| **Asignatura** | **Programación II** |
| --- | --- |
| **Carrera** | **Ing. en Informática** |
| **Plan** | **Ajuste 2023** |
| **Ciclo** | **2do** |
| **Cuatrimestre** | **1ero** |
| **Tema/Título** | **Practica 4** |
| **Profesor** | **Adrian Tozzi** |

**Grupo de Trabajo**

| **ID/Matrícula** | **APELLIDO, Nombres** | **Correo Electrónico** |
| --- | --- | --- |
| **000-20-2323** | Cannizzaro, Pablo | Pablom.cannizzaro@comunidad.ub.edu.ar |
| **000-20-1394** | Callizaya, Leandro | [leandroa.callizaya@comunidad.ub.edu.ar](mailto:leandro.callizaya@comunidad.ub.edu.ar) |
| **000-20-1212** | Franco Dalla Via Oliveros | Franco.dalla@comunidad.ub.edu.ar |

**Grilla de calificación**

| **Concepto** | **Propuesta** | **Marco Teórico** | **Desarrollo propio** | **Conclusiones** | **Fuentes y Referencias** |
| --- | --- | --- | --- | --- | --- |
| **Sobresaliente (10)** |  |  |  |  |  |
| **Distinguido (9-8)** |  |  |  |  |  |
| **Bueno (7-6)** |  |  |  |  |  |
| **Aprobado (5-4)** |  |  |  |  |  |
| **Insuficiente (3-2-1)** |  |  |  |  |  |
| **Reprobado (0)** |  |  |  |  |  |
| **NOTA** |  | | | | |

| Comentario adicional del Profesor: |
| --- |

**PRÁCTICA 4**

**EJERCICIO 1:**

// Clase abstracta base

abstract static class Empleado {

protected int legajo;

protected String nombre;

protected String apellido;

public Empleado(int legajo, String nombre, String apellido) {

this.legajo = legajo;

this.nombre = nombre;

this.apellido = apellido;

}

public abstract double calcularSueldo();

public void mostrarInformacion() {

System.*out*.println("Legajo: " + legajo);

System.*out*.println("Nombre: " + nombre + " " + apellido);

}

}

// Empleado Asalariado

static class EmpleadoAsalariado extends Empleado {

private double sueldoMensual;

public EmpleadoAsalariado(int legajo, String nombre, String apellido, double sueldoMensual) {

super(legajo, nombre, apellido);

this.sueldoMensual = sueldoMensual;

}

@Override

public double calcularSueldo() {

return sueldoMensual;

}

public double calcularSueldo(String dato) {

if (dato.toLowerCase().contains("bono")) {

return sueldoMensual + 5000;

}

return sueldoMensual;

}

@Override

public void mostrarInformacion() {

super.mostrarInformacion();

System.*out*.println("Tipo: Asalariado");

System.*out*.println("Sueldo base: $" + sueldoMensual);

}

}

// Empleado Por Horas

static class EmpleadoPorHoras extends Empleado {

private int horas;

private double valorHora;

public EmpleadoPorHoras(int legajo, String nombre, String apellido, int horas, double valorHora) {

super(legajo, nombre, apellido);

this.horas = horas;

this.valorHora = valorHora;

}

@Override

public double calcularSueldo() {

return horas \* valorHora;

}

public double calcularSueldo(String dato) {

if (dato.toLowerCase().contains("extra")) {

return (horas + 10) \* valorHora;

}

return calcularSueldo();

}

@Override

public void mostrarInformacion() {

super.mostrarInformacion();

System.*out*.println("Tipo: Por horas");

System.*out*.println("Horas trabajadas: " + horas);

System.*out*.println("Valor hora: $" + valorHora);

}

}

// Empleado Por Comisión

static class EmpleadoPorComision extends Empleado {

private double ventas;

private double tasa;

public EmpleadoPorComision(int legajo, String nombre, String apellido, double ventas, double tasa) {

super(legajo, nombre, apellido);

this.ventas = ventas;

this.tasa = tasa;

}

@Override

public double calcularSueldo() {

return ventas \* tasa;

}

public double calcularSueldo(String dato) {

if (dato.toLowerCase().contains("doble")) {

return calcularSueldo() \* 2;

}

return calcularSueldo();

}

@Override

public void mostrarInformacion() {

super.mostrarInformacion();

System.*out*.println("Tipo: Por comisión");

System.*out*.println("Ventas: $" + ventas);

System.*out*.println("Tasa: " + (tasa \* 100) + "%");

}

}

// Empleado Base + Comisión

static class EmpleadoBaseMasComision extends Empleado {

private double base;

private double ventas;

private double tasa;

public EmpleadoBaseMasComision(int legajo, String nombre, String apellido, double base, double ventas, double tasa) {

super(legajo, nombre, apellido);

this.base = base;

this.ventas = ventas;

this.tasa = tasa;

}

@Override

public double calcularSueldo() {

return base + (ventas \* tasa);

}

public double calcularSueldo(String dato) {

if (dato.toLowerCase().contains("incentivo")) {

return calcularSueldo() + 8000;

}

return calcularSueldo();

}

@Override

public void mostrarInformacion() {

super.mostrarInformacion();

System.*out*.println("Tipo: Base + comisión");

System.*out*.println("Base: $" + base);

System.*out*.println("Ventas: $" + ventas);

System.*out*.println("Tasa: " + (tasa \* 100) + "%");

}

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.*in*);

