Manar Echchachoui

Filière : IDAI

***DEVOIR 2***

EXERCICE 1

#include <iostream>

using namespace std;

void compter() {

int no\_appels = 0;

no\_appels++;

cout << "Appel numéro " << no\_appels << endl;

}

int main() {

compter();

compter();

compter();

return 0;

}

EXERCICE 2

#include <iostream>

using namespace std;

int nbr\_pair(int n)

{

return n % 2 == 0;

}

int multiple(int n) {

return n % 3 == 0;

}

int main() {

int n;

cout << "Entrez un nombre entier : ";

cin >> n;

if (nbr\_pair(n)) {

cout << "Le nombre " << n << " est pair." << endl;

}

if (multiple(n)) {

cout << "Le nombre " << n << " est multiple de 3." << endl;

} else

if (nbr\_pair(n) && multiple(n)) {

cout << "Le nombre " << n << " est multiple de 6." << endl;

}

return 0;

}

EXERCICE 3

1/

#include <iostream>

Using namespace std;

int main() {

int tableau[10];

int plus\_grand = tableau[0];

int plus\_petit = tableau[0];

for (int i = 0; i < 10; i++) {

cout << "Entrez un nombre entier : ";

cin >> tableau[i];

if (tableau[i] > plus\_grand) {

plus\_grand = tableau[i];

}

if (tableau[i] < plus\_petit) {

plus\_petit = tableau[i];

}

}

cout << "Le plus grand nombre est " << plus\_grand << endl;

cout << "Le plus petit nombre est " << plus\_petit << endl;

return 0;

}

2/

#include <iostream>

using namespace std;

int main() {

int \*tableau =new int [10];

int\*plus\_grand = tableau;

int\*plus\_petit = tableau;

for (int i = 0; i < 10; i++) {

cout << "Entrez un nombre entier : ";

cin >> tableau[i];

if (tableau[i] > \*plus\_grand) {

\*plus\_grand = tableau[i];

}

if (tableau[i] < \*plus\_petit) {

\*plus\_petit = tableau[i];

}

}

cout << "Le plus grand nombre est " << \*plus\_grand << endl;

cout << "Le plus petit nombre est " << \*plus\_petit << endl;

return 0;

}

EXERCICE 5

#include <iostream>

using namespace std;

int main() {

int a = 10;

int &ref\_a = a;

int \*p\_a = &a;

cout << "a = " << a << ", adresse de a = " << &a << ", valeur pointée par ref\_a = " << ref\_a << endl;

cout<< "ref\_a = " << ref\_a << ", adresse de ref\_a = " << &ref\_a << ", valeur pointée par p\_a = " << \*p\_a << endl;

return 0;

EXERCICE 6

1/

#include <stdio.h>

void incrementer(int \*val) {

(\*val)++;

}

void permuter(int \*a, int \*b) {

int t = \*a;

\*a = \*b;

\*b = t;

}

int main() {

int x;

int y ;

printf("donner la valeur de x");

scanf("%d",&x);

printf("donner la valeur de Y");

scanf("%d",&y);

printf(": x = %d\n", x);

incrementer(&x);

printf("l'incrémentation donne : x = %d\n", x);

printf(": x = %d, y = %d\n", x, y);

permuter(&x, &y);

printf("la permutation donne : x = %d, y = %d\n", x, y);

return 0;

}

2/

#include <stdio.h>

using namespace std;

void incrementer(int &val) {

(val)++;

}

void permuter(int &a, int &b) {

int t = a;

a = b;

b = t;

}

int main() {

int x;

int y ;

cout<<"donner la valeur de x"endl;

cin>>x endl;

printf<<"donner la valeur de Y"endl;

cin>>y endl;

cin<<"x="<<x<<endl;;

incrementer(&x);

cin<<"l'incrémentation donne : x ="<<x<<endl;

cin<<"x= "<<x<<,"y="<<y<<endl;

permuter(&x, &y);

cin<<"la permutation donne : x = "<<x<<," y =" <<y<<endl;

return 0;

}

EXERCICE 8

#include <iostream>

#include <math.h>

using namespace std;

class Complexe {

public:

float re;

float img;

public:

Complexe(float a, float b) {

re = a;

img = b;

}

void afficher() {

cout << re << " + " << img << "i " << endl;

}

Complexe somme(Complexe x){

Complexe s(0,0);

s.re=re+x.re;

s.img=img+x.img;

return s;

}

float module()

{

return sqrt(re\*re+ img \*img);

}

Complexe somme(float x){

Complexe s1(0,0);

s1.re=re+x;

s1.img=img;

return s1;

}

};

int main() {

float M;

float a=2.5;

Complexe s(0,0);

Complexe x1(5, 10), x2(2, 4);

x1.afficher();

x2.afficher();

s=x1.somme(x2);

s.afficher();

M=x1.module();

cout << "La Norme du nombre complexe est :" << M << endl;

s=x2.somme(a);

s.afficher();

return 0;}

EXERCICE 11

#include <iostream>

Using namespace std;

class vecteur3d {

public:

vecteur3d() {

x = 0;

y = 0;

z = 0;

}

void afficher() {

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

}

vecteur3d somme(const vecteur3d & v) {

return vecteur3d(x + v.x, y + v.y, z + v.z);

}

float produit\_scalaire(const vecteur3d & v) {

return x \* v.x + y \* v.y + z \* v.z;

}

bool coincide(const vecteur3d& v) {

return x == v.x && y == v.y && z == v.z;

}

float norme() {

return sqrt(x \* x + y \* y + z \* z);

}

Int vecteur3d normax(const vecteur3d & v1, const vecteur3d & v2) {

if (v1.norme() > v2.norme()) {

return v1;

} else {

return v2;

}

}

float x, y, z;

};

#include <iostream>

Using namespace std;

int main() {

vecteur3d v1(1, 2, 3);

vecteur3d v2(4, 5, 6);

v1.afficher();

v2.afficher();

vecteur3d v3 = v1 + v2;

v3.afficher();

float produit = v1.produit\_scalaire(v2);

cout << "Produit scalaire : " << produit << :endl;

bool coincide = v1.coincide(v2);

cout << "Les vecteurs sont-ils égaux ? " << coincide <<:endl;

float norme = v1.norme();

cout << "Norme du vecteur : " << norme << endl;

vecteur3d vmax = vecteur3d::normax(v1, v2);

vmax.afficher();

return 0;

}

EXERCICE 13

include<iostream>

Using namespace std;

#include "point.h"

class point {

public:

point(float x, float y);

void deplace(float dx, float dy);

void affiche();

private:

float x\_;

float y\_;

};

point::point(float x, float y) {

x\_ = x;

y\_ = y;

}

void point::deplace(float dx, float dy) {

x\_ += dx;

y\_ += dy;

}

void point::affiche() {

cout << "(" << x\_ << ", " << y\_ << ")" << endl;

}

int main() {

point p(1.0, 2.0);

p.affiche();

p.deplace(3.0, 4.0);

p.affiche();

return 0;

}

EXERCICE 14