1

UNIVERSIDAD DE EL SALVADOR
FACULTAD MULTIDISCIPLINARIA ORIENTAL
DEPARTAMENTO DE INGENIERIA Y ARQUITECTURA
INGENIERIA DE SISTEMAS INFORMATICOS

PROGRAMACIÓN III

GUIA DE LABORATORIO PRACTICO No. *1*
INTRODUCCIÓN A JAVA

Objetivo: Que el estudiante conozca la estructura básica de una aplicación sencilla de Java y pueda utilizarla para solucionar problemas sencillos en este lenguaje.

I. INTRODUCCION A LA PROGRAMACION CON JAVA

Aspectos a comprender antes de ver el primer programa Java.

1. Visibilidad de los nombres

En Java para producir un nombre no ambiguo para una biblioteca, el identificador utilizado no difiere mucho de un nombre de dominio en Internet. Los creadores de Java utilizaron los nombres de dominio de Internet a la inversa, dado que así es posible garantizar que éstos sean únicos. Una vez que se da vuelta al nombre de dominio, los nombres supuestamente representan subdirectorios.
Ejemplo: **com.bruceEckel.utilidad.manias**

2. Utilización de otros componentes

Mediante la palabra clave **import**, se le dice a Java que clase se quiere utilizar. Esta palabra, dice al compilador que traiga un paquete, que es una biblioteca de clases. La mayoría de las veces se utilizarán componentes de las bibliotecas de Java estándar que vienen con el propio compilador. Con ellas, no hay que preocuparse de los nombres de dominio largos y dados la vuelta; uno simplemente dice, por ejemplo:

import java.util.ArrayList;

para indicar al compilador que se desea utilizar la clase **ArrayList** de Java. Sin embargo, **util** contiene bastantes clases y uno podría querer utilizar varias de ellas sin tener que declararlas todas explícitamente. Esto se logra, usando el "*" como comodín:

*import java.util.**

3. La palabra clave static

Al decir que algo es **estático** se esta indicando que el dato o método no está atado a ninguna instancia de objeto de esa clase en particular. Por ello, incluso si nunca se creó un objeto de esa clase se puede invocar a un método **estático** o acceder a un fragmento de datos **estático**. Con los métodos y datos ordinarios, es necesario crear un objeto y utilizarlo para acceder al dato o método.

Para declarar un dato o un miembro a nivel de clase estático, basta con colocar la palabra clave static antes de la definición. Por ejemplo:

```
class PruebaEstatica {
        static int i = 47;
}
```

Ahora, aunque se construyan dos objetos de tipo **PruebaEstatica**, solo habrá un espacio de almacenamiento para **PruebaEstatica.i**. Ambos objetos compartirán la misma **i**.

Considere:

```
PruebaEstatica st1 = new PruebaEstatica();
PruebaEstatica st2 = new PruebaEstatica();
```

En este momento, tanto **st1.i** como **st2.i** tienen el valor 47, puesto que se refieren al mismo espacio de memoria.

4. Edición del programa fuente.

El programa fuente puede escribirse con cualquier editor de texto. Sin embargo, si esta utilizando un ambiente integrado de desarrollo como el caso del IDE Netbeans, estos tienen incorporado un editor de texto.

5. Nombre del archivo

El nombre del archivo para el sistema operativo debe ser el mismo nombre de la clase pública del mismo. En un archivo solamente puede haber una clase pública. También, no olvide que Java es “case sensitive” es decir, la mayúscula es diferente de la minúscula.

6. Compilación y ejecución

Para compilar y ejecutar un programa, es necesario disponer de un entorno de programación Java. El que nosotros vamos a utilizar es el JDK de Sun, el cual es gratuito.

Para trabajar en modo consola, la variable Path ya esta configurada en su computadora, si desea trabajar en su casa de la misma forma.

II. PRACTICA (consola)

II.1 Programación con salida en la ventana de comandos.

