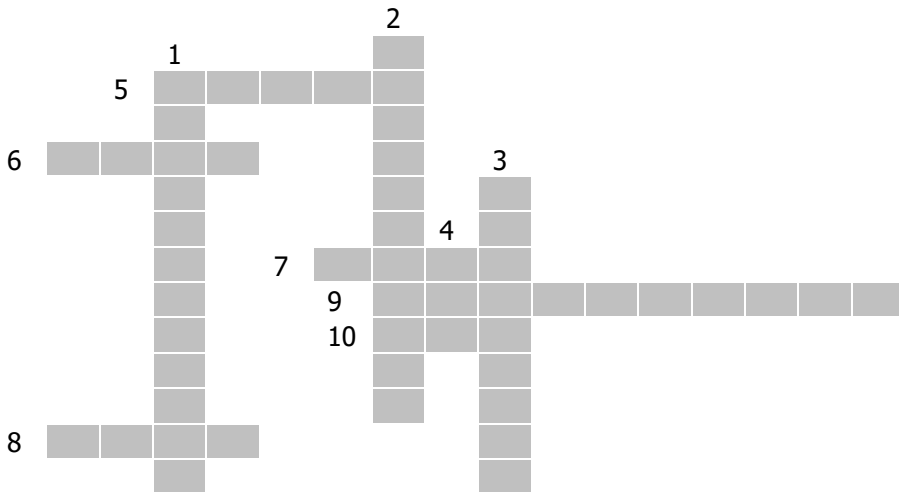

Modulo Java

Guia de Ejercicios Parte

1

Módulo 1: El lenguaje Java

Ejercicio I



Verticales:

- 1.- Palabra reservada en Java
- 2.- Acción de crear una variable (Sust.)
- 3.- Software con funcionalidad genérica, utilizado por los programadores para construir aplicaciones
- 4.- Video On Demand

Horizontales:

- 5.- Se llama así al lugar donde es válida una variable
- 6.- Tipo de datos de 64 bits que representa un entero
- 7.- Archivo importantísimo que es además una máquina (sin la extensión)
- 8.- Tecnología de Java sobre la que es posible construir aplicaciones Web
- 9.- Software que traduce un lenguaje de programación a otro lenguaje
- 10.- Software sobre el cual desarrolla el desarrollador.

Ejercicio II

a.- Crear un nuevo Proyecto Java desde File/New/Project... Java/Java Project. Llamarlo ej2 y clicar 'Finish'

b.- Crear una nueva clase Java con: click derecho en la carpeta src. Ingresar los valores como se ve en el siguiente cuadro de diálogo.

New Java Class

Java Class

The use of the default package is discouraged.

Source folder:

Package:

☐ Enclosing type:

Name:

Modifiers: ☒ public ☐ default ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass:

Interfaces:

Which method stubs would you like to create?

☒ public static void main(String[] args)

☐ Constructors from superclass

☐ Inherited abstract methods


Do you want to add comments? (Configure templates and default value [here](#))

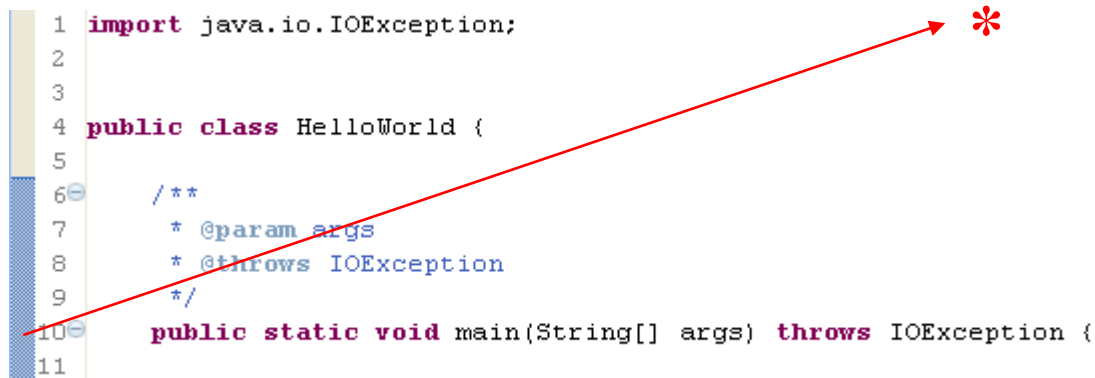
☐ Generate comments

c.-Modificar el método HelloWorld de modo que incluya la siguiente línea:

