



Facultad de Ingeniería UNAM



Maestro: Ing. Guadalupe Lizeth Parrales Romay

Asignatura: Programación Orientada a Objetos

Práctica 3

Alumno:

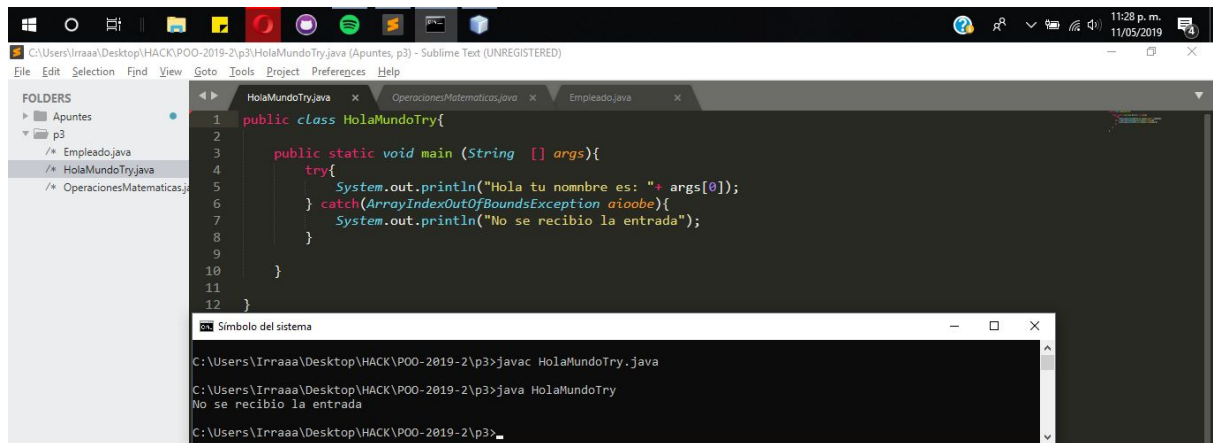
Mejía Alba Israel Hipólito

Grupo: 2

Actividades:

