Variables y constantes

Programación I

Objetivos del tema

- Utilizar sentencias de declaración de variables y constantes
- Comprender los tipos de datos primitivos
- Utilizar sentencias de asignación de variables
- Cargar datos de entrada y obtener salidas desde consola

Datos y variables

Un dato describe aquello con lo que opera el programa.

Para representar datos en un programa (datos de entrada, temporales, y de salida) se utilizan variables.

Para utilizar variables dentro de un programa es necesario:

- Declararlas mediantes sentencia de declaración
- Utilizar sentencias de asignación para asociar variables con datos

Nombre de variable: es lo que identifica a la variable o permite referenciarla.

Tipo de variable: define los valores posibles que puede tomar, y las operaciones que se van a poder realizar con ella.

Toda variable que se emplee en un programa Java debe definirse previamente a su utilización.

Nombre de variables o identificadores

Un nombre de variable o identificador comienza por una letra, un caracter de subrayado (_) o un caracter de peso($).

No hay límite máximo de caracteres para un nombre de variable.

En los identificadores de un programa en Java se distinguen las mayúsculas de las minúsculas.

- Por ejemplo, casa, CASA y Casa son tres variables diferentes.

El valor asociado a una variable puede cambiar varias veces durante la ejecución del programa.

Tipos de variables

- Un tipo es un "molde" que define los valores y operaciones posibles de las variables, y se dividen en dos grupos:
  - Tipos primitivos simples
  - Referencia (*se utilizarán más adelante*): permiten almacenar un conjunto de elementos, pueden ser definidos por el usuario, etc.
- En Java existen ocho tipos primitivos que se pueden clasificar en:
  - Números enteros (byte, short, int, long).
  - Números reales (float, double).
  - Carácter (char).
  - Booleano o lógico (boolean).
- Los tipos primitivos más utilizados en está materia son int, float, double, char y boolean.

Palabras reservadas

Existe una serie de palabras reservadas que no pueden emplearse como nombre de variables, y que tienen otros usos.

abstract	do	implements	protected	throw
boolean	double	import	public	throws
break	else	instanceof	rest	transient
byte	extends	int	return	true
case	false	interface	short	try
catch	final	long	static	void
char	finally	native	strictfp	volatile
class	float	new	super	while
const*	for	null	switch	
continue	goto*	package	synchronized	
default	if	private	this	

Declaración de variables

- Una variable corresponde a un dato cuyo valor puede modificarse durante la ejecución.
- Toda variable ha de declararse antes de ser usada en el código de un programa en Java.
- En la declaración de una variable debe indicarse el nombre de la variable y el tipo de la variable.
- Es posible declarar muchas variables de un mismo tipo en una misma sentencia de declaración (separadas por comas) o en múltiples sentencias de declaración.

Por ejemplo:

```
tipo1_de_variable nombre1, nombre2;
tipo2_de_variable nombre3;
tipo2_de_variable nombre4;
```

Ejemplo 1

```
/* Comentario: ejemplo de declaracion de variables
*/
public class Clase_2_Ejemplo_1{
  public static void main(String[] args) {
        //declaracion de la variable edad de tipo int
        //edad solo puede tomar valores enteros
        int edad;
        //declaracion de las variables altura y peso de tipo double
        //altura y peso solo pueden tomar valores reales
        double altura, peso;
        //declaracion de las variable existe de tipo boolean
        //existe solo puede tomar los valores boleanos (true y false)
        boolean existe;
  }
}
```

Sentencias de asignación de variables

- A una variable se le puede asignar un valor correspondiente con el tipo con el que fue declarada.
- La asignación se realiza usando =
- Una variable puede tomar el valor de otra variable del mismo tipo.

