

**(C++)**

**COMPTE RENDU DE Atelier 3**



**LST GI/S5/Groupe 2**

**GI-51 POO en C++**

* **Préparé par :**
* **L’KHAMAL FADOUA**

* **Encadre par :**
* **Pr.Lotfi ELAACHAK**
* **Pr. Ikram Ben Abdel Ouahab**
* **lien Github :**

//ATELIER 3

//EXERCICE 1

#include <iostream>

using namespace std;

class Myclass{

public://accees specifique

int num;//attribut

Myclass();//constructeur par defaut

~Myclass();//destructeur par utilisateur

//fonction definie a l'interieure de note classe

void print(){

cout<<"Faire entrer le numero : "<<endl;

cin>> num ;

cout<<"votre numero est : "<<num<<endl;

};

};

Myclass::Myclass(){

};

Myclass::~Myclass(){

};

int main(){

Myclass etu;

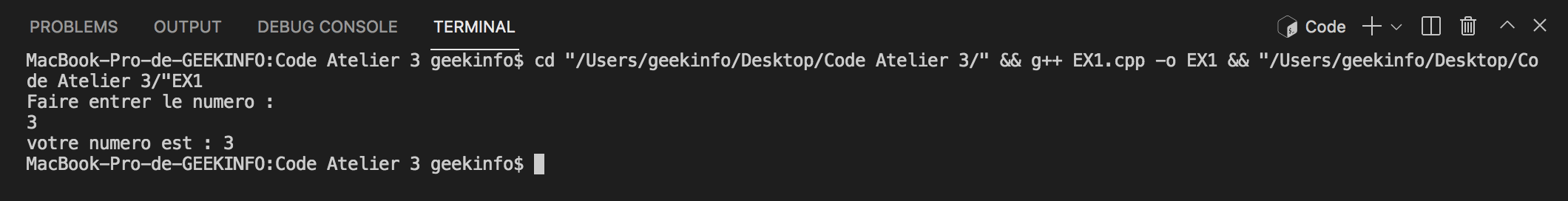
etu.print();

return 0;

};

* PROGRAMME 1 :
* SOLUTION DES PROGRAMMES ATELIER 3 :

|  |  |
| --- | --- |
| **NOM DE LA CLASSE** | * **Myclass** |
| **NOM DES ATTRIBUTS** | * **+nom : string** |
| **NOM DES METHODES** | * **+ Myclass ()** * **+~Myclass ()** * **+ print ()** |



* **Résultat final :**

* PROGRAMME 2 :

//ATELIER 3

//EXERCICE 2

#include <iostream>

using namespace std;

class Shape

{

public:

float x, y;

int R1,R2,T1,T2,n;

Shape()

{

cout<<"Calculez l'aire du triangle ou du rectangle ? "<<endl;//l'utilisateur doit choisir le reactangle ou le triangle

cout<<"Tapez 1 pour le rectangle ."<<endl;

cout<<"Tapez 2 pour le triangle ."<<endl;

cin>>n;

}

//fonction pour calculer l'aire:

void calculer(){

//l'utilisateur doit entrer les dimensions de rectangle

if (n==1){

cout<<"Donnez les dimensions du rectangle :"<<endl;

cout<<"Entrez l'hauteur :";

cin>>R1;

cout<<endl<<"Entrez le largeur :";

cin>>R2;

cout << "L'aire du rectangle est: " <<R1 \* R2 << endl;

}

Shape a;

  Shape a;

  a.calculer();

  return 0;

}

//l'utilisateur doit entrer les dimensions de rectangle

else if(n==2){

cout<<"Donnez les dimensions du triangle :"<<endl;

cout<<"Entrez l'hauteur :";

cin>>T1;

cout<<endl<<"Entrez le largeur :";

cin>>T2;

cout << "L'aire du triangle est: " <<(T1 \* T2)/2 << endl;

}

}

};

//Fonction principale

int main (){

Shape a;