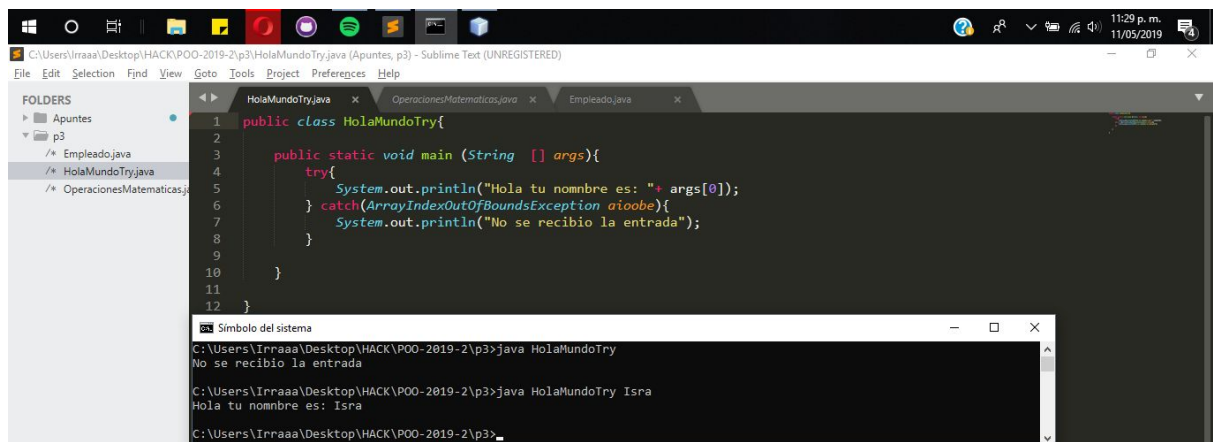
HolaMundo.java

Podemos ver como mediante las excepciones podemos notificarle al usuario a consola que no ingresó ningún valor para poderlo saludarlo con su nombre. Esto debido que se hace uso de la excepción `ArrayIndexOutOfBoundsException` la cual nos dice que el `args[0]` está fuera del alcance del arreglo dinámico de la clase principal, en otras palabras está vacío porque no se recibió nada. En caso de que si ingrese su nombre en la terminal, se le saluda como en la segunda captura de pantalla



```
public class HolaMundoTry{
    public static void main (String [] args){
        try{
            System.out.println("Hola tu nombre es: "+ args[0]);
        } catch(ArrayIndexOutOfBoundsException aioobe){
            System.out.println("No se recibio la entrada");
        }
    }
}
```

```
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>javac HolaMundoTry.java
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>java HolaMundoTry
No se recibio la entrada
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>
```



```
public class HolaMundoTry{
    public static void main (String [] args){
        try{
            System.out.println("Hola tu nombre es: "+ args[0]);
        } catch(ArrayIndexOutOfBoundsException aioobe){
            System.out.println("No se recibio la entrada");
        }
    }
}
```

```
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>java HolaMundoTry
No se recibio la entrada
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>java HolaMundoTry Isra
Hola tu nombre es: Isra
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>
```

OperacionesMatematicas.java:

Se realiza las operaciones matemáticas necesarias para poder realizar la famosa fórmula “chicharronera”, donde se avisa que los datos ingresados nos dan valores de x fuera de los números reales, pero de igual forma se realiza operacionesImaginarios el cual saca la parte real y opera para poderle dar resultados al usuario.

