

Exercices de TD

1. Conception de programmes

Pour écrire un programme il faut suivre les étapes suivantes :

1. Déterminer les Entrées du programme
2. Déterminer les sorties du programme
3. Concevoir l'algorithme qui à partir des données produira la sortie
4. Traduire l'algorithme sous forme d'une fonction C

Pour chacuns des problèmes suivants détaillez les étapes de conception décrites ci-dessus.

Exercice 1

Ecrire un programme demandant à l'utilisateur d'entrer son âge, et l'année courante, et qui affiche ensuite à l'écran l'année de sa naissance et le nombre de jours vécus.

► Correction

```
void main( ){
    ecrireString("Entrez votre âge\n");
    int age = lireInt();
    ecrireString("Entrez l'année\n");
    int annee = lireInt();
    int naissance = annee - age;
    ecrireString("Votre annee de naissance est ");
    ecrireInt(naissance);
    ecrireChar('\n');
    int nb_jour= age*365;
    ecrireString("Vous avez vécu ");
    ecrireInt(nb_jours);
    ecrireChar(' jours\n');
}
```

Exercice 2 — Moyenne et Somme Écrivez un programme *Moyenne* qui lit trois notes et affiche leur somme et leur moyenne.

► Correction

```
void main( ) {
    double sum=0;
    int nb_int=3;
    double temp;
    ecrireString("Entrez la première note :");
    temp=lireDouble();
    sum=sum+temp;
    ecrireString("Entrez la deuxième note :");
    temp=lireDouble();
    sum=sum+temp;
    ecrireString("Entrez la troisième note :");
```

```

    temp=lireDouble();
    sum=sum+temp;
    ecrireString("La somme est :");
    ecrireInt(sum);
    ecrireString("La moyenne est :");
    ecrireInt(sum/nb_int);
}

```

Exercice 3 — *Cercle*

Écrivez un programme **Cercle** qui demande le rayon d'un cercle et affiche son périmètre et sa surface. Vous pourrez vous servir de la constante π dont la valeur est une 3.1416.

► Correction

```

void main() {
    double rayon;
    ecrireString("Entrez le rayon du cercle :");
    rayon=lireDouble();
    ecrireString("Le périmètre est :");
    ecrireDouble(2*Math.PI*rayon);
    ecrireString("La surface est :");
    ecrireDouble(3.1416*(rayon*rayon));
}

```

Exercice 4 — *Conversion*

Écrivez un programme **Conversion** qui demande une température en degrés Celsius et la convertit en degrés Fahrenheit. On rappelle la formule

$$f = \frac{9c}{5} + 32$$

où f est la température en degrés Fahrenheit et c en degrés Celsius. (Pensez au type des variables — ce ne sont pas forcément des entiers.)

► Correction

```

void main( ) {
    double temp;
    ecrireString("Entrez la température en degrés Celsius :");
    temp=lireDouble();
    ecrireString("La température en degré Fahrenheit est :");
    ecrireDouble((9*temp)/5 +32);
}

```

Exercice 5 — *TTC*

Écrivez un programme **TTC** qui à partir d'un prix hors taxes et d'un taux de TVA calcule le prix toutes taxes comprises correspondant.

► Correction

```

void main( ) {
    double taux;
    double prix;
    ecrireString("Entrez le taux de tva :");
    taux=lireDouble();
    ecrireString("Entrez le prix :");
    prix=lireDouble();
    ecrireString("Le prix TTC est :");
    ecrireDouble(1.196*prix);
}

```

Exercice 6

Ecrire un programme qui permute deux valeurs entières. Ces valeurs seront lues au clavier.
Même question pour deux caractères.

► Correction

```
void main( ){
    int a = lireInt();
    int b= lireInt();
    int temp =a;
    a = b;
    b = temp;
}

void main( ){
    char a = lireChar();
    char b= lireChar();
    char temp =a;
    a = b;
    b = temp;
}
```

2. Evaluation d'expressions

Exercice 7 — *Valeur des expressions et des variables*

Quelle est la valeur des expressions suivantes ?

