Atelier 4: Classes, Objets et Constructeurs

Exercice 1 :

Effectuer les opérations arithmétiques sur des **nombres complexes** à l'aide d'une classe et d'un objet.
Le programme doit demander la partie réelle et imaginaire de deux nombres complexes et afficher les parties réelle et imaginaire de l'opération demandée. (égalité, addition, soustraction, multiplication, division). *Le choix de l'opération peut être fait par un Menu.*

Réponse :

```
#include <iostream>
#include <math.h>
#include <iomanip>
using namespace std;

class complexe{
 private:
    int reel1,imaginaire1,reel2,imaginaire2;
 public:
    complexe(){}
    complexe(int a1, int b1, int a2, int b2):reel1(a1),imaginaire1(b1),reel2(a2),imaginaire2(b2){}
    void complexe_egalite(int a1, int b1, int a2, int b2){
           if(a1==a2 && b1==b2)
            cout<<"Les deux nombres sont egaux"<<endl;
           else
            cout<<"Les deux nombres ne sont pas egaux"<<endl;
           }
    void complexe_addition(int a1, int b1, int a2, int b2){
        int ad1=a1+a2;
        int ad2=b1+b2;
        cout<<"L'addition de ces deux nombre complexes donne : "<<ad1<<"+("<<ad2<<")i"<<endl;
    }
    void complexe_soustraction(int a1, int b1, int a2, int b2){
        int sou1=a1-a2;
        int sou2=b1-b2;
        cout<<"La soustraction de ces deux nombre complexes donne : "<<sou1<<"+("<<sou2<<")i"<<endl;
    }
    void complexe_multiplication(int a1, int b1, int a2, int b2){
        int mu1=(a1*a2)-(b1*b2);
        int mu2=(a1*b2)+(a2*b1);
        cout<<"La multiplication de ces deux nombre complexes donne : "<<mu1<<"+("<<mu2<<")i"<<endl;
    }
     void complexe_division(int a1, int b1, int a2, int b2){
        if(a2 ==0 && b2 ==0)
            cout<<"Impossible de divise sur un nombre complexe null"<<endl;
        else if(a2 !=0 && b2==0){
```

```
            float div1=a1/a2;
            float div2=b1/a2;
            cout<<"La division de ces deux nombre complexes donne : "<<div1<<"+("<<div2<<")i"<<endl;

        }
        else if(a2 ==0 && b2!=0){
            float div1=a1/b2;
            float div2=b1/b2;
            cout<<"La division de ces deux nombre complexes donne : "<<div2<<"-("<<div1<<")i"<<endl;
        }else{
            int d1=(a1*a2)+(b1*b2);
            int d2=(b1*a2)-(a1*b2);
            int d3=pow(a2,2)+pow(b2,2);
            double div1=(d1/d3);
            double div2=(d2/d3);
            cout<<"La division de ces deux nombre complexes donne : ("<<d1<<"+("<<d2<<"i))/"<<d3<<" =
"<<setprecision(2) <<div1<<"+"<<setprecision(2) <<div2<<"i"<<endl;
        }

    }
};

int main()
{
    int r1,im1,r2,im2,y;
    cout<<"--------- Bienvenue dans les operateurs des nombres complexes ---------\n"<<endl;
    cout<<"Entrer la partie reel du premier nombre complexe : ";
    cin>>r1;
    cout<<"Entrer la partie imaginaire du premier nombre complexe : ";
    cin>>im1;
    cout<<"\nEntrer la partie reel du deuxieme nombre complexe : ";
    cin>>r2;
    cout<<"Entrer la partie imaginaire du deuxieme nombre complexe : ";
    cin>>im2;
    complexe c(r1,im1,r2,im2);
    do{
       int a;
       cout<<"\nVos nombres complexes : "<<r1<<"+("<<im1<<")i  ,  "<<r2<<"+("<<im2<<")i"<<endl;
       cout<<"\n--------- Menu des operateurs ---------\n"<<endl;
       cout<<"Pour l'egalite, entrer le numero : 1 "<<endl;
       cout<<"Pour l'addition, entrer le numero : 2 "<<endl;
       cout<<"Pour la soustraction, entrer le numero : 3 "<<endl;
       cout<<"Pour la multiplication, entrer le numero : 4 "<<endl;
       cout<<"Pour la division, entrer le numero : 5 "<<endl;
       cin>>a;

       switch(a){
          case 1: c.complexe_egalite(r1,im1,r2,im2);
                  break;
          case 2: c.complexe_addition(r1,im1,r2,im2);
                  break;
          case 3: c.complexe_soustraction(r1,im1,r2,im2);
                  break;
          case 4: c.complexe_multiplication(r1,im1,r2,im2);
```

```
                break;
        case 5: c.complexe_division(r1,im1,r2,im2);
                break;
        default : cout<<"Erreur nombre entre invalide"<<endl;
      }
      cout<<"Voulez vous continuer ? Oui(1)/Non(0) : ";
      cin>>y;
      }while(y != 0);
   return 0;
}
```

Exercice 2 :

Ecrire un programme en C++ avec une classe mère **Animal**. À l'intérieur, définir des variables nom et d'âge, et la fonction set_value(). Créer ensuite deux sous classes de base **Zebra** et **Dolphin** qui écrivent un message indiquant l'âge, le nom et donnant des informations supplémentaires (par exemple, le lieu d'origine), Créer 2 variables un de type Zebra et l'autre Dolphin puis appeler la méthode set_value() pour chaque instance.