1. Edite, compile y corra el siguiente programa.

```
// HolaFecha.java
import java.util.*;

public class HolaFecha {
   public static void main(String[] args)
        { System.out.println ("Hola, hoy es:
        "); System.out.println (new Date());
   }
}
```

¿Tuvo errores de compilación? **R// No**
¿Cuáles? **R// Ninguno**
¿Cómo los solucionó? **R// No tuve que hacerlo**

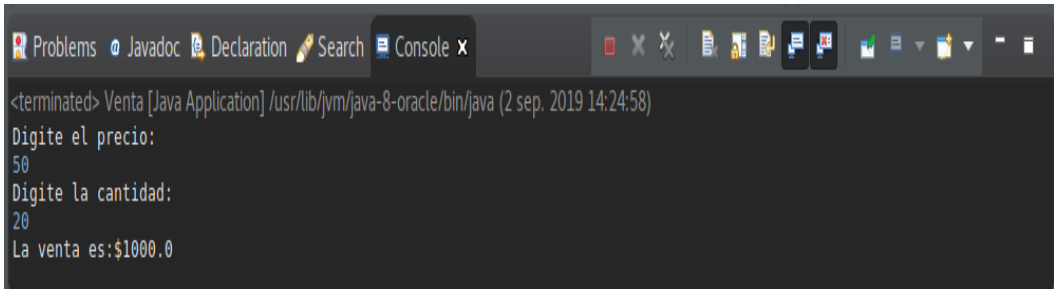
¿Cuál es la salida del programa?
R// Hola, hoy es:
Sun Sep 01 15:09:17 CST 2019

2. Edite, compile y corra el siguiente programa.

```
// Muestra el costo de la venta de un producto.
import java.util.*;
    import java.util.Scanner;

    public class Venta{
            public static void main(String[] args) {
            Double precio, costo;
            int    cantidad;
            Scanner stdin = new Scanner(System.in);

             System.out.println ("Digite el precio:");
```

```
            precio = stdin.nextDouble();
            System.out.println ("Digite la cantidad:");
            cantidad = stdin.nextInt();
            costo = precio * cantidad;
            System.out.println ("La venta es:$" + costo);

             }//main
      }//clase
```

R//Salida de la aplicación.



3. Edite, compile y corra el siguiente programa.

```
import java.util.*; import
java.util.Scanner;

public class Grados{
      public static void main(String[] args) {
         final Double PUNTO_CONGELACION = 32.0;
         final Double FACTOR_CONVERSION = 9.0/5.0;
         Double farenheit, celcius;
         Scanner stdin = new Scanner(System.in);

        System.out.println ("Digite los grados celcius:");
         celcius = stdin.nextDouble();
         farenheit = celcius*FACTOR_CONVERSION+PUNTO_CONGELACION;
         System.out.println ("Los grados farenheit son:" + farenheit);

      }
}
```

¿Qué elementos identifica en el programa?

- **Librerías**
- **Clase**
- **Metodos**

- **Método Estático5**
- **Variables**
- **Variables constantes**
- **Objeto Instanciado**

II.2 Programación mostrando la salida usando ventanas.

4. Edite, compile y corra el siguiente programa.

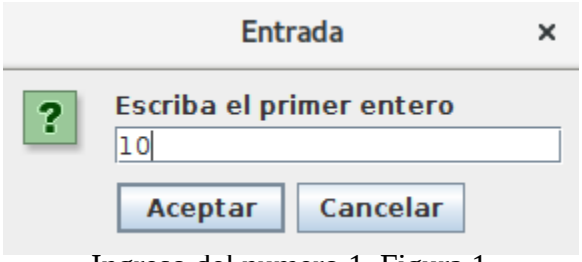
```
//programa que muestra la suma de dos números

//paquetes de JAVA
import javax.swing.JOptionPane;

public class Suma {
   public static void main ( String args[])
      {
         String primerNumero;    //primera cadena introducida por el
                                 // usuario
         String segundoNumero;   //segunda cadena introducida por el
                                   // usuario

           int numero1;          //primer numero a sumar
           int numero2;          //segundo numero a sumar
           int suma;             // suma de numero1 y numero2



           //leer el primer numero del usuario como una cadena
           primerNumero = JOptionPane.showInputDialog( "Escriba el
primer entero" );

           //leer el segundo numero del usuario como una cadena
           segundoNumero = JOptionPane.showInputDialog( "Escriba el
segundo entero");

           //convertir los numeros de tipo String a tipo int
           numero1 = Integer.parseInt( primerNumero );
           numero2 = Integer.parseInt ( segundoNumero );

           //sumar los numeros
           suma = numero1 + numero2;

           //mostrar el resultado
           JOptionPane.showMessageDialog ( null, "La suma es "+ suma,
"Resultados", JOptionPane.PLAIN_MESSAGE );

           System.exit (0);   // terminar aplicación con la ventana

     } // fin de main }
//fin de la clase suma
```

¿Cómo luce el programa en ejecución?