1. $53\%7+5*5-3$
2. $12+53\%7*(4*3\%2)$

Si $a=1$, quelles sont les valeurs des expressions booléennes suivantes :

1. $!(a>0) \ \&\& \ (a>0);$
2. $(a!=1) \ == \ !(a==1);$

► Correction

1. $53\%7+5*5-3 : \mathbf{26}$
2. $12+53\%7*(4*3\%2) : \mathbf{12}$
1. $!(a>0) \ \&\& \ (a>0); : \neq 0$
2. $(a!=1) \ == \ !(a==1); : \mathbf{1}$

Exercice 8 — *Evaluation d'expressions booléennes*

Quelles sont les valeurs des expressions booléennes suivantes ?

- | | |
|-----------|--------------------------|
| $10 > 5$ | $5 == 11 - 6$ |
| $10 == 5$ | $!(30 \% 3 == 0)$ |
| $5 == 5$ | $0.3 - 0.2 == 0.2 - 0.1$ |

2. La conditionnelle

Exercice 9 — *Tarif réduit*

Ecrire un programme qui prend l'âge d'un utilisateur en paramètres et affiche le message "tarif réduit" si il a strictement moins de 26 ans, et "tarif normal" si il a plus de 26 ans.

► Correction

```
void tarif(int age){
    if (age <26)
        écrireString("Tarif réduit");
    else
        écrireString("Tarif normal");
}
```

Exercice 10

Concevoir le programme qui calcule la note finale d'une unité d'enseignement étant données la note du partiel et celle de l'examen et sachant que :

- une note inférieure à 7 à l'examen est éliminatoire
- si la note d'examen est supérieure à 7, la note finale est la moyenne des deux notes si elle avantage l'étudiant. Sinon, la note finale est la note d'examen.

► Correction

1. Entrées : la note d'examen *ne* et celle du partiel *np*
2. Sorties : la note finale *nf*
3. Algorithme :

```
Si (ne < 7) alors
    nf = ne
Sinon
    moy = moyenne de ne et np
    si (moy > ne) alors
        nf = moy
    sinon
        nf = ne
```

4. C :

```
double calculNote(double ne, double nf ){
    double nf,moy;
    if (ne < 7) {
        nf = ne;
    }
    else {
        moy = (ne + np)/2;
        if (moy > ne){
            nf = moy;
        }
        else {
            nf = ne;
        }
    }
    return nf;
}
```

Exercice 11

On définit le "type" **BOOL**, grâce aux entiers, pour modéliser les booléens, où **TRUE** est la valeur entière 1 et **FALSE** est la valeur entière 0.

```
#define BOOL int
#define TRUE 1
#define FALSE 0
```

Concevoir le programme qui teste si un entier appartient à l'intervalle donné par deux nombres entiers.

► Correction

1. Entrées : un entier *e*, un intervalle $[a, b]$
2. Sorties : un **BOOL** *res*
3. Algorithme :

```
Si (e < a) ou (e > b) alors
    appartient = false
sinon
    appartient = true
```

4. C :

```
BOOL appartient(int e, int a, int b){
    BOOL res;
    if ((e < a) || (e > b))
        res = FALSE;
    else
        res = TRUE;
    return res;
}
```

Exercice 12

Concevoir le programme qui teste si une année est bissextile. On sait qu'une année divisible par 4 est bissextile sauf si elle est divisible par 100, cependant les années divisibles par 400 sont également bissextiles.

► Correction

1. Entrées : une année *annee*
 2. Sorties : un booléen *bissextile*
 3. Algorithme :
- ```
si (annee divisible par 4) alors
 si (annee divisible par 100) alors
 si (annee divisible par 400) alors
 bissextile = true
 sinon
 bissextile = FALSE
 sinon
 bissextile = true
sinon
 bissextile = FALSE
```

4. C :

```
BOOL bissextile(int annee){
 BOOL bissextile;
 if (annee \% 4 == 0){
 if (annee \% 100){
 if (annee \% 400)
 bissextile = TRUE;
 else
 bissextile = FALSE;
 }
 else
 bissextile = TRUE;
 }
 else
 bissextile = FALSE;
 return bissextile;
}
```

### Exercice 13

Concevoir le programme qui détermine et renvoie le plus grand parmi trois nombres entiers.