Réponse :

```
#include <iostream>

using namespace std;

class Animal{
 protected:
 string nom;
 int age;
 public :
 void set_value(string a, int b){
     nom=a;
     age=b;
 }
};

class Zebra : public Animal{
  int poid;
  public :
      void set_value(string a, int b, int c){
          nom=a;
          age=b;
          poid=c;
      }
      void Affichage(){
       cout<<"Le nom : "<<nom<<" ,l'age : "<<age<<" , le poid(en kg) : "<<poid<<endl;
      }
};

class Dolphin : public Animal{
 string couleur;
 public :
      void set_value(string a, int b, string c){
```

```
          nom=a;
          age=b;
          couleur=c;
          }
      void Affichage(){
       cout<<"Le nom : "<<nom<<" ,l'age : "<<age<<" , la couleur du peau : "<<couleur<<endl;
      }
};

int main()
{
    Zebra z;
    Dolphin d;
    z.set_value("Roky",15,100);
    d.set_value("Doli",40,"Bleu");
    z.Affichage();
    d.Affichage();


    return 0;
}
```

Exercice 3 :

Créer une classe **Personne** qui comporte trois champs privés, nom, prénom et date de naissance. Cette classe comporte un constructeur pour permettre d'initialiser des données. Elle comporte également une méthode polymorphe *Afficher* pour afficher les données de chaque personne.

- Créer une classe **Employe** qui dérive de la classe Personne, avec en plus un champ Salaire accompagné de sa propriété, un constructeur et la redéfinition de la méthode Afficher.
- Créer une classe **Chef** qui dérive de la classe Employé, avec en plus un champ Service accompagné de sa propriété, un constructeur et la redéfinition de la méthode Afficher.
- Créer une classe **Directeur** qui dérive de la classe Chef, avec en plus un champ Société accompagné de sa propriété, un constructeur et la redéfinition de la méthode Afficher.

Réponse :

```
#include <iostream>

using namespace std;

class Person{
protected:
    string nom;
    string prenom;
    string date;
public:
    Person():nom("Vide"),prenom("Vide"),date("Vide"){}
    Person(string a, string b, string c):nom(a),prenom(b),date(c){}
    void Afficher(){}
};

class Employe : public Person{
    protected:
    int salaire;
    public:
```

```
    Employe():Person("Vide","Vide","Vide"),salaire(0){}
    Employe(string a, string b, string c, int d):Person(a,b,c),salaire(d){}
    void Afficher(){
    cout<<"Le nom de l'employe : "<<nom<<", le prenom de l'employe : "<<prenom<<", la date de
naissance de ce employe : "<<date<<", \nson salaire(DH) : "<<salaire<<endl;
    }
};
class Chef : public Employe{
    protected:
    string service_accompagne;
    public:
    Chef():Employe("Vide","Vide","Vide",0),service_accompagne("Vide"){}
    Chef(string a, string b, string c, int d , string e):Employe(a,b,c,d),service_accompagne(e){}
    void Afficher(){
    cout<<"\nLe nom de chef : "<<nom<<", le prenom de chef : "<<prenom<<", la date de naissance de
chef : "<<date<<", son salaire(DH) : \n"<<salaire<<", le service qui accompagne :
"<<service_accompagne<<endl;
    }
};
class Directeur : public Chef{
    protected:
   string societe_accompagne;
    public:
    Directeur():Chef("Vide","Vide","Vide",0,"Vide"),societe_accompagne("Vide"){}
    Directeur(string a, string b, string c, int d , string e, string
f):Chef(a,b,c,d,e),societe_accompagne(f){}
    void Afficher(){
    cout<<"\nLe nom du directeur : "<<nom<<", le prenom du directeur : "<<prenom<<", la date de
naissance du directeur : "<<date<<",\n son salaire(DH) : "<<salaire<<", le service qui accompagne :
"<<service_accompagne<<", societe qui accompagne : "<<societe_accompagne<<endl;
    }
};

int main()
{
    Employe e("El bakkali","Said","2 janvier 1985",5000);
    Chef c("Ben taib","Omar","7 juillet 1972",9000,"maintenance");
    Directeur d("El aarabi","Ismail","26 avril 1965",14000,"Gestion de l'entreprise","IBM");
    e.Afficher();
    c.Afficher();
    d.Afficher();

    return 0;
}
```