Se hace uso de excepciones para mandarle distintos mensajes al usuario como se ve en el código, como por ejemplo cuando no ingresa un número.

```
1 import java.util.Scanner;
2
3 public class OperacionesMatematicas{
4
5     public static void main ( String [] args){
6
7         System.out.println("\n\tSe programara la chicharroneras");
8
9         Scanner num = new Scanner(System.in);
10
11         try{
12             System.out.println("Ingresa sus numeros");
13             System.out.println("Introduce el coeficiente a de ax^2");
14             double a = num.nextDouble();
15
16             System.out.println("Introduce el coeficiente b de bx");
17             double b = num.nextDouble();
18
19             System.out.println("Introduce a c");
20             double c = num.nextDouble();
21
22             boolean res = discriminante(a,b,c);
23
24             if (res){
25                 operacion(a,b,c);
26             }else{
27                 operacionImaginarios(a,b,c);
28             }
29         }catch(Exception e){
30             System.err.println("No se ingreso un numero, vuelvalo a ingresar");
31         }
32     }
33 }
```

```
34 public static boolean discriminante(double a, double b, double c){
35     double cuadrado = Math.pow(b,2);
36     double resultado = (cuadrado - (4*a*c));
37     boolean res = (resultado >= 0) ? true : false;
38     return res;
39 }
40 /*Para ver si es posible realizar la operacion*/
41
42 public static void operacion(double a, double b, double c){
43     double cuadrado = Math.pow(b,2);
44     double r1 = ((-b) + Math.sqrt((cuadrado - (4*a*c)))) / (2*a);
45     double r2 = ((-b) - Math.sqrt((cuadrado - (4*a*c)))) / (2*a);
46
47     System.out.println("El primer valor de x es de : "+ r1);
48     System.out.println("El sgundo valor de x es de : "+ r2);
49     /*operaciones de la chicharronera*/
50
51 }
52
53 public static void operacionImaginarios(double a, double b, double c){
54     System.out.println("\nLos numeros ingresados no dan como resultado dos numeros reales, ");
55     System.out.println("pero puedo operarlos para darte un resultado ;)\n");
56     double cuadrado = Math.pow(b,2);
57     double real = ((-b)/(2*a));
58     double imaginaria = (-1)*((cuadrado - (4*a*c)))/(2*a);
59     //Se convierte positiva la parte imaginaria
60     imaginaria = (Math.sqrt(imaginaria));
61     System.out.println("Parte Real: "+ real);
62     System.out.println("Parte imaginaria: +/- "+ imaginaria + "i");
63 }
64 /*Se saca su parte real para poderla operar y la imaginaria se multiplica por -1 para operar*/
```

```
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>java OperacionesMatematicas
Se programara la chicharroneras
Ingresa sus numeros
Introduce el coeficiente a de ax^2
1
Introduce el coeficiente b de bx
2
Introduce a c
3
Los numeros ingresados no dan como resultado dos numeros reales,
pero puedo operarlos para darte un resultado ;)
Parte Real: -1.0
Parte imaginaria: +/- 2.0i
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>
```

```
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>java OperacionesMatematicas
Se programara la chicharroneras
Ingresa sus numeros
Introduce el coeficiente a de ax^2
1
Introduce el coeficiente b de bx
-5
Introduce a c
6
El primer valor de x es de : 3.0
El sgundo valor de x es de : 2.0
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>
```

```
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>java OperacionesMatematicas

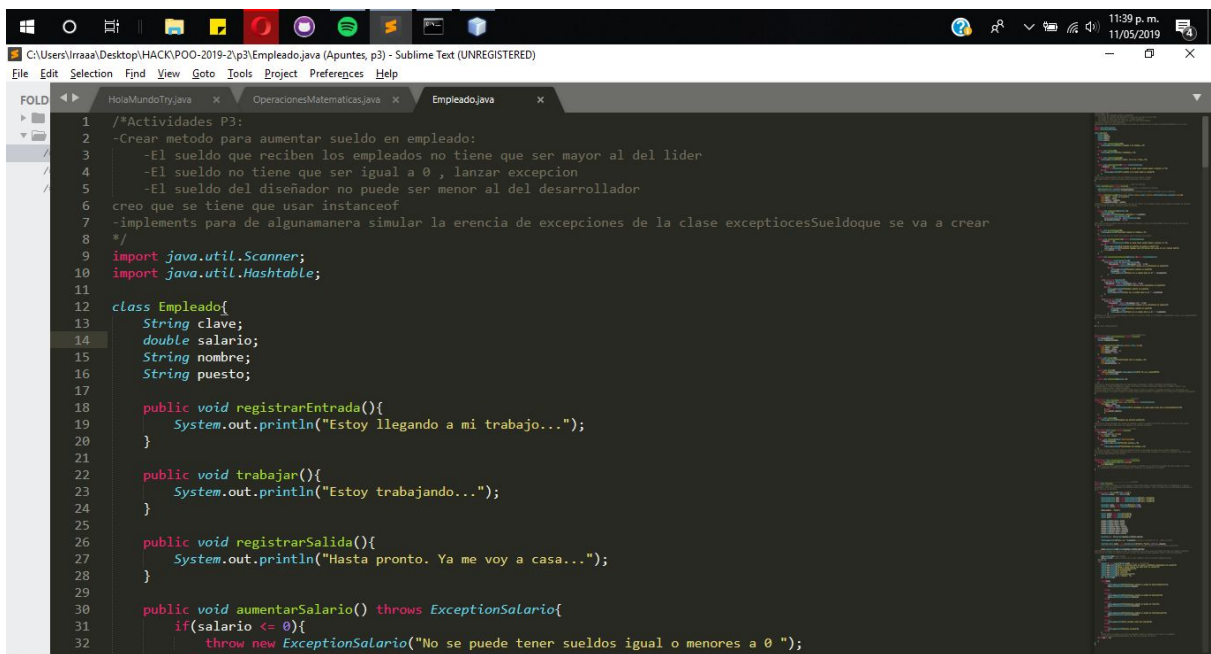
Se programara la chicharronerias
Ingrese sus numeros
Introduce el coeficiente a de ax^2
a
No se ingreso un numero, vuelvalo a intentar
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3>
```

```
65     }
66
67 }
```

Empleado.java

Esta clase simula distintos escenarios en un equipo de trabajo, hace uso de distintas herramientas de java como se explico en la práctica anterior, en el caso de esta se hace uso del manejo de excepciones para aumentar o no el salario a distintos empleados del equipo de desarrollo.