System.out.println("Hello World");

d.-Correr el ejemplo con el botón  (Run Java Application)

e.-Debuggear la aplicación colocando un Breakpoint con doble click donde se indica (*) y luego correr con el botón 



```
1 import java.io.IOException;
2
3
4 public class HelloWorld {
5
6     /**
7      * @param args
8      * @throws IOException
9      */
10    public static void main(String[] args) throws IOException {
11
```

f.- Modificar ahora el método main para que imprima las tablas de multiplicar del número 1 al 10. Cada tabla debe ir separada por un línea en blanco.

g.- Modificar ahora el método main para que pida por teclado la nota de un alumno e imprima en pantalla la calificación correspondiente según la siguiente tabla:

1. De 0 a 50 puntos -> Suspenso
2. De 51 a 75 puntos -> Recuperar
3. De 76 a 90 puntos -> Aprobado
4. De 91 a 100 puntos -> Aprobado con mérito
5. Menor que 0 o mayor que 100 -> puntuación inválida.

Para tomar el input del teclado, se utiliza el siguiente fragmento de código:

```
System.out.println("Introduzca un Numero: ");
Scanner sc = new Scanner(System.in);
int i = sc.nextInt();
```

Ejercicio III

Modelar en un diagrama de clases utilizando el plugin de Eclipse un sistema de facturación utilizando los conceptos de POO vistos hasta ahora en el curso. Se cuenta únicamente con la siguiente 'Especificación'

El sistema debe procesar órdenes para distintos clientes. De los clientes nos interesa registrar su nombre y dirección. Cada orden tiene una fecha, un status (A PROCESAR y PROCESADA). Una orden se compone de uno o varios detalles. Un detalle es esencialmente un 'renglón' dentro de la orden y tiene una cantidad y un item asociado. Los items son los productos que se pueden ordenar. Para cada item se registra un peso y una descripción. Las órdenes pueden tener una o varias formas de pago, a saber: Tarjeta de crédito, cheque o efectivo. Todas estas formas de pago tienen en común el importe. Si es tarjeta de crédito, registramos el número, el tipo y la fecha de expiración. En caso de efectivo, queremos saber si son dólares o pesos, y la tasa de conversión. Los cheques deben tener el nombre de quien los emite y el ID del banco de donde provienen. El sistema debe permitir calcular el total y el peso total de una orden. El sistema debe poder indicar si una tarjeta de crédito o un cheque están autorizados.

Hints

- Las fechas en Java son de tipo Date:
Date a;

Ejercicio IV

a.- Crear un nuevo proyecto Java llamado ej4

b.- Crear un nuevo tipo de datos Pila en un nuevo paquete com.empresa.training.java.ej2, que vamos a implementar sobre un arreglo de tamaño variable de Java que se llama ArrayList.

Se busca implementar las siguientes operaciones:

pop()	Quita el tope de la pila y lo devuelve (pop=pick on pile)
push(e)	Inserta un elemento e en el tope de la pila
peek()	Devuelve el tope de la pila, pero sin quitarlo
reverse()	Invierte todos los elementos de la pila

Hints:

- Crear una clase llamada MiPila, que adentro tenga un vector de char

```
public class MiPila{  
    int índice = 0;  
    char[] pila=new char[6];
```

```
    public void push(char c){  
        // ... agregar "c" a pila  
        // incrementar el indice  
    }
```

```
    Public char pop(){
```

```
        return pila[indice];  
    }
```

- El array se crea así, como variable privada de la clase:

```
private List arreglo = new ArrayList();
```

- Los parámetros deberían ser de tipo 'Object'
- Se itera sobre un arreglo así:

```
for (Object o: arreglo) {  
    o.toString(); // por decir algo  
    ... //trabajar con el objeto  
}
```

- Otros métodos útiles son add(), set() y get(). Notar que no se puede hacer un set sin antes hacer un add.

Ejercicio V

Supongamos que una orquesta puede dar un concierto. Esta orquesta tiene un conjunto de instrumentos musicales, que debe hacer sonar, uno por uno. Una guitarra y un piano son instrumentos musicales. Representaremos el sonido de cada instrumento como un string. Por ejemplo, guitarra.sonar() devuelve "guitarra.sonar()";¹. El resultado del concierto también es un string.