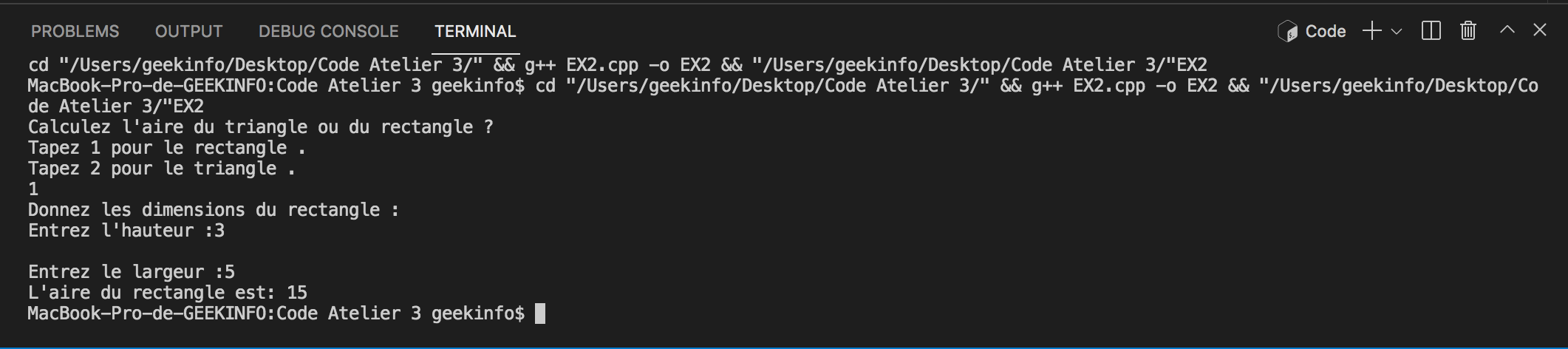
a.calculer();

return 0;

}

|  |  |
| --- | --- |
| **NOM DE LA CLASSE** | * **Shape** |
| **NOM DES ATTRIBUTS** | * **+ R1 : float** * **+ R2 : float** * **+ T1 : float** * **+ T2 : float** * **+ n : int** |
| **NOM DES METHODES** | * **+ Shape()** * **+ calculer()** |

* **Résultat final :**



//ATELIER 3

//EXERCICE 3

#include <iostream>

using namespace std;

class complexe{

public:

int RE1;

int RE2;

int IM1;

int IM2;

int op;

//FAIRE ENTRER LES COMPLEXES

complexe(){

cout<<"Pour le premier complexe "<<endl;

cout<<"Faire entrer la partie reele"<<endl;

cin>>RE1;

cout<<"Faire entrer la partie imaginaire"<<endl;

cin>>IM1;

cout<<"Pour le deuxieme complexe "<<endl;

cout<<"Faire entrer la partie reele "<<endl;

cin>>RE2;

cout<<"Faire entrer la partie imaginaire"<<endl;

cin>>IM2;

cout<<"Choisissez une operation"<<endl;

cout<<"1.Somme"<<endl;

cout<<"2.Soustraction"<<endl;

cout<<"3.Multiplication"<<endl;

cout<<"4.Division"<<endl;

cin>>op;

};

//FONCTION DE CALCULE

void calculer(){

if(op==1){

cout<<"la somme de deux complexes est : " <<RE1+RE2<< " +"<<IM1+IM2<<"i"<<endl;//LA SOMME

} else if (op==2){

cout<<"la soustraction de deux complexes est : " <<RE1-RE2<< " +"<<IM1-IM2<<"i"<<endl;//LA SOUSTRACTION

}

else if(op==3){

cout<<"la multiplication de deux complexes est : "<<RE1\*RE2+IM1\*IM2<< " +"<<RE1\*IM2+IM1\*RE2<<"i"<<endl;//LA MULTIPLICATION

}

else if(op==4){

if(IM2==0 && RE2==0){

cout<<"erreur division sur 0";

}

* PROGRAMME 3 :

else{

cout<<"la division de deux complexes est : " <<float(IM1\*IM2-RE1\*RE2)/float(IM2\*IM2+RE2\*RE2)<< " +"<<float(IM1\*RE2-RE1\*IM2)/float(RE2\*RE2+IM2\*IM2)<<"i"<<endl;//LA DIVISION

}

}

else{

cout<<"Choisissez une des quatres operations!"<<endl;

}

}

};

//FONCTION PRINCIPALES

int main(){

complexe a;

a.calculer();

return 0;

};

|  |  |
| --- | --- |
| **NOM DE LA CLASSE** | * **complexe** |
| **NOM DES ATTRIBUTS** | * **+ RE1 : int** * **+ RE2 : int** * **+ IM1 : int** * **+ IM2 : int** * **+ op : int** |
| **NOM DES METHODES** | * **+ complexe ()** * **+ calcul ()** |

* **Résultat final :**



//ATELIER 3

//EXERCICE4

#include <iostream>

using namespace std;

//la class mere

class mere{

public:

string mere;

void display(){

cout<<" Test1 "<<endl;

};

};

//la class fille

class fille:public mere{

public:

string fille;

void display(){

cout<<" Test2 "<<endl;