Logre hacer el código necesario con su lógica propia, pero al compilarlo no pude resolver los errores que marca en consola, solo fueron 3, pero no supe cómo solucionarlos.



```
C:\Users\Irraaa\Desktop\HACK\POO-2019-2\p3\Empleado.java (Apuntes, p3) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

FOLD
1  /*Actividades P3:
2  -Crear metodo para aumentar sueldo en empleado:
3  -El sueldo que reciben los empleados no tiene que ser mayor al del lider
4  -El sueldo no tiene que ser igual a 0 , lanzar excepcion
5  -El sueldo del diseñador no puede ser menor al del desarrollador
6  creo que se tiene que usar instanceof
7  -implements para de algunamnera simular la erencia de excepciones de la clase excepcionesSueldo que se va a crear
8  */
9  import java.util.Scanner;
10 import java.util.Hashtable;
11
12 class Empleado{
13     String clave;
14     double salario;
15     String nombre;
16     String puesto;
17
18     public void registrarEntrada(){
19         System.out.println("Estoy llegando a mi trabajo...");
20     }
21
22     public void trabajar(){
23         System.out.println("Estoy trabajando...");
24     }
25
26     public void registrarSalida(){
27         System.out.println("Hasta pronto. Ya me voy a casa...");
28     }
29
30     public void aumentarSalario() throws ExceptionSalario{
31         if(salario <= 0){
32             throw new ExceptionSalario("No se puede tener sueldos igual o menores a 0 ");
33         }
34     }
35 }
```

```

33     }else{
34         System.out.println("Se analiza si te puedo subir el sueldo");
35     }
36 }
37 //Se crea la clase empleado con sus atributos de tipo cadena y double.
38 //De igual forma se crean sus metodos para interactuar con el equipo
39 }
40
41 //-----Lider de Proyecto
42 class LiderDeProyecto extends Empleado{
43     //se crea el Lider de proyecto que hereda metodos y atributos de empleado
44     Hashtable<String, Empleado> equipoDeTrabajo;
45     //El lider de proyecto crea la estructura del hashtable del equipo de trabajo con los empleados
46
47     public LiderDeProyecto(String clave, String nombre, double salario, Hashtable<String, Empleado> equipo){
48         this.puesto = "Lider de Proyecto";
49         this.clave = clave;
50         this.salario = salario;
51         this.equipoDeTrabajo = equipo;
52     }
53     //Metodo constructor del liderDeProyecto el cual con "this" se le asignan valores sus atributos heredados de empleado
54     // asi como se le asigna su propio equipo de trabajo
55 }
56
57 public void asignarTarea(Empleado e){
58     this.trabajar();
59     System.out.println("Asignando actividad a "+e.nombre);
60     if(e instanceof DesarrolladorJava){
61         DesarrolladorJava dj = (DesarrolladorJava)e;
62         dj.trabajoTerminado = false;
63     }
64 }
65
66 /*Se crea el metodo asignar tarea donde recibe un empleado, si este es desarrollador java se se le dice que aun no
67 ha acabado su trabajo*/

