CONSTRUCTORES Y DESTRUCTORES

Constructores: Es un método especial que se ejecuta automáticamente cuando se crea (instancia) un objeto de una automáticamente cuando se crea (instancia) un objeto de una clase.

Ejem: def __init__ (self,…)

Ejemplo:

```
Class Persona:
    Def __init__(self,nombre,edad):
        Self.nombre = nombre
        Self.edad = edad
```

Destructor: Es un método especial que se ejecuta automáticamente cuando el objeto esta apunto de ser destruido. (por ejemplo) cuando no se usa más o se elimina.

```
Def __del__(self):
    Print(f" {self.nombre} fue elimidano")
```

Ciclo de vida de un objeto: el ciclo de vida de un objeto es el conjunto de etapas por las que pasa un objeto desde que se crea hasta que se destruye.

Las etapas del siglo de vida son:

1. Creación: el objeto se construye he inicializa (__init__).
2. Uso: se llama sus métodos y se accede a asus atributos.
3. Destrucción: el objeto se elimina de la memoria (__del__)

Ejemplo N°1:

Clase:trianguloRectangulo

Atributo:catetoa, catetob

Acción:calcular_hipotenusa()

Objeto:

```
import math
class TrianguloRectangulo:
   def __init__(self,cateto_a,cateto_b): #constructor
      self.cateto_a = cateto_a
      self.cateto_b = cateto_b

   def calcular_hipotenusa(self):
      hipotenusa = math.sqrt(self.cateto_a**2+ self.cateto_b**2)
      return hipotenusa

   def __del__(self):
      print("Objeto TrianguloRectangulo destruido")
```

```
def main():
    try:
        cateto1 = float(input("ingrese el valor del primer cateto:"))
        cateto2 = float(input("ingrese el valor del segundo cateto:"))

        triangulo = TrianguloRectangulo(cateto1,cateto2)
        resultado = triangulo.calcular_hipotenusa()

        print(f"La hipotenusa del triangulo es {resultado}")
    except NameError:
        print("El obajeto Triangulo ya no existe (fue destruido)")
if __name__== "__main__":
    main()
```

Ejemplo N°2:

Clase: comida

Atributo: proteínas, carbohidratos, grasas

Acción: calcular_calorias()

Objeto: almuerzo = comida

```
import math
class Comida:
    def __init__(self,proteinas,carbohidratos,grasas):
        self.proteinas = proteinas
        self.carbohidratos = carbohidratos
        self.grasas = grasas
        print("Objeto comida creado")
        print(f"{self.proteinas}g. {self.carbohidratos}g. {self.grasas}g")

    def calcula_calorias(self):
        calorias = (self.proteinas*4 + self.carbohidratos*4 + self.grasas*9)
        return calorias

    def mostrar_informacion(self):
        print("INFORMACION NUTRICIONAL")
        print(f"proteinas: {self.proteinas}g.")
        print(f"carbohidratos: {self.carbohidratos}g.")
        print(f"grasas: {self.grasas}g.")
        print(f"Calorias totales :{self.calcula_calorias()}kcal")

almuerzo = Comida(proteinas=30,carbohidratos=50,grasas=20)

almuerzo.mostrar_informacion()

del Comida
try:
    comida.calcular_calorias()
except  NameError:
    print("EL objeto ya no existe")
```

Ejemplo N°3:

Clase: circulo

Atributo: radio

Acción: calcular_area()

Objeto: circulo = Circulo

```
import math
class Circulo:
    def __init__(self,radio):
        self.radio = radio
        print("Objeto circulo creado")

    def calcular_area(self):
        area = math.pi*self.radio**2
        return area

radio_usuario = float(input("Ingrese el radio del circulo"))

circulo = Circulo(radio_usuario)
resultado = circulo.calcular_area()
print(f"El area del circulo es {resultado}")

del circulo

try:
    circulo.calcular_area()
except NameError:
    print("El objeto ya no existe")
```

Ejemplo N°4:

Clase: producto

Atributo: nombre, precio, cantidad

Acción: cuandos productos se desea mostrar

Objeto:producto =Producto

```
import gc

class Producto:
    def __init__(self, nombre, precio, cantidad):
        self.nombre = nombre
        self.precio = precio
        self.cantidad = cantidad
        print(f"\nProducto registrado: {self.nombre} - ${self.precio} en stock {self.cantidad}")

    def mostrar_informacion(self):
        print(f"{self.nombre} precio ${self.precio:.2f} en stock {self.cantidad}")
```

```
    def __del__(self):
        print(f"Producto eliminado: {self.nombre}")


inventario = []

cantidad = int(input("¿Cuántos productos deseas registrar? "))

for i in range(cantidad):
    print(f"\nRegistro del producto {i+1}:")
    nombre = input("Nombre: ")
    precio = float(input("Precio: "))
    cantidad_stock = int(input("Cantidad en stock: "))

    producto = Producto(nombre, precio, cantidad_stock)
    producto.mostrar_informacion()
    inventario.append(producto)

# Liberar memoria
inventario.clear()
del producto
gc.collect()

print("\nFin de programa")
```

Ejemplo N°5:

Clase:estudiante

Atributo: nombre,edad carrera

Acción:

Objeto:

```
import gc

class Estudiante:
    def __init__(self, nombre, edad, carrera):
        self.nombre = nombre
        self.edad = edad
        self.carrera = carrera
        print(f"Estudiante registrado {self.nombre}. {self.edad} años {self.carrera}")

    def mostrar_informacion(self):
        print(f"{self.nombre} estudia {self.carrera} y tiene {self.edad} años")

    def __del__(self):
        print(f"Estudiante eliminado {self.nombre}")
```

```
grupo = []

cantidad = int(input("¿Cuántos estudiantes deseas registrar? "))

for i in range(cantidad):
   print(f"\nRegistro del estudiante {i+1}:")
   nombre = input("Nombre: ")
   edad = int(input("Edad: "))
   carrera = input("Carrera: ")

   estudiante = Estudiante(nombre, edad, carrera)
   estudiante.mostrar_informacion()
   grupo.append(estudiante)

grupo.clear()
gc.collect()

print("\nFin de programa")
```

Ejemplo N°6:

Clase:libro

Atributo: titulo,autor, año

Acción:

Objeto:

```
import gc
class Libro:
   def __init__(self,titulo,autor,anio):
      self.titulo = titulo
      self.autor = autor
      self.anio = anio
      print(f"Libro registrado {self.titulo} de {self.autor} {self.anio}")

   def mostrar_informacion(self):
       print(f"{self.titulo} fue escrito por {self.autor} en {self.anio}")

   def __del__(self):
      print(f"Libro eliminado {self.titulo}")


biblioteca = []

cantidad = int(input("¿Cuántos libros deseas registrar? "))

for i in range(cantidad):
   print(f"\nRegistro del libro {i+1}:")
   titulo = input("Título: ")
   autor = input("Autor: ")
```

```
    anio = int(input("Año de publicación: "))

    libro = Libro(titulo, autor, anio)
    libro.mostrar_informacion()
    biblioteca.append(libro)

biblioteca.clear()
del libro
gc.collect()

print("Fin de programa")
```