



Documento anónimo

## Parcial2-PolimorfismoA2018.pdf

EXAMENES



3º Informática Industrial I



Grado en Ingeniería Electrónica Industrial y Automática



Escuela Politécnica Superior  
Universidad Carlos III de Madrid

¿Estudiar con una cachimba? 🤔🔧

Si!!!!!! 😄

🤔 Aprovecha este descuentazo en toda la web, solo para los estudiantes madrileños. CÓDIGO: UNIBENGALA





# montero espinosa

## Academia Universitaria

En nuestra academia TE OFRECEMOS:

### LOS MEJORES PROFESORES



Son los protagonistas: Calidad, preparación y motivación es lo que define a nuestro equipo. Ingenier@s y licenciad@s se ciñen estrictamente a los contenidos de la Uni, y lo mejor, van al grano.

### CERCA DE TI



Nos ubicamos cerca de las Universidades, para que no pierdas tiempo.

### QUE EL RITMO NO PARE



Todas nuestras clases se imparten tanto presenciales como online.

### A NUESTROS APUNTES, LOS QUIERE TODO EL MUNDO



Los más completos, claros y ordenados apuntes de teoría. También tenemos la mayor colección de ejercicios de examen resueltos.

academia universitaria



*Montero Espinosa, patrocinador oficial de tus aprobados.*

**¡Infórmate!**  **689 71 67 71**

Tenemos un programa que “llena una caja de herramientas” con herramientas de distinto tipo. Teniendo en cuenta el main.cpp que está a continuación y la salida por terminal, rellena los huecos con el código necesario en cada clase.

```
main.cpp
#include <iostream>
#include <cajaherramientas.h>
#include <martillo.h>
#include <destornillador.h>

using namespace std;

int main(int argc, char *argv[])
{
    CajaHerramientas C1(3);
    Martillo M1("Martillo de maza", "Madera");
    Destornillador D1("Destornillador de estrella", "Plástico y metal", 3), D2(" Destornillador
plano", "Madera y metal", 5);
    C1.add_herramienta(&M1);
    C1.add_herramienta(&D1);
    C1.add_herramienta(&D2);
    C1.muestra_contenido();
    return 0;
}
```

```
Herramienta 0
Martillo de maza
El martillo está hecho de madera y metal
El martillo se usa para clavar
=====
Herramienta 1
Destornillador de estrella
El destornillador esta hecho de metal y plástico
El destornillador se usa para atornillar y desatornillar
=====
Herramienta 2
Destornillador plano
El destornillador esta hecho de metal y plástico
El destornillador se usa para atornillar y desatornillar
=====
Press <RETURN> to close this window...
```

```

Herramienta.h
#ifndef HERRAMIENTA_H
#define HERRAMIENTA_H
#include <iostream>
#include <string>
using namespace std;

class Herramienta
{
public:
    Herramienta();
[1]                // [1] Constructor parametrizado
[2]                // [2] Método void uso(void)
[3]                // [3] Método string material(void) que devuelve un string indicando
el tipo de material de la herramienta
[4]                // [4] Método get_name que devuelve el atributo _name
[5]:               // [5] Accesibilidad

    string _name;
    string _material;

};

#endif // HERRAMIENTA_H

```

[6] Implementación de los métodos de Herramienta

```

Destornillador.h
#ifndef DESTORNILLADOR_H
#define DESTORNILLADOR_H

#include "herramienta.h"
class Destornillador :[7]                // [7] Hereda de herramienta
{
public:
    Destornillador();
[8]                // [8] Constructor parametrizado
[9]                // [9] Métodos que consideres imprescindibles

```

```
[10]                                     // [10] Método get_tam que devuelve el atributo _tam
private:
    int _tam;
};

#endif // DESTORNILLADOR_H
```

[11] Implementación de los métodos de Destornillador

```
Martillo.h

#ifndef MARTILLO_H
#define MARTILLO_H

#include "herramienta.h"

class Martillo : [12]                // [12] Hereda de herramienta
{
public:
    Martillo();                      // [13] Constructor parametrizado
    [14]                             // [14] Métodos que consideres imprescindibles
};

#endif // MARTILLO_H
```

[15] Implementación de los métodos de Martillo

CajaHerramientas.h

```
#ifndef CAJAHERRAMIENTAS_H
#define CAJAHERRAMIENTAS_H

#include "herramienta.h"

class CajaHerramientas
{
public:
    CajaHerramientas();
    CajaHerramientas(int n);
[16]           //[16]Método que permite añadir una herramienta a _contenido
[17]           //[17] Método para mostrar por pantalla el contenido de la caja de
herramientas
    ~CajaHerramientas();
private:
[18]           //[18]Vector de 5 posiciones para almacenar las
herramientas de la caja
    int _capacidad;
    int _num_herramientas;
};

#endif // CAJAHERRAMIENTAS_H
```