```

```

65 }
66
67 public void convocarReunion(){
68     System.out.println("Convocando reunion de trabajo...");
69 }
70 //De igual forma el lider del proyecto puede convocar una reunion
71
72 public void aumentarseSalario() throws ExceptionSalario{
73     if(salario <= 0){
74         throw new ExceptionSalario("No se puede tener sueldos igual o menores a 0 ");
75     }else{
76         System.out.println("Me deberia de aumentar el sueldo yo mismo? ");
77         System.out.println("Obviamente si!!!!, solo 200 dolares mas porque no voy a abusar jaja");
78         this.salario += 200;
79     }
80 }
81
82 public void aumentarSalarioEmpleados(Empleado e) throws ExceptionSalario{
83
84     if(e instanceof DesarrolladorJava){
85         DesarrolladorJava dj= (DesarrolladorJava)e;
86         if(dj.salario > 1500 || (dj.salario*100) > 1500){
87             throw new ExceptionSalario("Lo siento, no me autorizaron el aumento");
88         }else{
89             System.out.println("Intentare subirte el sueldo");
90             dj.salario += 100;
91             System.out.println("Listo :D, tu sueldo ahora es de " + dj.salario);
92         }
93     }
94
95     if( e instanceof Diseñador){
96         Diseñador d= (Diseñador)e;
97         if(d.salario > 1500 || (d.salario*100) > 1500){
98             throw new ExceptionSalario("Lo siento, no me autorizaron el aumento");
99         }else{
100             System.out.println("Intentare subirte el sueldo");
101             d.salario += 100;
102             System.out.println("Listo :D, tu sueldo ahora es de " + d.salario);
103         }
104     }
105
106     if( e instanceof Tester){
107         Tester t= (Tester)e;
108         if(t.salario > 1500 || (t.salario*100) > 1500){
109             throw new ExceptionSalario("Lo siento, no me autorizaron el aumento");
110         }else{
111             System.out.println("Intentare subirte el sueldo");
112             t.salario += 100;
113             System.out.println("Listo :D, tu sueldo ahora es de " + t.salario);
114         }
115     }
116 }
117
118 /*Metodo en el que al invocarlo mediante el lider de proyecto recibe a un empleado y dependiendo a este y sus características
119 se le sube su sueldo o no
120 */
121 }
122
123 }
124
125 //fin class LiderDeProyecto
126
127 //-----DesarrolladorJava
128 public static class DesarrolladorJava extends Empleado{

```

```

95     if( e instanceof Diseñador){
96         Diseñador d= (Diseñador)e;
97         if(d.salario > 1500 || (d.salario*100) > 1500){
98             throw new ExceptionSalario("Lo siento, no me autorizaron el aumento");
99         }else{
100             System.out.println("Intentare subirte el sueldo");
101             d.salario += 100;
102             System.out.println("Listo :D, tu sueldo ahora es de " + d.salario);
103         }
104     }
105
106     if( e instanceof Tester){
107         Tester t= (Tester)e;
108         if(t.salario > 1500 || (t.salario*100) > 1500){
109             throw new ExceptionSalario("Lo siento, no me autorizaron el aumento");
110         }else{
111             System.out.println("Intentare subirte el sueldo");
112             t.salario += 100;
113             System.out.println("Listo :D, tu sueldo ahora es de " + t.salario);
114         }
115     }
116 }
117
118 /*Metodo en el que al invocarlo mediante el lider de proyecto recibe a un empleado y dependiendo a este y sus características
119 se le sube su sueldo o no
120 */
121 }
122
123 }
124
125 //fin class LiderDeProyecto
126
127 //-----DesarrolladorJava
128 public static class DesarrolladorJava extends Empleado{

```



```

127 public static class DesarrolladorJava extends Empleado{
128     int horasSinDormir;
129     boolean trabajoTerminado;
130
131
132     public DesarrolladorJava(String nombre, String clave){
133         this.nombre = nombre;
134         this.clave = clave;
135         this.horasSinDormir = 0;
136         this.salario = 1000;
137     }
138
139     public void trabajar(){
140         System.out.println("Transformando café en energía...");
141         super.trabajar();
142         this.horasSinDormir ++;
143     }
144
145     public void dormir(){
146         if(this.trabajoTerminado) System.out.println("Al fin voy a dormir!!!");
147         else this.trabajar();
148     }
149
150     public void SubirSueldo(Empleado e){
151     }
152
153 }
154 /*Se crea la clase DesarrolladorJava el cual hereda de empleado y tiene 2 atributos adicionales como
155 horasSinDormir y trabajoTerminado. De igual forma tiene su metodo constructor donde se le asignan valores a sus
156 atributos nombre, clave y horas sin dormir.
157 De igual forma tiene sus metodos trabajar donde manda a llamar su metodo de empleado trabajar y se incrementan sus
158 horas sin dormir. Por ultimo contiene su metodo dormir donde si acabo su trabajo se ire a dormir si no seguira trabajando*/