¹ La calidad musical de un concierto así es discutible.

Implementar en Java las clases Orquesta, Guitarra y Piano, de forma que cuando la orquesta dé un concierto, el resultado aparezca en el cuadro de diálogo de la aplicación.Cuál es la forma más adecuada de definir un instrumento musical?

Hint: Cuando decimos que una orquesta tiene un conjunto de instrumentos musicales, quiere decir que tiene un ArrayList de ellos:

```
private ArrayList instrumentos = new ArrayList();
```

Ejercicio VI

Mejorar el diseño de esta horrrrrrible clase, con los contenidos vistos el día de hoy.

```
public class ItemInventario {

    enum TipoItem {
        PC, SILLA
    };

    // El tipo de item que será este objeto
    final TipoItem tipo;

    // Solo se usa si es una PC
    public int nroSerie;
    public boolean esNotebook;

    // Solo se usa si es una SILLA
    public boolean conRueditas;
    public int lote;
    public int numeroDentroDeLote;

    // Constructor para PC
    public ItemInventario(int nroSerie, boolean esNotebook) {
        tipo = TipoItem.PC;
        this.esNotebook = esNotebook;
        this.nroSerie = nroSerie;
    }

    // Constructor para SILLA
    public ItemInventario(int lote, int numeroDentroDeLote, boolean conRueditas) {
        tipo = TipoItem.SILLA;
        this.lote = lote;
        this.numeroDentroDeLote = numeroDentroDeLote;
        this.conRueditas = conRueditas;
    }

    public int getID () {
        switch (tipo) {
            case PC:
                return nroSerie;
        }
    }
}
```

```
        case SILLA:  
            return lote * 1000 + numeroDentroDeLote;  
        default:  
            throw new AssertionError();  
    }  
}
```

Ejercicio VI bis

Qué pasa si ahora queremos que algunos items del Inventario, pero no todos (por ejemplo, sólo las PCs). Queremos que estos items sean de un tipo BarcodeReadable, para poder identificar todos los items de Inventario que tiene código de barras. Queremos además que todos tengan un método public int getBarcode()? Cómo se podría hacer?

Ejercicio VII

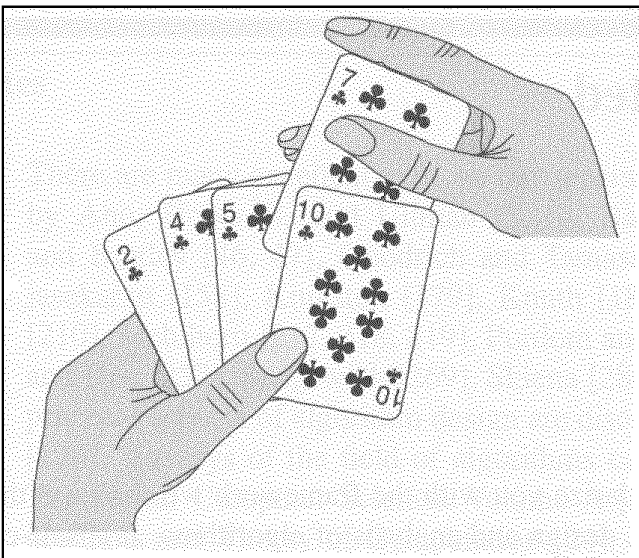
Implementar en Java el siguiente algoritmo para ordenar un arreglo (en este caso, de enteros):

Ordenar

Input: Un arreglo de enteros A

Output: El arreglo A modificado, de forma que sus elementos están ordenados ascendentemente.

Procedimiento:



Ejercicio VII bis

Ordenar un arreglo de n enteros, que sabemos que pueden estar en el rango $0 \dots k$, SIN usar fors anidados, y SIN que los for que se utilicen iteren más de n veces.

Hint:

- Dado un elemento del arreglo, es útil saber cuántos elementos menores tengo...
- Conviene usar un arreglo auxiliar para guardar el arreglo ordenado.

Ejercicio VIII

1.- Crear una clase A con dos métodos, `void a()` y `void b()`. Crear además una clase C con un método main que llame al método `a`;

2.- La implementación de `void a()` debe solamente lanzar una checked exception **ExceptionA** definida por nosotros. Qué nos obliga a hacer el compilador en el método `a()` y en el main de C?