#### ► Correction

1. Entrées : 3 entiers a, b et c.
2. Sorties : un entier max
3. Algorithme :

```
Si (a > b) alors
 max = a
sinon
 max = b
Si (c > max) alors
 max = c
```

4. C :

```
int max(int a, int b, int c){
 int max;
 if (a > b)
 max = a;
 else
 max = b;
 if (c > max)
 max = c;
 return max;
}
```

### Exercice 14

Concevoir le programme calcule la valeur absolue d'un entier.

#### ► Correction

1. Entrées : un entier a
2. Sorties : un entier absolue

3. Algorithme :

```
si (a<0)
 absolue = -a
sinon
 absolue = a;
```

4. C :

```
int valeurAbsolue(int a){
 int absolue;
 if (a<0)
 absolue = -a;
 else
 absolue = a;
 return absolue;
}
```

### Exercice 15

Concevoir le programme qui, étant donné la date d'un jour, calcule la date du lendemain de ce jour et l'affiche à l'écran.

#### ► Correction

1. Entrées :

2. Sorties :

3. Algorithme :

```
si ((m est février et que j est 29 ou que j est 28 et que a est bissextile) ou
 (m est un mois de 31 jours et j est 31) ou
 (m est un mois de 30 jours et j est 30)) alors
 nj = 1
 si (m est décembre) alors
 nm = 1
 sinon
 nm = m +1
sinon
 nj=j+1
 nm =m
si (j est 31 et m est décembre) alors
 na = a +1
sinon
 na = a
```

4. C :

```
void lendemain(int j, int m, int a){
 int nj;
 int nm;
 int na;
 if (((m ==2) && ((j == 29) || ((j==28) && !bissextile(a))))
 || (((m==1) || (m==3) || (m == 5) || (m==7) || (m==8) || (m == 10)
 || (m == 12)) && (j == 31))
 || (((m==4) || (m==6) || (m == 9) || (m==11)) && (j == 30)))
 nj = 1;
```

```

 if (m==12)
 nm = 1 ;
 else
 nm=m+1;
 else
 nj=j+1;
 nm = m;
 if ((j == 31) && (m ==12))
 na = a +1;
 else
 na = a;
 ecrireString("La date de demain est :");
 ecrireInt(nj);
 ecrireChar('/');
 ecrireInt(nm);
 ecrireChar('/');
 ecrireInt(na);
 ecrireChar('\n');
}

```

#### 4. Les tableaux et les boucles

##### Exercice 16 — *Vrai ou faux*

1. Tous les éléments d'un tableau ont le même type
2. La taille du tableau peut être fixée au moment de sa déclaration
3. La taille du tableau peut être fixée au moment de sa création
4. Les instructions suivantes sont-elles valides ? Si non corrigez.

```

— int i = {1,2,3};
— double[] d;
— float f[3] ;
— char r[] = {1,2,3};
— int i[] = (3,4,6,8);
— float f[] = {3.2,4.5,5.6};
— char [] c = {'a', 'b','c'};

```

##### ► Correction

1. Tous les éléments d'un tableau ont le même type : **vrai**
2. La taille du tableau peut être fixée au moment de sa déclaration : **faux**
3. La taille du tableau peut être fixée au moment de sa création : **vrai**
4. Les instructions suivantes sont-elles valides ? Si non corrigez.
 

```

— int i = {1,2,3}; non
 int i[] = {1,2,3}; ou int i[3] = {1,2,3};
— double[] d; non
 double d[] ;
— float f[3] ; oui
— char r[] = {1,2,3}; non
 char r[] = {'a', 'b','c'};

```



```

— int i[] = (3,4,6,8); non
 int i[] = {3,4,6,8}; ou int i[4] = {3,4,6,8};
— float f[] == {3.2,4.5,5.6}; non
 float f[] = {3.2,4.5,5.6}; ou float f[3] = {3.2,4.5,5.6};
— char c [] = {'a', 'b', 'c'}; oui