```

```

159 }
160
161 //-----Diseñador
162 public static class Diseñador extends Empleado{
163     public Diseñador(String clave, int salario) throws ExceptionSalario{
164         this.clave = clave;
165         if(salario < 1000){
166             throw new ExceptionSalario("Soy diseñador, no puedo ganar menos que un DesarrolladorJava");
167         }else{
168             this.salario = salario;
169         }
170     }
171
172     public void trabajar(){
173         System.out.println("Diseñando una interfaz gráfica");
174     }
175
176 /*Se crea la clase diseñador que hereda de empleado y tiene su metodo constructor donde se le asigna con una cadena
177 su clave y otro metodo donde dice que trabaja en la interfaz grafica*/
178 }
179 //-----Tester
180 public static class Tester extends Empleado{
181     int salario = 900;
182     public Tester(String clave){
183         this.clave = clave;
184     }
185     public void trabajar(boolean faseTerminada){
186         if(faseTerminada){
187             System.out.println("Haciendo pruebas...");
188         }else{
189             System.out.println("Planificando las pruebas...");
190         }
191     }
192 }

```

```

191 /* Se crea la clase Tester que hereda de empleado donde se le asigna su clave con su metodo constructor.
192 Por ultimo contiene su metodo trabajar donde se crea su boolean fase terminada la cual si es verdadera hace las pruebas
193 y si no esta planificando las mismas*/
194 }
195
196 //-----Exception
197 public static class ExceptionSalario extends Exception{
198     public ExceptionSalario (String mensaje){
199         super(mensaje);
200     }
201     /*Se crea la clase ExceptionSalario que hereda de Exption con el cual con el metodo del mismo nombre se imprime
202     // a pantalla el mensaje que se decida poner respecto a la excepcion en el codigo
203     }
204
205 //-----Proyecto
206 public class Proyecto{
207     /* Se crea la clase Proyecto, el cual contiene el hash table equipo, 3 desarrolladores java, 2 diseñadores y 3 tester.
208     Se agregan a todos los integrantes al hashtable equipo con el metodo ".put", me parece que los diseñadores comparten la
209     misma clave en el Hashtable
210     */
211     public static void main(String[] args) {
212         Hashtable equipo = new Hashtable();
213
214         DesarrolladorJava hugo = new DesarrolladorJava("Hugo", "devj01");
215         DesarrolladorJava paco = new DesarrolladorJava("Paco", "devj02");
216         DesarrolladorJava luis = new DesarrolladorJava("Luis", "devj03");
217
218         Diseñador daisy = new Diseñador("des01", 1100);

```


Conclusión:

Durante la práctica se lograron entender de manera teórica el uso de excepciones y se repaso muy bien aspectos del api de java visto en las clases y en las prácticas anteriores.

Logre realizar correctamente las dos primeras actividades, pero en la última ya no pude corregir los errores que me aparecieron, estuve días analizandolos y en ocasiones generaba más. Me gustaria ver como se tenía que hacer este ejercicio en clase ya que puede que a muchos les pase lo mismo.

Dejando a un lado mis errores en el último ejercicio, siento que después de las clases, prácticas y búsquedas en internet realizadas, logre comprender como se tiene que hacer el uso de excepciones y distintas herramientas de java por lo que considero que fue de gran provecho esta clase.