TP 1 : Programmation C++ correction

Exercice 1 :

```
#include<iostream>
using namespace std;
double b(double x, double y)
{
	double m;
	m = (x + y) / 2;
	return m;
}
int main()
{
	double a;
	a = b(3.2, 4.2);
	cout << "Le résultat vaut :" << a << endl;
	return 0;
}
```

Exercice 2 :

```
#include <iostream>
using namespace std;
const int n = 4;
void saisir(int t[n])
{
	int i;
	for (i = 0; i < n; i++)
	{
		cout << "Tapez la valeur numero " << i << " : ";
		cin >> t[i];
	}
}
void affiche(int t[n])
{
	int i;
	for (i = 0; i < n; i++) cout << "La valeur numero " << i << " est : " <<
t[i] << endl;
}
int main()
{
	int a[n];
	saisir(a);
	affiche(a);
	return 0;
}
```

Exercice 3 :

```
#include<iostream>
using namespace std;
```

```
//void echange(double a, double b)
void echange(double& a, double& b)
{
    double temporaire(a); //On sauvegarde la valeur de 'a'
    a = b;                //On remplace la valeur de 'a' par celle de 'b'
    b = temporaire;       //Et on utilise la valeur sauvegardée pour mettre
l'ancienne valeur de 'a' dans 'b'
}

int main()
{
    double a(1.2), b(4.5);

    cout << "a vaut " << a << " et b vaut " << b << endl;

    echange(a, b);   //On utilise la fonction

    cout << "a vaut " << a << " et b vaut " << b << endl;
    return 0;
}
```

Exercice 4 :

```
#include <iostream>
using namespace std;

int main()
{
    int const nombreNotes(6);
    double notes[nombreNotes];

    notes[0] = 12.5;
    notes[1] = 19.5;  //Bieeeen !
    notes[2] = 6.;    //Pas bien !
    notes[3] = 12;
    notes[4] = 14.5;
    notes[5] = 15;

    double moyenne(0);
    for (int i(0); i < nombreNotes; ++i)
    {
        moyenne += notes[i];   //On additionne toutes les notes
    }
    //En arrivant ici, la variable moyenne contient la somme des notes (79.5)
    //Il ne reste donc qu'à diviser par le nombre de notes
    moyenne /= nombreNotes;

    cout << "Votre moyenne est : " << moyenne << endl;

    return 0;
}
//Un tableau statique est toujours passé par référence. Et il n'y a pas besoin
d'utiliser l'esperluette &
```

Exercice 5 :

```
#include <iostream>
#include <vector> //Ne pas oublier !
using namespace std;
```

```
int main()
{
    vector<double> notes; //Un tableau vide
    //   vector<int> tableau(5); //si on veut indique la taille du tableaux
    //vector<int> tableau(5, 3);  //Crée un tableau de 5 entiers valant tous 3

    notes.push_back(12.5);  //On ajoute des cases avec les notes
    notes.push_back(19.5);
    notes.push_back(6);
    notes.push_back(12);
    notes.push_back(14.5);
    notes.push_back(15);
    //tableau.pop_back(); //suprrime une case
    double moyenne(0);
    for (int i(0); i < notes.size(); ++i)
        //On utilise notes.size() pour la limite de notre boucle car il indique
la taille du tableau
    {
        moyenne += notes[i];   //On additionne toutes les notes
    }

    moyenne /= notes.size();
    //On utilise à nouveau notes.size() pour obtenir le nombre de notes

    cout << "Votre moyenne est : " << moyenne << endl;

    return 0;
}
```

Exercice 6 :

```
#include<iostream>
using namespace std;
struct point
{
      double x, y;
};
int main()
{
      point a, b, c;
      a.x = 3.2;
      a.y = 6.4;
      cout << "Tapez l'abscisse de b : ";
      cin >> b.x;
      cout << "Tapez l'ordonnée de b : ";
      cin >> b.y;
      c.x = (b.x + a.x) / 2;
      c.y = (b.y + a.y) / 2;
      cout << "Abscisse de c : " << c.x << endl;
      cout << "Ordonnée de c : " << c.y << endl;
      return 0;
}
```