```

### Exercice 17 — *Les tableaux*

1. Déclarer un tableau **t** de taille 3 de réels simple précision.
2. Créer un tableau **tab** d'entiers de taille 8.
3. Déclarer et initialiser un tableau **x** avec les 5 premières lettres de l'alphabet
4. Remplir le tableau **t** précédent avec 3 valeurs quelconques (en une ligne de code)

#### ► Correction

1. Déclarer un tableau **t** de taille 3 de réels simple précision.  

```
float t[3];
```
2. Créer un tableau **tab** d'entiers de taille 8.  

```
int tab [8];
```
3. Déclarer et initialiser un tableau **x** avec les 5 premières lettres de l'alphabet  

```
char x[] = {'a','b','c','d','e'}; ou char x[5] = {'a','b','c','d','e'};
```
4. Remplir le tableau **t** précédent avec 3 valeurs quelconques (en une ligne de code)  

```
t = new float[]{5.3f,7.8f,9.0f};
```

### Exercice 18 — *Un programme éronné*

Corriger les 4 erreurs du programme suivant :

```

void main(){
 float[] t;
 for(i=0;i<=10;i++)
 t(i) = i*100;
}

```

#### ► Correction

```

void main(){
 float t[10];
 int i;
 for(i=0;i<10;i++)
 t[i] = i*100;
}

```

### Exercice 19 — *Tableau multidimensionnel*

Un agenda est vu comme un tableau de 52 semaines, chaque semaine étant composée de 7 jours eux mêmes divisés en 8 plages horaires. Chaque plage peut contenir un message sous la forme d'une chaîne de caractères.

1. Déclarer et construire un tableau représentant un agenda.
2. Noter le message "RDV avec Maman" le lundi de la 25ème semaine à la 4ème plage horaire.

3. Modifier le tableau de telle manière que le nombre de plages horaires passe à 12 par jour la 36ème semaine.
4. Modifier le tableau de telle manière que le nombre de plages horaires passe à 6 le mardi de la 45ème semaine.

#### ► Correction

1. Déclarer et construire un tableau représentant un agenda.  
`String[] [] agenda = new String[53][8][9];`
2. Noter le message "RDV avec Maman" le lundi de la 25ème semaine à la 4ème plage horaire.  
`agenda[25][1][4] = "RDV avec Maman";`
3. Modifier le tableau de telle manière que le nombre de plages horaires passe à 12 par jour la 36ème semaine.  
`agenda[36] = new String[8][13];`
4. Modifier le tableau de telle manière que le nombre de plages horaires passe à 6 le mardi de la 45ème semaine.  
`agenda[45][2] = new String[7];`

#### Exercice 20 — *Fonctions sur les tableaux*

1. Ecrire une fonction qui cherche si un élément appartient à un tableau de char. Le caractère recherché et le tableau seront les deux paramètres de la fonction.
2. Ecrire une fonction qui compte le nombre d'occurrences d'un caractère dans un tableau, c'est à dire le nombre de fois où un élément apparaît dans un tableau de caractères. Le caractère recherché et le tableau seront les deux paramètres de la fonction.
3. Ecrire une fonction qui prend deux tableaux de caractères en paramètres et qui teste si tous les éléments du premier tableau apparaissent au moins une fois dans le deuxième tableau. Il est possible d'utiliser dans le corps de cette fonction la fonction écrite pour la réponse à la question 1.

#### ► Correction

```

BOOL appartient(char elem, char tab[]){
 int i;
 BOOL res = FALSE;
 for (i = 0; i<tab.length; i++){
 if (tab[i] == elem){
 res = TRUE;
 }
 }
 return res;
}

int nbOccurrences(char elem, char[] tab){
 int res = 0;
 int i;
 for (i = 0; i<tab.length; i++){
 if (tab[i] == elem){
 res++;
 }
 }
 return res;
}

```

```

 }
}
return res;
}

BOOL inclus(char[] t1, char[] t2){
 BOOL res = TRUE;
 int i;
 for (i = 0; i<t1.length; i++){
 if (!appartient(t1[i],t2)){
 res = FALSE;
 }
 }
 return res;
}
}

```

### Exercice 21 — *Les chaînes de caractères*

Ecrire une méthode qui prend en paramètres 2 chaînes de caractères `c1` et `c2` et leurs tailles (`n1` et `n2`) et les compare au regard de l'ordre lexicographique (l'ordre du dictionnaire pour les caractères non accentués). Elle doit renvoyer la valeur 0, si les chaînes sont les mêmes, -1 si `c1` est avant `c2`, et 1 sinon.