Por ejemplo:

```
tipo_de_variable nombre1, nombre2;
…
nombre1 = valor perteneciente a tipo_de_variable;
nombre1 = nombre2;
```

Ejemplo 2

```
/* Comentario: ejemplo de declaracion de variables y asignación de
* valores
*/
public class Clase_2_Ejemplo_2{
  public static void main(String[] args) {
        //antes de usar variables hay que declararlas
        int edad;
        double altura, peso;
        boolean existe;
        //las variables declaradas solo pueden tomar valores
        //segun los tipos declarados
        edad = 30;
        altura = 1.7;
        existe = false;
        peso = 75.5;
  }
}
```

Ejemplo 3

```
/* Comentario: ejemplo de declaracion de variables y asignación de
* valores
*/
public class Clase_2_Ejemplo_3{
  public static void main(String[] args) {
        int edad;
        int numero;
        float peso;
        //se puede asignar un valor a una variable en la declaracion
        boolean existe = true;
        numero = 30;
        //una variable puede tomar el valor de otra variable de igual tipo
        edad = numero;
        //una variable de tipo float necesita especificar su tipo
        //sobre el valor que toma (75.5 es un valor double por defecto)
        peso = (float) 75.5;
  }
}
```

Salida de datos por consola

- Para imprimir por consola el valor de una variable se puede utilizar la sentencia System.out.println():
- El texto entre doble comillas “” a colocar entre los paréntesis de System.out.println() puede concatenarse con otro texto utilizando +.

```
System.out.println("El valor " + "es: ");
```

- El texto con el que se va a concatenar puede estar entre doble comillas (otro texto) o directamente una variable de cualquier tipo. Para una variable que no es texto dentro System.out.println() se toma su valor y se lo convierte automáticamente en texto.

```
System.out.println("El valor es: " + entero);
```

Ejemplo 3

```
/* Comentario: ejemplo de declaracion de variables y asignación de valores
*/
public class Clase_2_Ejemplo_3{
  public static void main(String[] args) {
        int edad;
        double altura, peso;
        boolean existe;
        edad = 30;
        altura = 1.7;
        existe = false;
        peso = 75.5;
        //imprime dos textos concatenados
        System.out.println("El valor de " + "peso es: ");
        //imprime el valor de peso convertido en texto
        System.out.println(peso);
        //imprime un texto concatenado con el valor de peso convertido en texto
        System.out.println("El valor de peso es: "    + peso);
  }
}
```

Literales

- Un literal es una representación escrita de un valor pre-asignado en Java.
- Los literales boleanos son false y true.
- Los literales de tipo caracter aparecen entre comillas simples: caracteres letras mayúsculas (‘A’, ‘B’, ‘C’,...), caracteres letras minúsculas (‘a’, ‘b’, ‘c’,...), caracteres signos de puntuación (‘,’ ‘;’ ‘:’ ...), caracteres dígitos (‘0’, ’1’ ...), caracteres símbolos especiales (‘#’, ‘&’, ‘%’,...) y caracteres de control (‘\n’,’\t’,...).

Literal	Valor
\b	Retroceso o backspace
\t	Tabulación
\n	Nueva línea (enter)
\f	Salto de página

Ejemplo 4

```
/* Comentario: ejemplo de declaracion de variables y asignación de
* valores
*/
public class Clase_2_Ejemplo_4{
  public static void main(String[] args) {
        char caracter1;
        caracter1 = 'c';
        char caracter2;
        //una variable caracter2 toma el valor del caracter digito '1'
        caracter2 = '1';
        //una variable numero toma el valor del numero entero 1
        int numero = 1;
        System.out.println("El valor de caracter2 es: "    + caracter2);
        System.out.println("El valor de numero es: "    + numero);
  }
}
```

Constantes

- Constantes o variables finales: sirven para almacenar datos que no pueden modificarse posteriormente
  - Por ejemplo, el número PI, la aceleración de la gravedad G.
- Una vez inicializada una variable final su valor no puede ser modificado

Por ejemplo:

```
final double PI = 3.1415926;
final double g = 9.81;
```

Ejemplo 5

```
/* Comentario: ejemplo de variables y constantes
*/
public class Clase_2_Ejemplo_5{
  public static void main(String[] args) {
        final double g = 9.81;//constante
        double altura;
        int edad = 20;
        boolean existe = true;
        g = 6.3;//GENERA UN ERROR AL INTENTAR CAMBIAR SU VALOR
  }
}
```

Ejemplo 6

```
/* Comentario: uso de literales en la salida por consola
*/
public class Clase_2_Ejemplo_6{
  public static void main(String[] args) {
        System.out.println ("Hola Mundo. Estoy programando.");
        System.out.println ("Hola Mundo. \nEstoy programando.");
        System.out.println ("Hola Mundo. \n\tEstoy programando.");
        System.out.println ("Hola Mundo. \n\t\tEstoy programando.");
  }
}
```