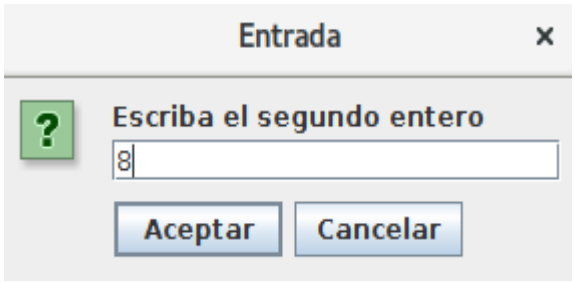
R// El programa pide el primer numero y el segundo de forma gráfica atraves de una pequeña ventana que contiene una caja de texto para ingresar los datos,

seguidamente de su ingreso nos muestra su suma atraves de otra pequeña ventana gráfica

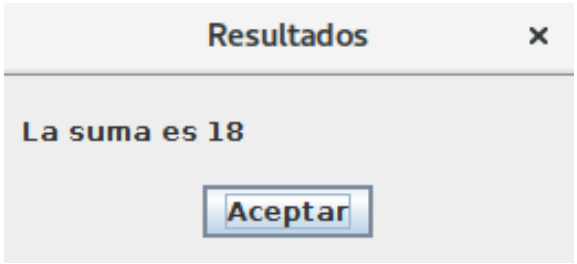
A continuación unas capturas de lo descrito anteriormente



Ingreso del numero 1 Figura 1.



Ingreso del numero 2 Figura 2.



Resultado de la suma del numero 1 y 2 Figura 3.

¿cuáles son las diferencias en cuanto al primer tipo de aplicaciones de Java (ejercicio 1)?
R// La utilización de una clase de la API Swing como JOptionPane que permite el uso de métodos estáticos para ingresar los datos y mostrarlos de forma gráfica, también esta la utilización de métodos estáticos para convertir los dos objetos de tipo String a int como los es parseInt y el uso del método exit de la clase System para finalizar la aplicación.

III. EJERCICIOS

5. Si un código fuente Java tiene varias clases, lo habitual es que sólo una de ellas cuente con método main(aunque en códigos posteriores podríamos ver que se usaran dos métodos main). Si hubiera varios definidos en clases del mismo código fuente, sólo se ejecuta uno de ellos; el resto no se ejecuta.

La pregunta es: ¿qué método main va a ejecutarse? **R// Se ejecutara el método main de la clase principal es decir la clase que tiene el mismo nombre que el del archivo**

La respuesta es: el asociado a la clase principal; esta clase principal la define el programador cuando ejecuta el código con: java <clase_principal>.

```
1  public class Primera{
2                  public static void main(String args[]){
3                  System.out.println("Esta es la clase Primera");
4          }}
5  class Segunda{
6          public static void main(String args[]){
7                  System.out.println("Esta es la clase Segunda");
8          }}
9  class Tercera{
10         public static void main(String args[]){
11                 System.out.println("Esta es la clase Tercera");
12         }}
13
```

El fichero donde se guarda el código Java debe llamarse "Primera.java" pues es la clase pública del código. El programador puede controlar qué clase va a ser la principal, es decir, qué clase va a contener el método main. El resto de métodos main no se tendrán en cuenta.

6. Escriba una aplicación que reciba del usuario el radio de un círculo como un entero, y que imprima el diámetro, circunferencia y área de ese círculo. Utilice el valor de 3.14159 para *PI*. También puede utilizar la constante predefinida **Math.PI** para el valor. Esta constante es mas precisa que 3.14159. La clase **Math** se define en el paquete **java.lang** por lo que no es necesario importarla.

R//

```
//HECHO POR CARLOR ALFONSO LEMUS RODEZNO LR18001


//pequeña aplicacion que calcula el diametro, circuferencia y area de un circulo


import javax.swing.JOptionPane;


public class Circulo {


        public static void main(String[] args) {


                String ingresoDeRadio;

                int radio;
```

```
                double diametro; //diametro
                double cfr; //circuferencia


                double area; //area
                //Se ingresan el radio atravez del metodo estatico showInputDialog
                //y se almacena en ingresoDeRadio
                ingresoDeRadio = JOptionPane.showInputDialog("Ingresa el radio ");

                //el metodo parseInt es un metodo estatico que convierte un String a un int
                radio = Integer.parseInt(ingresoDeRadio);

                diametro = (double)(radio *2); //se calcula el diametro

                cfr = Math.PI*diametro; //se calcula la circuferencia

                //el metodo pow es metodo que eleva un numero hasta una n potencia

                area = Math.PI * Math.pow(radio, 2); //se calcula el area

                //se muestra el resultado de las operaciones anteriores atravez
                //del metodo estatico showMessangeDialog
                JOptionPane.showMessageDialog(null,"Radio: " + radio + "\nDiametro: " +
                                diametro + "\nCircuferencia: " + cfr + "\nArea: " + area);
        }

}
```