3.- La implementación de `void b()` debe solamente lanzar una unchecked exception **ExceptionB** también definida por nosotros. Cambiar el main para que llame a `void b()` en lugar de `void a()`. Qué pasa ahora con el compilador? Se calienta?

Ejercicio IX

Dada la siguiente clase, implementar los métodos `equals` y `hashCode`. Asumimos que dos empleados son iguales si se llaman igual, tienen la misma edad, el mismo salario y fueron contratados el mismo día. (Sí, sería mejor que tuvieran un número de legajo, pero asumamos que no lo hay).

```
public class Employee {  
    private Date fechaIngreso;  
    private double salario;  
    private String nombre;  
  
    //getters y setters  
}
```

Ejercicio X

1.- Modificar la implementación de la Pila del Ej 4 para que utilice Generics.

2.- Crear una enumeration Class que contenga la siguiente información, para los siguientes planetas del sistema solar:

Nombre	Masa	Radio
MERCURIO	3.303e+23	2.4397e6
VENUS	4.869e+24	6.0518e6
TIERRA	5.976e+24	6.37814e6
MARTE	6.421e+23	3.3972e6
JUPITER	1.9e+27	7.1492e7
SATURNO	5.688e+26	6.0268e7
URANO	8.686e+25	2.5559e7
NEPTUNO	1.024e+26	2.4746e7

Por supuesto, se debe poder acceder al radio y masa de un planeta dado.

Ejercicio XI

Completar la clase CalendarPrint, para que imprima el mes actual en el siguiente formato:

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19*	20	21	22
23	24	25	26	27	28	29
30	31					

El asterisco indica el día actual.

Hint:

- `System.out.printf("%4s", "Sun");`
- `System.out.printf("Hola, %s. Tu edad es %d", name, age);`
- `System.out.printf("%8.2f", 3333.3333333333335);`

Ejercicio XII

Implementar una clase SetOperations con las siguientes operaciones de conjuntos:

- `public static <T> Set<T> union(Set<T> setA, Set<T> setB)`
- `public static <T> Set<T> interseccion(Set<T> setA, Set<T> setB)`
- `public static <T> Set<T> diferencia(Set<T> setA, Set<T> setB)`
- `public static <T> Set<T> difSimetrica(Set<T> setA, Set<T> setB)`

Ejercicio XIII

```
public class Persona {  
    private int legajo;  
    private int edad;  
    private String nombre;  
  
    //getters y setters  
}
```

Supongamos que tenemos una lista de Personas. Vamos a buscar frecuentemente en memoria por legajo en esta lista, con lo cual nos conviene más una estructura de diccionario.

Implementar una clase PersonasUtil con un método estático getPersonas, que reciba una lista de Personas y devuelva un Map con los legajos como claves y las personas como valores.

- **Crear un TestCase para probar el procedimiento.**

Hint: Click en el paquete donde vamos a querer el TestCase, click derecho, New /JUnit Test Case.

New JUnit Test Case

Select the name of the new JUnit test case. You have the options to specify the class under test and on the next page, to select methods to be tested.

☐ New JUnit 3 test ☒ New JUnit 4 test

Source folder:

Package:

Name:

Superclass:

Which method stubs would you like to create?

☐ setUpBeforeClass() ☐ tearDownAfterClass()
☐ setUp() ☐ tearDown()
☐ constructor

Do you want to add comments? (Configure templates and default value [here](#))

☐ Generate comments

Class under test:

Hint: Considerar además el método assertEquals.

Ejercicio XIV

Dentro de la clase Util del Ej XIII, implementar un método estático ordenarPersonas, que recibe una lista de Personas (la misma clase del Ej XIII) y las ordena. Asumimos que una persona es menor que otra si sus edades lo son.

Hint: No hay que implementar el algoritmo de ordenamiento.

- **Crear un TestCase para probar el procedimiento.**



Pero crearlo antes de la implementación!

RESPUESTAS:

1) Crucigrama

- 1. Synchronized**
- 2. Declaración**
- 3. Framework**
- 4. Vod**
- 5. Scope**
- 6. Long**
- 7. Java**
- 8. J2ee**
- 9. Compilador**
- 10. Ide**