

3. Indicar la salida por pantalla (2 puntos-20 minutos)

```
#include <iostream.h>
class EUITelemento;
class EUITipila
{
int num;
    EUITelemento *pila[10];
public:
    EUITipila():num(0){};
    bool push(EUITelemento *in){
        if(num>=10)return false;
        pila[num++]=in;
        return true;
    }
    EUITelemento *pop(){
        cout<<"pop!"<<endl;
        if(num==0)return 0;
        return pila[--num];
    }
    int getNum(){return num;}
};
class EUITelemento
{
protected:
    float valor;
public:
    virtual bool ejecuta(EUITipila *)=0;
    float getValor(EUITipila *p=0){
        ejecuta(p);
        return valor;
    }
};
class EUITivalor:public EUITelemento
{
public:
    bool ejecuta(EUITipila *p){return true;}
    EUITivalor(float v){valor=v;}
};

class EUITIsuma:public EUITelemento
{
public:
    bool ejecuta(EUITipila *p){
        EUITelemento *aux1=p->pop();
        if(aux1==0)return false;
        float v1=aux1->getValor(p);
        EUITelemento *aux2=p->pop();
        if(aux2==0)return false;
        float v2=aux2->getValor(p);
        valor=v1+v2;
        cout<<"suma de "
        <<v1<<"+"<<v2<<"="<<(v1+v2)<<endl;
        delete aux1;
        delete aux2;
        return true;
    }
};

void main()
{
    EUITipila calculadora;
    calculadora.push(new EUITivalor(5));
    calculadora.push(new EUITivalor(4));
    calculadora.push(new EUITivalor(3));
    calculadora.push(new EUITIsuma);
    calculadora.push(new EUITIsuma);
    calculadora.push(new EUITivalor(2));
    calculadora.push(new EUITIsuma);
    calculadora.pop()->getValor(&calculadora);
}
```

línea	Impresión por pantalla
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

**CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70**

**ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70**



4. Ejercicio de programación (2,5 puntos-50 minutos)

Dadas las siguientes clases en C++:

```
class Instrumento
{
protected:
int peso_;
char* nombre_;
public:
Instrumento(): peso_(0)
{
nombre_ = new char[8];
strcpy(nombre_, "Ninguno");
}
~Instrumento()
{
delete nombre_;
}
int peso() const
{
return peso_;
}

void peso(const int& peso)
{
peso_ = peso;
}
char* nombre() const
{
return nombre_;
}
void nombre(const char* nombre)
{
delete nombre_;
if (0 == nombre)
{
nombre_ = 0;
}
else
{
int len = strlen(nombre);
nombre_ = new char[len + 1];
strcpy(nombre_, nombre);
}
}
void mostrar ()
{
cout << "Mostrando Clase
Instrumento con nombre:" <<
nombre_ << endl;
}
virtual void tocar_nota()=0;
};

class Guitarra : public
Instrumento
{
private:
int cuerdas_;
public:
Guitarra() : Instrumento(),
cuerdas_(6) {};
~Guitarra() {};
void mostrar()
{
cout << "Mostrando Clase
Guitarra con nombre:" << nombre_
<< endl;
};
void tocar_nota()
{
cout << "Tocando nota con
Guitarra:" << endl;
};
};
```

Hacer un programa principal que realice lo siguiente:

- Crear un array de 6 punteros a Instrumento.
- Inicializar los punteros del array con objetos de la clase Guitarra.
- Aplicar a los seis elementos la función tocar_nota() ¿Qué sale por pantalla?
- Aplicar a los seis elementos la función mostrar() ¿Qué sale por pantalla?
- ¿Podríamos crear un objeto de la clase Instrumento directamente? Razonar la respuesta.
- ¿Sería necesario crear un constructor copia para la clase Instrumento? Razonar la

**CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70**

**ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70**

Cartagena99

5. Problema de Análisis y Diseño Orientado a Objetos (2.5 puntos - 50 minutos)

Para el código adjuntado se pide:

- Ingeniería inversa: Diagrama de clases.
- Ingeniería inversa: Diagrama de secuencia.
- Resultado de su ejecución en la consola.
- Indicar los patrones GRASP empleados en este patrón.
- Diseñar e implementar el servicio *rotar()*, tal que

$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$

Empléese sobre el punto *p2*.

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

class Punto {
public:
    double x, y;
    Punto(double xi, double yi) : x(xi), y(yi) {}
    Punto(const Punto& p) : x(p.x), y(p.y) {}
    Punto& operator=(const Punto& rhs) {
        x = rhs.x;
        y = rhs.y;
        return *this;
    }
    friend ostream&
    operator<<(ostream& os, const Punto& p) {
        return os << "x=" << p.x << " y=" << p.y;
    }
};

class Vector {
public:
    double magnitud, direccion;
    Vector(double m, double d) : magnitud(m), direccion(d) {}
};

class Espacio {
public:
    static Punto trasladar(Punto p, Vector v) {
        p.x += (v.magnitud * cos(v.direccion));
        p.y += (v.magnitud * sin(v.direccion));
        return p;
    }
};

