, int taille);
```

23. Ecrire le code de la fonction copier de dest vers src

```
void copier(int *src, int *dest, int taille);
```