}

};

//fonction principale

int main(){

mere M;

M.display();

fille F;

F.display();

};

* PROGRAMME 4 :

|  |  |
| --- | --- |
| **NOM DE LA CLASSE** | * **mere** |
| **NOM DES ATTRIBUTS** |  |
| **NOM DES METHODES** | * **+ display ()** |

* **Résultat final :**



* PROGRAMME 5 :

//ATELIER 5

//EXERCICE 5

#include<iostream>

using namespace std;

class Animal{

private:

int Age;

string Nom;

public:

Animal(int Age , string Nom){

this->Age=Age;

this->Nom=Nom;

}

void set\_value();

};

class Zebra{

protected:

int Age;

private:

string Nom;

public:

Zebra(int Age , string Nom){

this->Age=Age;

this->Nom=Nom;

}

void show1(){

cout<<"Age = "<<Age<<" , Nom = "<<Nom<<endl;

}

};

class Dolphin{

protected:

int Age;

private:

string Nom;

public:

Dolphin(int Age , string Nom){

this->Age=Age;

this->Nom=Nom;

} void show2(){

cout<<"Age = "<<Age<<" , Nom = "<<Nom<<endl;

}

};

//FONCTION PRINCIPALES

int main(){

Zebra z(16,"ZEBRA");

Dolphin d(20,"DOLPHIN");

z.show1();

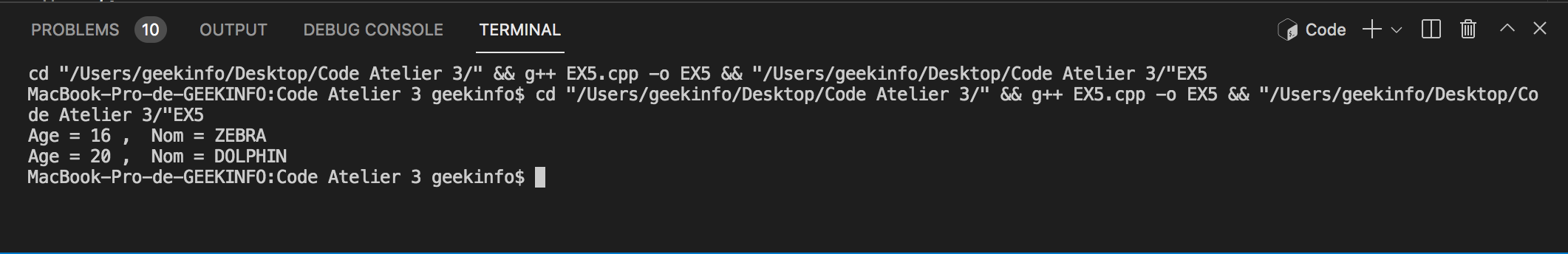
d.show2();

return 0;

};

|  |  |
| --- | --- |
| **NOM DE LA CLASSE** | * **Animal** * **Zebra** * **Dolphin** |
| **NOM DES ATTRIBUTS** | * **-Nom : string** * **#Age : int** |
| **NOM DES METHODES** | * **+ Animal ()** * **+ Zebra ()** * **+ Dolphin ()** * **+ Show1 ()** * **+ Show2 ()** |

* **Résultat final :**



* PROGRAMME 6 :

#include <iostream>

#include <string>

using namespace std;

class Personne{ // classe m�re

private:

string nom ;

string prenom;

int datedenaissance;

public:

Personne(string n,string p,int d  ):nom{n},prenom{p},datedenaissance{d}{

}

 virtual void afficher()

 {

     cout<<"Nom: "<<nom<<endl<<"Prenom: "<<prenom<<endl<<"date de naissance: "<<datedenaissance<<endl;

 }

};

class Employe : Personne // classe fille : employe

{

    private:

    double salaire;

    public:

    Employe(string n,string p,int d,double s):Personne(n,p,d){

        salaire = s;

    }

  void afficher(){

        Personne::afficher();

         cout<<"salaire : "<< salaire <<endl;

    }

};

class Chef : Employe{ // classe fille : chef

    private :

    string service ;

    public:

    Chef(string n,string p ,int d ,double s ,string ser):Employe(n,p,d,s){

       service=ser;

    }

  void  afficher(){ // fonction d'affichage

Employe::afficher();

cout<<"service  : "<< service <<endl;

    }

};

class Directeur : Chef{ // classe fille : directeur de la classe m�re : chef

private:

     string societe;

     public:

