	**Carátula para entrega de prácticas**
Facultad de Ingeniería	Ingeniería en Computación

Profesor(a): _____

Asignatura: Programación Orientada a Objetos

Grupo: 1

No de Práctica(s): 8

Integrante(s): 11700202-9

_____

_____

_____

No. de Equipo de cómputo empleado: _____

Semestre: 2024-1

Fecha de entrega: 29 de octubre de 2023

Observaciones: _____

_____

CALIFICACIÓN: _____

Index.

Practice 8 - Inheritance and Polymorphism.

Calculator.

1.- Introduction.

This report has the objective of explaining the eighth practice of the Object oriented programming subject, in which we must explore inheritance and polymorphism concepts again, typical of the object oriented paradigm.

We must build a calculator with two modes, decimal and binary. The decimal must perform the following operations: Factorial, middle point between two lines, solutions of a quadratic equation and a transposed matrix. The binary must perform the following operations: rest, division, A1 complement, A2 complement.

1.1.- Hypothesis

We have to create an abstract class Calculator, because we don't need to declare an instance of this father class, and then inherit two classes: BinaryCalculator and DecimalCalculator. As the methods are quite different, for polymorphism we will use a randomOperation method.

2.- Development

2.1.- "Calculadora" class

We need only one attribute, the memory:

- Mem (Long integer)

We define it's setter and getter:

```
public long getMem(){return this.mem;}

public void setMem(long mem){this.mem = mem;}
```

Builder: Default.

We need only one method, the random operation.

randomOperation (Abstract, void): Chooses an option from the calculator methods and displays it on the screen.

<<Abstract>> **Class Calculadora**
- Mem (Long integer)
- **randomOperation (Abstract, void)**

2.2.1.- Inherited classes

We need to define the specific methods of each class.

CalculadoraDecimal

Factorial (Long integer): Calculates the factorial recursively multiplying the current number by (n-1) until we get to our base case (1).

Chicharronera (Void): Implements the Bhaskara 's formula, using doubles, handling the complex root case.

Create (Matrix of integer): Generates a matrix of random numbers of the order given by the user.

PrintM (Void): Prints a nxn matrix

PuntoMedio (Void): By dividing the rest of the coordinates by two, calculates the middle point between two lines.

(Overridden) randomOperation.

<<Abstract>> **Class CalculadoraDecimal**
- Mem (Long integer)
- **randomOperation (Abstract, void)** - **Factorial (Long integer)** - **Chicharronera (Void)** - **Create (Matrix of integer)** - **PrintM (Void)** - **PuntoMedio (Void)**

CalculadoraBinaria

CreateBin (Integer): Converts a decimal number to a binary number.

RestarBin (Integer): Rest two numbers in their binary representation by operating the least significant bit.

DividirBin (Integer): Divides the numbers in their binary representation by using a for loop and bit shifts.

ComplementoUno (Integer): Makes the NOT operation to the number in its decimal representation.

ComplementoDos (Integer): Makes the NOT operation to the number in its decimal representation and adds one to the result.

(Overridden) randomOperation.

2.3.- Main class.

We need to initialize two calculators and set their mem attribute to zero. Then make some variables for menu choices and validations between them.

The menu is divided in two layers using two loops in which the user decides when wants to exit. The first layer contains the calculator menu and the second the options of each calculator.

In each iteration we ask the user if they wants to maintain the memory or overwrite it (doesn't apply for matrices and float number operations) and let the user make operations with that value which it's now stored in the object attribute.

Conclusion: The hypothesis was correct, but with some issues, we could make the inheritance more relevant just declaring n operation methods for the calculators and just make polymorphic functions instead of creating an additional function, also for menus, avoiding programming the menus on the main class (two times).

The objective was completed and the calculator works, but we're not discarding extending its functions in some future.

References:

- [1] Oracle. (2023, July). Java API, Math Class [Online]. Available: https://docs.oracle.com/javase/8/docs/api/java/lang/Math.html