Exercice 7 :

```
#include<iostream>
using namespace std;
#include<cmath>
struct point
{
      double x, y;
};
void saisir_point(point& p)
```

```
{
	cout << "Tapez l'abscisse du point : "; cin >> p.x;
	cout << "Tapez l'ordonnée du point : "; cin >> p.y;
}
void afficher_point(point p)
{
	cout << "Abscisse du point : " << p.x << endl;
	cout << "Ordonnée du point : " << p.y << endl;
}
double distance(point a, point b)
{
	double dx, dy;
	dx = a.x - b.x;
	dy = a.y - b.y;
	return sqrt(dx * dx + dy * dy);
}
void milieu(point a, point b, point& m)
{
	m.x = (a.x + b.x) / 2;
	m.y = (a.y + b.y) / 2;
}
int main()
{
	point X, Y, Z;
	double d;
	cout << "SAISIE DE X" << endl;
	saisir_point(X);
	cout << "SAISIE DE Y" << endl;
	saisir_point(Y);
	d = distance(X, Y);
	cout << "La distance de X à Y est : " << d << endl;
	milieu(X, Y, Z);
	cout << "AFFICHAGE DU POINT Z" << endl;
	afficher_point(Z);
	return 0;
}
```

Exercice 8 :

```
#include <iostream>
using namespace std;
//------------------structure employe-----------------------------------------
struct employe
{
    char nom[10];
    char prenom[10];
    double salaire;
};
void saisir_employe(employe& e)
{
    cout << "Tapez le nom : "; cin >> e.nom;
    cout << "Tapez le prenom : "; cin >> e.prenom;
    cout << "Tapez le salaire : "; cin >> e.salaire;
}

void affiche_employe(employe e)
{
    cout << e.nom << " " << e.prenom << " " << e.salaire << endl;
}
//-------- end employe --------------------
//----------- liste----------------------------

const int liste_pleine = -1;
```

```
const int liste_nb_max = 100;

struct liste
{
    int nb;
    employe t[liste_nb_max];
};
void init_liste(liste& l)
{
    l.nb = 0;
}

int ajoute(liste& l, employe e)
{
    int r;
    if (l.nb == liste_nb_max) r = liste_pleine;
    else { r = 0; l.t[l.nb] = e; l.nb++; }
    return r;
}

void affiche(liste l)
{
    int i;
    if (l.nb == 0) cout << "LISTE VIDE" << endl;
    for (i = 0; i < l.nb; i++) affiche_employe(l.t[i]);
}

void recherche(liste l1, char nom[], liste& l2)
{
    int i;
    init_liste(l2);
    for (i = 0; i < l1.nb; i++) if (strcmp(l1.t[i].nom, nom) == 0) ajoute(l2,
l1.t[i]);
}
//------------end liste--------------------------

//-----------menu---------------------------------
int choix()
{
    int i;
    cout << "1.Ajoute un employe" << endl;
    cout << "2.Afficher la liste" << endl;
    cout << "3.Rechercher un employe" << endl;
    cout << "4.Quitter" << endl;
    cout << "Votre choix :"; cin >> i;
    return i;
}
bool traiter_choix(liste& l, int choix)
{
    employe e;
    char nom[10];
    liste l2;
    int r;
    bool fini = false;

    switch (choix)
    {
    case 1:
        saisir_employe(e);
        r = ajoute(l, e);
        if (r == liste_pleine) cout << "La liste est pleine" << endl;
        break;
```

```
    case 2:
        affiche(l);
        break;

    case 3:
        cout << "Tapez le nom :"; cin >> nom; recherche(l, nom, l2);
        cout << "Voici le resultat de la recherche :" << endl;
        affiche(l2);
        break;

    case 4:
        fini = true;
        break;
    }
    return fini;
}

void menu(liste& l)
{
    bool fini;
    int i;
    do {
        i = choix();
        fini = traiter_choix(l, i);
    } while (fini == false);
}


int main()
{
    liste l;
    //----------stucture employe-------------------
    employe x;
    saisir_employe(x);
    affiche_employe(x);
    //-------- end employe --------------------

    //-----------liste-----------------------

    liste l2;
    char nom[] = "salma";
    init_liste(l);
    ajoute(l, x);
    cout << "votre liste est: ";
    affiche(l);
    recherche(l, nom, l2);
    cout << "votre liste 2 est: ";
    affiche(l2);
    //--------------end liste-------------------

    //-----------menu---------------------

    menu(l);
}
```