#### ► Correction

```

int compare(char c1[], int n1, char c2[], int n2){
 int i;
 int taille_min =n1;
 if (n2<n1)
 taille_min =n2;
 for (i=0; i<taille_min;i++)
 {
 if (c1[i] < c2[i])
 return -1;
 else {
 if (c1[i] > c2[i])
 return 1;
 }
 }
 if (n1 == n2)
 return 0;
 if (n1 < n2)
 return -1;
 if (n1 > n2)
 return -1;
}

```

### Exercice 22 — *Sondages - tableaux à 2 dimensions*

Un institut de sondage veut faire une enquête sur les intentions de vote à un référendum. Il y a trois intentions possibles :

- voter oui
- voter non
- voter blanc ou s'abstenir

L'institut veut distinguer les intentions de vote des hommes et des femmes. Le résultat du sondage sera donné sous la forme d'un tableau de la forme :

|       | oui | non | abstention |
|-------|-----|-----|------------|
| homme | 12  | 37  | 25         |
| femme | 9   | 47  | 13         |

La représentation en Java d'un tel tableau se fera de la façon suivante :

|   | 0  | 1  | 2  |
|---|----|----|----|
| 0 | 12 | 37 | 25 |
| 1 | 9  | 47 | 13 |

Ici pour les colonnes, l'indice 0 signifie une intention de vote oui, l'indice 1 signifie non et l'indice 2 signifie l'abstention, et pour les lignes l'indice 0 signifie les hommes et l'indice 1 signifie les femmes.

Ecrivez un programme qui étant donné un tel tableau et qui comporte les fonctions suivantes :

1. une fonction qui calcule le nombre de femmes prises en compte dans l'enquête.
2. une fonction qui calcule le nombre total de personnes ayant une intention de vote donnée. Cette intention de vote, de même que le tableau, sera un paramètre de la fonction. Elle pourra être donnée sous forme de l'indice encodant cette intention.
3. une fonction qui prédit le résultat du vote. Ce résultat ne prend pas en compte les abstentionnistes. Seuls les votes exprimés sont pris en compte (en France).
4. une fonction qui détermine si les hommes et les femmes sont équitablement représentés dans l'enquête.

## ► Correction

```
int nbFemmes(int tab[][3]){
 return tab[1][0] + tab[1][1] + tab[1][2];
}

int nbPersonnes(int intention, int tab[][3]){
 return tab[0][intention] + tab[1][intention];
}

int resultat(int tab[][3]){
 int oui, non;
 oui = tab[0][0] + tab[1][0];
 non = tab[0][1] + tab[1][1];
 if (oui == non){
 return 2;
 }
 else
 if (oui > non){
 return 0;
 }
 else{
 return 1;
 }
}

BOOL parite(int tab[][3]){
 int hommes = 0;
 int femmes = 0;
 int col;
 for (col=0; col<3; col++){
 hommes = hommes + tab[0][col];
 }
}
```

```

 femmes = femmes + tab[1][col];
}
return hommes == femmes;
}

void main(){
 int resEnq[2][3] = {{12,37,25},{9,47,13}};
 int res;

 ecrireString("Nombre de femmes: ");
 ecrireInt(nbFemmes(resEnq));
 ecrireString("Nombre de personnes votant oui: ");
 ecrireInt(nbPersonnes(0,resEnq));
 ecrireString("Nombre de personnes s'abstenant: ");
 ecrireInt(nbPersonnes(2,resEnq));
 res = resultat(resEnq);
 if (res == 0){
 ecrireString("Victoire du oui");
 }
 else {
 if (res == 1){
 ecrireString("Victoire du non");
 }
 else{
 ecrireString("Egalite du oui et du non");
 }
 }
 if (parite(resEnq)){
 ecrireString("La parite est respectee");
 }
 else{
 ecrireString("La parite n'est pas respectee");
 }
}

```