[18]Implementación de los métodos de CajaHerramientas



## Solución.

### Herramienta.h

```
#ifndef HERRAMIENTA_H
#define HERRAMIENTA_H
#include <iostream>
#include <string>
using namespace std;

class Herramienta
{
public:
    Herramienta();
    Herramienta(string nombre, string material);          // [1] Constructor parametrizado
    virtual void uso(void)=0;                             // [2] Método void uso(void)
    virtual string material(void)=0;                       // [3] Método string material(void) que devuelve un
    string indicando el tipo de material de la herramienta
    string get_name(void);                                // [4] Método get_name que devuelve el atributo _name
protected:                                              // [5] Accesibilidad

    string _name;
    string _material;

};

#endif // HERRAMIENTA_H
```

### Herramienta.cpp

```
-----
#include "herramienta.h"

Herramienta::Herramienta()
{

}

Herramienta::Herramienta(string nombre, string material){
    _name=nombre;
    _material=material;
}

string Herramienta::get_name(){
    return(_name);
}
```

```

-----
Destornillador.h

#ifndef DESTORNILLADOR_H
#define DESTORNILLADOR_H

#include "herramienta.h"
class Destornillador :public Herramienta    // [7] Hereda de herramienta
{
public:
    Destornillador();
    Destornillador(string nombre, string material, int tam);          // [8] Constructor
    parametrizado
    void uso(void);           // [9] Métodos que consideres imprescindibles
    string material(void);
    int get_tam(void); // [10] Método get_tam que devuelve el atributo _tam
    private:
        int _tam;
};

#endif // DESTORNILLADOR_H

```

```

-----
Destornillador.cpp

#include "destornillador.h"

Destornillador::Destornillador() :Herramienta()
{

}

Destornillador::Destornillador(string nombre, string material, int tam):Herramienta(nombre,
material){
    _tam=tam;
}

void Destornillador::uso(){
    cout<<"El destornillador se usa para atornillar y desatornillar"<<endl;
}

string Destornillador::material(){
    string texto="El destornillador esta hecho de ";
    texto=texto+_material;
    return(texto);
}

```



```
int Destornillador::get_tam(){
    return(_tam);
}
```

---

Martillo.h

```
#ifndef MARTILLO_H
#define MARTILLO_H
```

```
#include "herramienta.h"
```

```
class Martillo :public Herramienta           //[12]Hereda de herramienta
{
public:
    Martillo();
    Martillo(string nombre, string material);    //[13] Constructor parametrizado
    void uso(void);           //[14] Métodos que consideres imprescindibles
    string material(void);
};
```

```
#endif // MARTILLO_H
```

---

Martillo.cpp

```
#include "martillo.h"
```

```
Martillo::Martillo()
{

}

Martillo::Martillo(string nombre, string material):Herramienta(nombre, material){

}

void Martillo::uso(){
    cout<<"El martillo se usa para clavar"<<endl;
}

string Martillo::material(){
    string texto="El martillo está hecho de ";
    texto=texto+_material;
    return(texto);
}
```



**TU** **ADEMÁS TE FINANCIAMOS**  
**CARNET**  
**POR 20€**  
+ 4 clases de circulación

autoescuelalara.com  
647 63 53 39



DE TODOS LOS PERMISOS  
**GRATUITAS**

CLASES  
TEÓRICAS  
ONLINE



CajaHerramientas.h

```
#ifndef CAJAHERRAMIENTAS_H
#define CAJAHERRAMIENTAS_H
```

```
#include "herramienta.h"
```

```
class CajaHerramientas
```

```
{
```

```
public:
```

```
    CajaHerramientas();
```

```
    CajaHerramientas(int n);
```

```
void add_herramienta(Herramienta *H);    //[16] Método que permite añadir una  
herramienta a _contenido
```

```
void muestra_contenido(void);            //[17] Método para mostrar por pantalla el  
contenido de la caja de herramientas
```

```
private:
```

```
    Herramienta *_contenido[5];
```

```
    //[18] Vector de 5 posiciones para
```

```
    almacenar las herramientas de la caja
```

```
    int _capacidad;
```

```
    int _num_herramientas;
```

```
};
```

```
#endif // CAJAHERRAMIENTAS_H
```

```
-----  
CajaHerramientas.cpp
```

```
#include "cajaherramientas.h"
```

```
CajaHerramientas::CajaHerramientas()
```

```
{
```

```
    _num_herramientas=0;
```

```
}
```

```
CajaHerramientas::CajaHerramientas(int n){
```

```
    _capacidad=n;
```

```
    _num_herramientas=0;
```

```
}
```

```
void CajaHerramientas::add_herramienta(Herramienta *H){
```

```
    if(_num_herramientas==5){
```

```
        cout<<"No se pueden añadir más herramientas"<<endl;
```

```
        return;
```

```
    }
```

```
    _contenido[_num_herramientas]=H;
```

```
    _num_herramientas++;
```

WUOLAH

```
}  
  
void CajaHerramientas::muestra_contenido(){  
    for(int i=0;i<_num_herramientas;i++){  
        cout<<"Herramienta "<<i<<endl;  
        _contenido[i]->get_name();  
        _contenido[i]->uso();  
        cout<<_contenido[i]->material()<<endl;  
    }  
}
```