     Directeur(string n,string p ,int d ,double s ,string ser,string soc):Chef(n,p,d,s,ser){

         societe=soc;

     }

   void  afficher(){

         Chef::afficher();

         cout<<"societe : "<< societe <<endl;

     }

};

int main(){

    Directeur dir("hajar","balirh",2002,100000,"oui","oui");

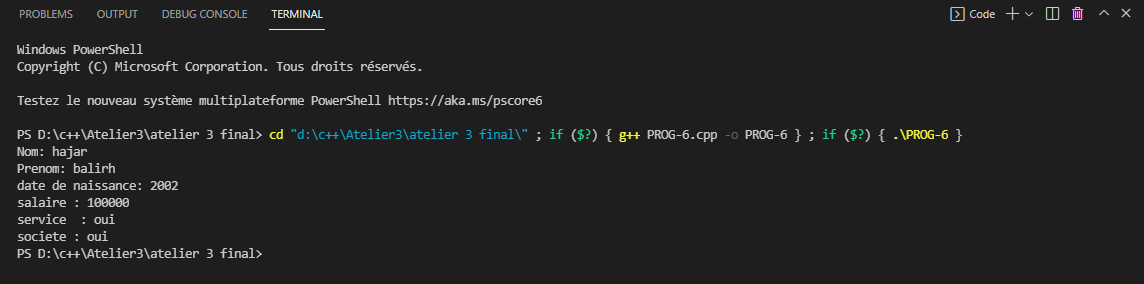
    dir.afficher();

    return 0;

}

|  |  |
| --- | --- |
| **NOM DE LA CLASSE** | * **Personne** * **Employe** * **Chef** * **Directeur** |
| **NOM DES ATTRIBUTS** | * **-Nom : string** * **-Prenom : string** * **-Datedenaissance : int** * **-Salaire : double** * **-Service : string** * **-Societe : string** |
| **NOM DES METHODES** | * **+ Personne ()** * **+ afficher ()** * **+ Employe ()** * **+ Chef ()** * **+ Directeur ()** |

* **Résultat final :**



* PROGRAMME 7 :

//ATELIER 3

//EXERCICE 7

#include<iostream>

#include<math.h>

using namespace std;

class Vecteur3d{

private:

float x,y,z;

public:

Vecteur3d(float x=0,float y=0,float z=0)

{

this->x=x;

this->y=y;

this->z=z;

}

void Afficher()

{

cout<<"("<<x<<","<<y<<","<<z<<")"<<endl;

}

Vecteur3d Somme(Vecteur3d A)

{

Vecteur3d B;

B.x=x+A.x;

B.y=y+A.y;

B.z=z+A.z;

return A;

}

float Produit(Vecteur3d C)

{

float p;

p=x\*C.x+y\*C.y+z\*C.z;

return p;

}

bool Coincide(Vecteur3d Z)

{

if(x==Z.x && y==Z.y && z==Z.z)

return true;

return false;

}

float Norme(Vecteur3d N)

{

float n;

n= sqrt(x\*x+y\*y+z\*z);

return n;

}

};

//FONCTION PRINCIPALES

int main()

{

float d;

Vecteur3d M(9,7,-4) , N(11,-5,7) , O;

M.Afficher();

O=N.Somme(M);

O.Afficher();

d=M.Produit(N);

cout<<"le produit est"<<d<<endl;

if(M.Coincide(N)==true){

cout<<"M et N se coincide"<<endl;

}

else{

cout<<"M et N ne se coinside pas"<<endl;

}

}

|  |  |
| --- | --- |
| **NOM DE LA CLASSE** | * **Vecteur3d** |
| **NOM DES ATTRIBUTS** | * **- x = float** * **- y = float** * **- z = float** * **- p = float** * **- n = float** * **- d = float** * **- nom = string** |
| **NOM DES METHODES** | * **+ Vecteur3d ()** * **+ afficher ()** * **+ produit ()** * **+ coincide()** * **+ somme ()** |

* **Résultat final :**



//ATELIER 3

//EXERCICE 9

#include<iostream>

using namespace std;

class Test {

private:

static int cmp;

public:

void call(){

cmp++;

cout<<cmp<<endl;

}

};

int Test :: cmp=0;

//FONCTION PRINCIPALES

int main(){

Test A1,A2;

A1.call();

A2.call();

}

* PROGRAMME 9 :

|  |  |
| --- | --- |
| **NOM DE LA CLASSE** | * **Test** |
| **NOM DES ATTRIBUTS** | * **- n = int** |
| **NOM DES METHODES** | * **+ call ()** |

* **Résultat final :**

