SEMANA # 5 CONCEPTOS BASICOS DE POO

Para iniciar con este tema es necesario entender varios conceptos que tendremos que dominar para la aplicación de los mismos en nuestro programa.

Estas son las ideas más básicas que todo aquel que trabaja en programación Orientada a Objetos debe comprender y manejar constantemente:

-Clase

-Objeto

-Abstracción

-Encapsulamiento

-Herencia

-Polimorfismo

Clase:

Una clase, es simplemente una **abstracción** que hacemos de nuestra experiencia sensible. El ser humano tiende a agrupar seres o cosas –objetos- con características similares en grupos –clases-.

Objeto

Un objeto es un conjunto de **atributos y métodos**, un objeto se deriva de una clase.

Atributos (Características):
- Raza
- Color de Pelo
- Años

Métodos (Acciones):
- Comer
- Dormir
- Jugar

Atributos son las características de ese objeto y los métodos son las acciones.

Abstracción:

Proceso mental de extracción de las características esenciales de algo, ignorando los detalles superfluos.

Es decir toma lo principal que necesitas y olvídate de lo que no es importante.

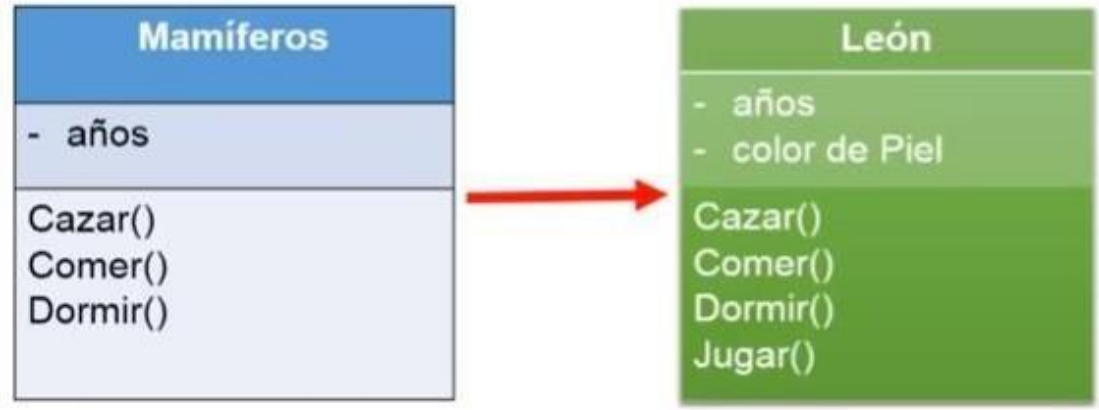
- DNI
- Nombre
- Edad
- Talla
- Peso

Encapsulación:

Proceso por el que se ocultan los detalles del soporte de las características de una abstracción. Es decir encapsulamos características importantes y no nos metemos con ellos en absoluto.

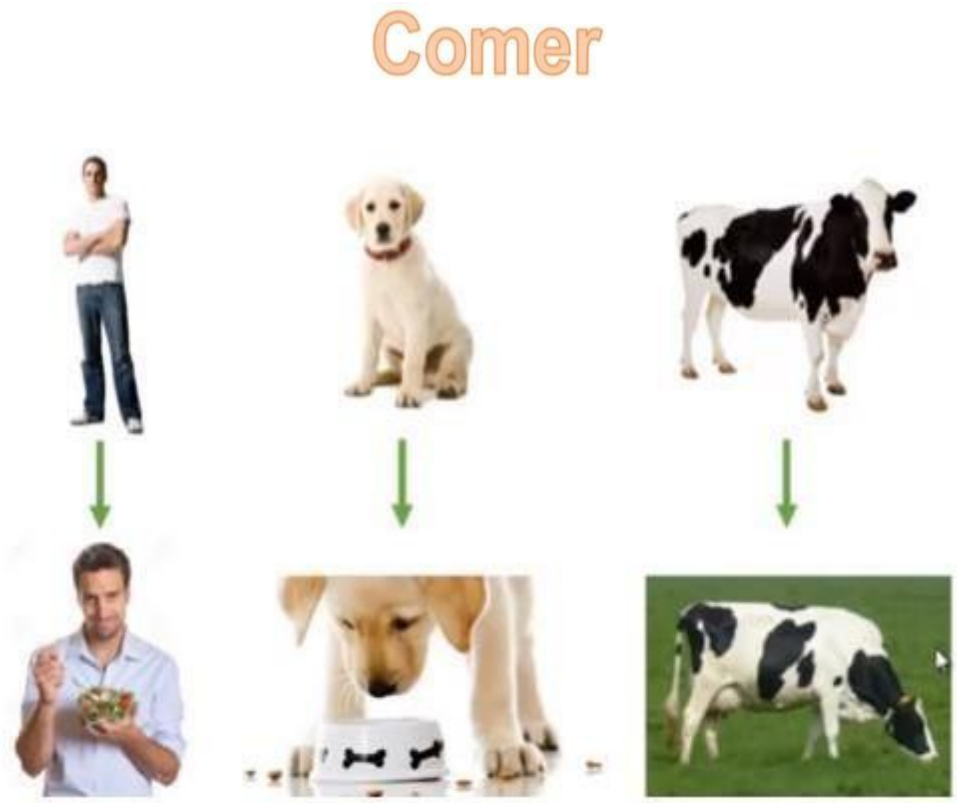
Herencia:

La herencia es donde una clase nueva se crea a partir de una clase existente, heredando todos sus atributos y métodos.



Polimorfismo:

Es aquella cualidad que poseen los objetos para responder de distinto modo ante el mismo mensaje.