int main() {
    Punto p1(1, 2);
    Punto p2 = Espacio::trasladar(p1, Vector(3, 3.1416/3));
    cout << "p1: " << p1 << " p2: " << p2 << endl;

    return 0;
}
```

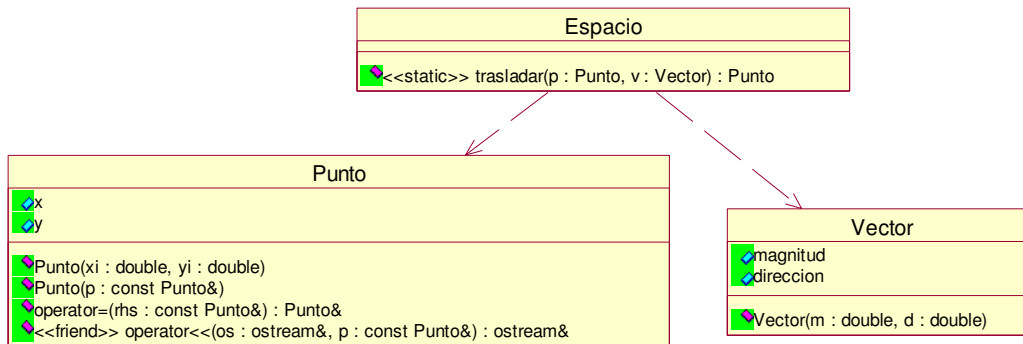
The logo for 'Cartagena99' features the text 'Cartagena99' in a stylized, outlined font. The '99' is significantly larger and more prominent than the rest of the text. The logo is set against a background of a blue and orange gradient with a white starburst effect behind the text.

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

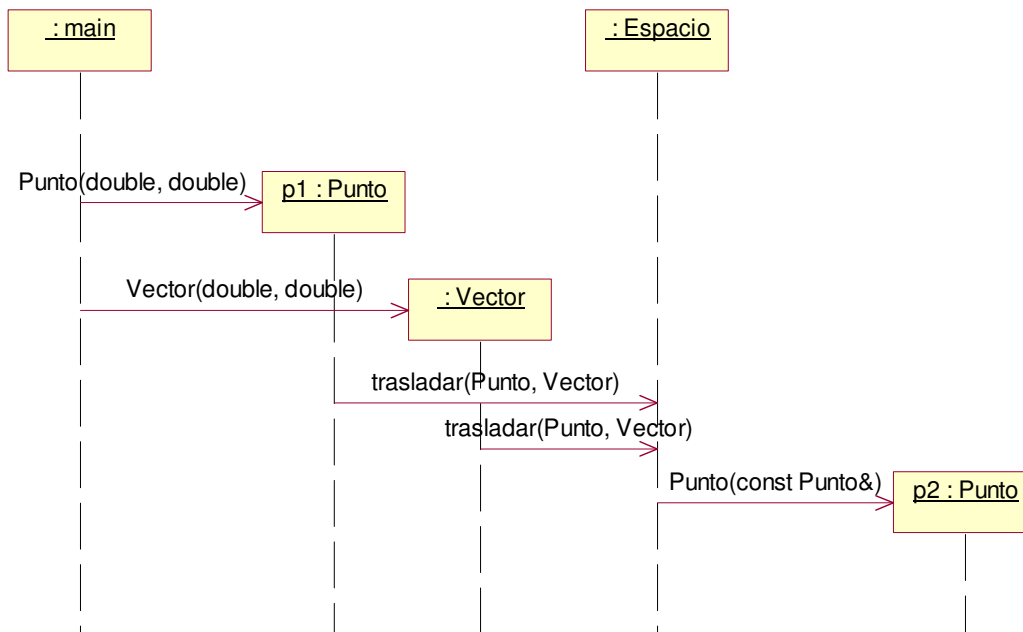
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Resolución

a)



b)



c) p1: x=1 y=2 p2: x=2.5 y=4.6

d) Se ha aplicado Experto de Información en la clase Punto y Vector. Para evitar el acoplamiento entre ambas clases se ha aplicado el patrón Indirección y por tanto una Fabricación Pura con la clase Espacio.

e)

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Cartagena99

```

class Espacio {
public:
    static Punto trasladar(Punto p, Vector v) {
        p.x += (v.magnitud * cos(v.direccion));
        p.y += (v.magnitud * sin(v.direccion));
        return p;
    }
    static Punto rotar(Punto p, double theta) {
        Punto res(0,0);
        res.x = (p.x * cos(theta)) - (p.y *sin(theta));
        res.y = (p.x * sin(theta)) + (p.y *cos(theta));
        return res;
    }
};

int main() {
    Punto p1(1, 2);
    Punto p2 = Espacio::trasladar(p1, Vector(3, 3.1416/3));
    Punto p3 = Espacio::rotar(p2,3.1416/6);

    cout << "p1: " << p1 << " p2: " << p2 << " p3: " << p3 <<endl;

    return 0;
}

```



Cartagena99

**CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70**

**ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70**