7. Escriba un programa que lea el primer nombre y el apellido del usuario, como dos entradas separadas, y que concatene el primer nombre y el apellido, separándolos por un espacio. Muestre el nombre concatenado en un cuadro de dialogo de mensaje.

R//

```
//HECHO POR CARLOR ALFONSO LEMUS RODEZNO LR18001
//pequeña aplicacion que concatena el nombre con el apellido
import javax.swing.JOptionPane;
public class ConcatenarNombres {
        public static void main(String[] args) {
```

```
                String nombre;
                String apellido;
                String na;

                //Se ingresan el bombre atravez del metodo estatico showInputDialog
                //y se almacena en nombre
                nombre = JOptionPane.showInputDialog("Ingresa el nombre ");

                //Se ingresan el apellido atravez del metodo estatico showInputDialog
                //y se almacena en apellido
                apellido = JOptionPane.showInputDialog("Ingresa el radio ");

                //se concatena nombre y apellido y se almacena en na
                na = nombre + " " + apellido;

                //se muestra el nombre y apellido concatenado con el metodo
                //estatico showMessangeDialog
                JOptionPane.showMessageDialog(null, na);

        }

}
```

IV. EJERCICIOS DE INVESTIGACION

Conteste a las siguientes preguntas:

8. ¿Qué es un GUI?

R// La Interfaz Gráfica de Usuario o GUI (Graphical User Interface) es la parte visual de una aplicación y que permite al usuario interactuar al usuario con el software de forma gráfica atravez de iconos, botones, menús, gráficos etc.

9. ¿De la clase JOptionPane, qué hacen los métodos: showMessageDialog, y showInputDialog? **R//**

- **El método showMessageDialog permite ingresar un dato atravez de una caja de texto en una ventana y retorna un String.**

- **El metodo showInputDialog muestra un mensaje atravez de una ventana**

¿Cuáles son los argumentos que necesitan?.

- **El método showMessageDialog posee un único parámetro de tipo Object por lo tanto solo puede poseer un argumento de tipo String ya que esta clase hereda de Object**

- **El metodo showInputDialog tiene dos parametros el primero de tipo Component indica la ventana padre (o bien se le pasa null como argumento) y el segundo de tipo Object .**

10. JOptionPane tiene 5 constantes para los cuadros de dialogo de mensajes, que son: ERROR_MESSAGE, INFORMATION_MESSAGE, WARNING_MESSAGE, QUESTION_MESSAGE y PLAIN_MESSAGE. Investigue lo que hacen.

- **ERROR_MESSAGE: Pone un icono de error en el recuadro de dialogo del mensaje.**
- **INFORMATION_MESSAGE: Pone un icono de información en el recuadro de dialogo del mensaje.**
- **QUESTION_MESSAGE: Pone un icono de pregunta en el recuadro de dialogo del mensaje.**
- **PLAIN_MESSAGE: Deja sin icono al recuadro de dialogo del mensaje.**

11. ¿Qué hace el método exit de System?
R//Este metodo sale del programa actual al finalizar la maquina virtual de java sobre la cual se corren sus aplicaciones.

¿cuál es la diferencia al usar 0 o 1 como argumento de este método?
R//El parámetro que posee el método exit indica un estado, si el método se le pasa un cero como argumento entonces eso indicara que se finalizo el programa de forma exitosa pero si se le pasa cualquier numero diferente de cero esto indicara que hubo una finalización anormal o fallida es decir si este método se el pasara un 1 como argumento indicaría que la finalización del programa fue de forma anormal o fallida.

12. ¿Qué es Scanner ?
R//Scanner posee métodos que nos permiten ingresar datos de diferentes tipos, se pasa System.in como argumento debido a que representa el sistema estándar de entrada de información en Java y este se relaciona con el Objeto de tipo Scanner