Empleado[] empleados = new Empleado[4];

empleados[0] = new EmpleadoAsalariado(1, "Ana", "Martínez", 120000);

empleados[1] = new EmpleadoPorHoras(2, "Luis", "Gómez", 160, 900);

empleados[2] = new EmpleadoPorComision(3, "Carlos", "López", 300000, 0.05);

empleados[3] = new EmpleadoBaseMasComision(4, "Sofía", "Fernández", 80000, 150000, 0.03);

for (Empleado emp : empleados) {

emp.mostrarInformacion();

System.*out*.print("Ingrese palabra clave (bono, extra, incentivo, doble): ");

String input = sc.nextLine();

double sueldoFinal;

if (emp instanceof EmpleadoAsalariado) {

sueldoFinal = ((EmpleadoAsalariado) emp).calcularSueldo(input);

} else if (emp instanceof EmpleadoPorHoras) {

sueldoFinal = ((EmpleadoPorHoras) emp).calcularSueldo(input);

} else if (emp instanceof EmpleadoPorComision) {

sueldoFinal = ((EmpleadoPorComision) emp).calcularSueldo(input);

} else if (emp instanceof EmpleadoBaseMasComision) {

sueldoFinal = ((EmpleadoBaseMasComision) emp).calcularSueldo(input);

} else {

sueldoFinal = emp.calcularSueldo();

}

System.*out*.println("Sueldo final calculado: $" + sueldoFinal);

System.*out*.println("------------------------------------------------");

}

sc.close();

}

**Consola:**

Legajo: 1

Nombre: Ana Martínez

Tipo: Asalariado

Sueldo base: $120000.0

Ingrese palabra clave (bono, extra, incentivo, doble): BONO

Sueldo final calculado: $125000.0

------------------------------------------------

Legajo: 2

Nombre: Luis Gómez

Tipo: Por horas

Horas trabajadas: 160

Valor hora: $900.0

Ingrese palabra clave (bono, extra, incentivo, doble): EXTRA

Sueldo final calculado: $153000.0

------------------------------------------------

Legajo: 3

Nombre: Carlos López

Tipo: Por comisión

Ventas: $300000.0

Tasa: 5.0%

Ingrese palabra clave (bono, extra, incentivo, doble): INCENTIVO

Sueldo final calculado: $15000.0

------------------------------------------------

Legajo: 4

Nombre: Sofía Fernández

Tipo: Base + comisión

Base: $80000.0

Ventas: $150000.0

Tasa: 3.0%

Ingrese palabra clave (bono, extra, incentivo, doble): DOBLE

Sueldo final calculado: $84500.0

------------------------------------------------

Process finished with exit code 0

**EJERCICIO 2:**

abstract class Poligono {

public String color;

public long altura;

public long base;

abstract long area();

abstract void pintar(String color);

}

class Rectangulo extends Poligono {

public Rectangulo( long x, long y) {

altura = x;

base = y;

}

public void pintar( String color) {

System.out.println("Rectangulo color " + color);

}

public long area() {

return base \* altura;

}

}

class Cuadrado extends Poligono{

public Cuadrado (long x) {

base = x;

altura = x;

}

public void pintar( String color) {

System.out.println("cuadrado color " + color);

}

public long area() {

return base \* altura;

}

}

public class EjercicioPoligono {

public static void main(String[] args) {

Rectangulo rect = new Rectangulo(10, 5);

pintarObjeto(rect, "Rojo");

Cuadrado cuad = new Cuadrado(4);

long i = areaObjeto(cuad);

System.out.println("El area del cuadrado es " + i);

}

public static void pintarObjeto(Poligono p, String c) {

p.pintar(c);

}

public static long areaObjeto(Poligono p) {

return p.area();

}

}

**Lo que imprime este programa es:**

Rectángulo color Rojo

El área del cuadrado es 16

**EJERCICIO 3:**

### **Elección múltiple:**

1. Si una clase contiene al menos un método abstracto es una clase \_\_\_\_\_\_\_\_\_\_\_.  
    **Respuesta: abstracta**
2. Las clases a partir de las cuales pueden instanciarse objetos se llaman \_\_\_\_\_\_\_\_\_\_\_.  
    **Respuesta: concretas**
3. \_\_\_\_\_\_\_\_\_\_\_ implica el uso de una variable de superclase para invocar métodos en objetos de la superclase y subclase.  
    **Respuesta: El polimorfismo**
4. Los métodos abstractos se declaran utilizando la palabra clave \_\_\_\_\_\_\_\_\_\_\_.  
    **Respuesta: abstract**
5. En una interfaz no se pueden declarar \_\_\_\_\_\_\_\_\_\_\_.  
    **Respuesta: variables de instancia**

### **Verdadero o falso:**

1. Todos los métodos de una clase abstracta deben ser declarados como métodos abstract.  
    **FALSO : Puede tener métodos concretos (implementados).**
2. Una clase abstracta debe tener por lo menos un método abstracto.  
    **FALSO : Puede no tener ninguno (pero no se puede instanciar igualmente).**
3. Una clase se hace abstracta declarándose como abstract.  
    **VERDADERO**
4. Si una superclase declara un método como abstract, una subclase debe implementar ese método para convertirse en clase concreta.  
    **VERDADERO**
5. Una interfaz es un conjunto de declaraciones de constantes y métodos abstractos.  
    **VERDADERO**
6. Una clase puede implementar una sola interfaz.  
    **FALSO : Puede implementar varias interface**s.
7. Una interfaz puede ser implementada por muchas clases.  
    **VERDADERO**