Ahora vemos como definimos eso en C++

Clases en C | (Ámbito global) | main()

```cpp
//Clases en C
#include<iostream>
#include<stdlib.h>
using namespace std;

class Persona {
private://ATRIBUTOS
    int edad;
    string nombre;
public://METODOS
    Persona(int, string);//CONSTRUCTOR
    void leer();
    void correr();
};
//Declaracion de constructor, inicializamos los atributos
Persona::Persona(int _edad, string _nombre) {
    edad = _edad;
    nombre = _nombre;
}
//Declaramos los metodos(leer)
void Persona::leer() {
    cout << "Soy " << nombre << " y estoy leyendo un libro" << endl;
}
//Declaramos el metodo(correr)
void Persona::correr() {
    cout << "Soy " << nombre << " y estoy corriendo un a maraton" << endl;
}
int main() {
    //Creamos los objetos
    Persona p1 = Persona(20, "Alberto");
    Persona p2(19, "Maria");
    Persona p3(21, "Juan");
    //Les asignamos los metodos
    p1.leer();
    p2.correr();
    p3.correr();
    p3.leer();

    system("pause");
    return 0;

}
```

Sobrecarga de constructores

Antes de iniciar repasemos los aprendido la semana anterior.

```
1  /*Ejercicio 1. Construya una clase llamada Rectángulo que
2  tenga los siguientes atributos: largo y ancho; y los siguientes
3  métodos: perimetro() y área()*/
4  #include<iostream>
5  #include<stdlib.h>
6  using namespace std;
7
8  class Rectangulo{
9      private://Atributos
10         float largo,ancho;
11     public:
12         Rectangulo(float,float);//constructor
13         void perimetro();
14         void area();
15 };
16   Rectangulo::Rectangulo(float _largo,float _ancho){
17     largo= _largo;
18     ancho= _ancho;
19
20    }
21  void Rectangulo::perimetro(){
22     float _perimetro;
23     _perimetro= (2*largo)+(2*ancho);
24     cout<<"El perimetro es:  "<<_perimetro<<endl;
25  }
26  void Rectangulo::area(){
27     float _area;
28     _area= largo*ancho;
29     cout<<"El area es:  "<<_area<<endl;
30  }
31 int main(){
32     Rectangulo r1(11,7);
33
34     r1.perimetro();
35     r1.area();
36
37
38     system("pause");
39       return 0;
40 }
41
```

Prof: Alberto Espinoza Zamora Cel: 8932-1139

Los constructores son funciones, también pueden definirse varios constructores para cada clase, es decir, el constructor puede **sobrecargarse**. La única limitación(como en todos los casos de sobrecarga) es que **no pueden declararse varios constructores con el mismo número y tipo de argumentos.**

Veamos el siguiente ejemplo para aclarar este concepto:

```
1  //Sobrecarga de constructores
2
3  #include<iostream>
4  #include<stdlib.h>
5  using namespace std;
6
7  class Fecha{
8      private://Atributos
9      int dia,mes, anio;
10     public://Metodos
11     Fecha(int,int,int);//constructor I
12     Fecha(long);//Constructor 2
13     void mostrarFecha();
14 };
15 //Constructor I
16 Fecha::Fecha(int _dia,int _mes,int _anio){
17     anio= _anio;
18     mes= _mes;
19     dia= _dia;
20 }
21 //constructor 2
22 Fecha::Fecha(long fecha){
23     anio=int(fecha/10000);//extraer el anio
24     mes=int((fecha-anio*10000)/100);//Extraer el mes
25     dia=int(fecha-anio*10000-mes*100);//Extraer dia
26
27 }
28 void Fecha::mostrarFecha(){
29     cout<<"La fecha es:  "<<dia<<"/"<<mes<<"/"<<anio<<endl;
30
31 }
32 int main(){
33     Fecha hoy(20,03,19);
34     Fecha ayer(20190319);
35
36     hoy.mostrarFecha();
37     ayer.mostrarFecha();
38     system("pause");
39     return 0;
40 }
41
```

Prof: Alberto Espinoza Zamora Cel: 8932-1139

SEMANA # 10 Herencia en POO

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

class Persona{
    private://Atributos
    string nombre;
    int edad;
    public://metodos
        Persona(string,int);//Constructor
        void mostrarPersona();

};
class Alumno:public Persona{
    private://atributo
    string codigoAlumno;
    float notaFinal;
    public://metodos
    Alumno(string,int,string,float);
    void mostrarAlumno();


};

//Constructor de persona(clase padre)
Persona::Persona(string _nombre,int _edad){
    nombre=_nombre;
    edad=_edad;
}
Alumno::Alumno(string _nombre,int _edad,string _codigoAlumno,float _notaFinal):Persona(_nombre,_edad){
    codigoAlumno=_codigoAlumno;
    notaFinal=_notaFinal;
}

void Persona::mostrarPersona(){
    cout<<"Nombre:  "<<nombre<<endl;
    cout<<" Edad:   "<<edad<<endl;
}
```

```
38  void Alumno::mostrarAlumno(){
39      mostrarPersona();
40      cout<<"Codigo de Alumno: "<<codigoAlumno<<endl;
41      cout<<"Nota Final: "<<notaFinal<<endl;
42  }
43
44  int main(){
45      Alumno alumno1("Alberto",20,"3350-12",89.7);
46
47      alumno1.mostrarAlumno();
48
49      system("pause");
50      return 0;
51  }
```

UNIVERSIDAD DE CIENCIAS EMPRESARIALES – UCEM

INTRODUCCIÓN A LA PROGRAMACIÓN (IN-1100; 3 créditos)

Prof: Alberto Espinoza Zamora Cel: 8932-1139