Práctico primera parte

1. Escribir un programa que para los tipos int, double, char y boolean declare una variable en cada caso, luego asigne un valor a cada una correspondiente al tipo que la variable tiene, e imprima por pantalla cada una de las variables.
2. Escribir un programa que declare tres variables de tipo double y una constante de tipo double con valor 1.0 . Luego deberá asignar el valor de la constante a una de la variables declaradas, y posteriormente sobre las dos restantes variables se le deberá asignar el valor de la variable que inicialmente fue seteada con el valor de la constante. Finalmente imprima por pantalla cada una de las variables.

Entrada y salida de datos desde consola

- Importación de librerías de un programa: una librería ofrece un conjunto de utilidades que se acceden mediante sentencias para hacer operaciones predefinidas.
- Para acceder a una librería se utiliza la sentencia import seguido del nombre de la librería.

```
/* importación de dos librerías de operaciones de
*  entrada/salida
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;

public class Clase_2_Ejemplo_7{
  public static void main(String[] args){
 }
}
```

Entrada y salida de datos desde consola

- Para leer un valor desde la consola y asignarlo a una variable primero hay utilizar una sentencia que permita capturar un error o excepción en el caso que el valor ingresado no se corresponda con el tipo de la variable. Por ejemplo asignar un valor real a una variable de tipo entero. La sentencia a utilizar es try{…}catch{}

```
try {
//try define un bloque {…} para capturar posibles errores o excepciones
   Sentencia_1;
   Sentencia_2;
   //…
   }
catch (Exception exc) {
//catch define un bloque {…} que se ejecuta solo cuando ocurre un
//error en el bloque del try
//exc es una variable de tipo Exception (tipo predefinido de java)
//cuyo valor es el tipo de error
   Sentencia_3;
   //…
}
```

Ejemplo 7

```
/* importación de dos librerías de operaciones de
*  entrada/salida
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;

public class Clase_2_Ejemplo_7{
  public static void main(String[] args){
        try {
        }
        catch (Exception exc ) {
        }
  }
}
```

Entrada y salida de datos desde consola

- El paso siguiente es utilizar una sentencia o varias para solicitar desde teclado el ingreso de una dato.

```
BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
```

- Cuando se ejecuta la sentencia múltiple anterior (sentencias con funcionalidades pertenecientes a las librerías utilizadas) se prepara un buffer (entrada) donde se van a cargar los datos de la consola.

Ejemplo 7

```
/* importación de dos librerías de operaciones de
*  entrada/salida
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;

public class Clase_2_Ejemplo_7{
   public static void main(String[] args){
          try {
                BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
          }
          catch (Exception exc ) {
          }
   }
}
```

Entrada y salida de datos desde consola

- Antes de solicitar el ingreso de un valor es importante imprimir un mensaje por consola sobre el tipo de valor que el usuario debe ingresar.

```
System.out.println ("Ingrese un entero: ");
```

- Posteriormente se programa una sentencia que va permitir el ingreso del dato.

```
entero = Integer.valueOf(entrada.readLine());
```

- Cuando se ejecuta la sentencia anterior en la consola aparece un cursor en espera de que el usuario ingrese un dato.
- Una vez que lo tipea y aprieta enter el valor ingresado (mediante la sentencia entrada.readLine()) se convierte a un valor entero (mediante la sentencia Integer.valueOf()).

Ejemplo 7

```
/* importación de dos librerías de operaciones de
*  entrada/salida y carga de una variable de tipo entero desde consola
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;

public class Clase_2_Ejemplo_7{
   public static void main(String[] args){
           //declaro la variable donde quiero cargar el valor ingresado
           int entero;
           try {
                       BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
                       //imprimo por consola un mensaje que indique al usuario el valor a ingresar
                       System.out.println ("Ingrese un entero: ");
                       //cuando se ejecuta Integer.valueOf(entrada.readLine()) sobre la pantalla
                       //de la consola aparece un cursor en espera de que el usuario ingrese un dato.
                       //Una vez que lo tipea y aprieta enter el valor ingresado se convierte a un
                       //valor entero
                       entero = Integer.valueOf(entrada.readLine());
           }
           catch (Exception exc ) {
           }
   }
}
```

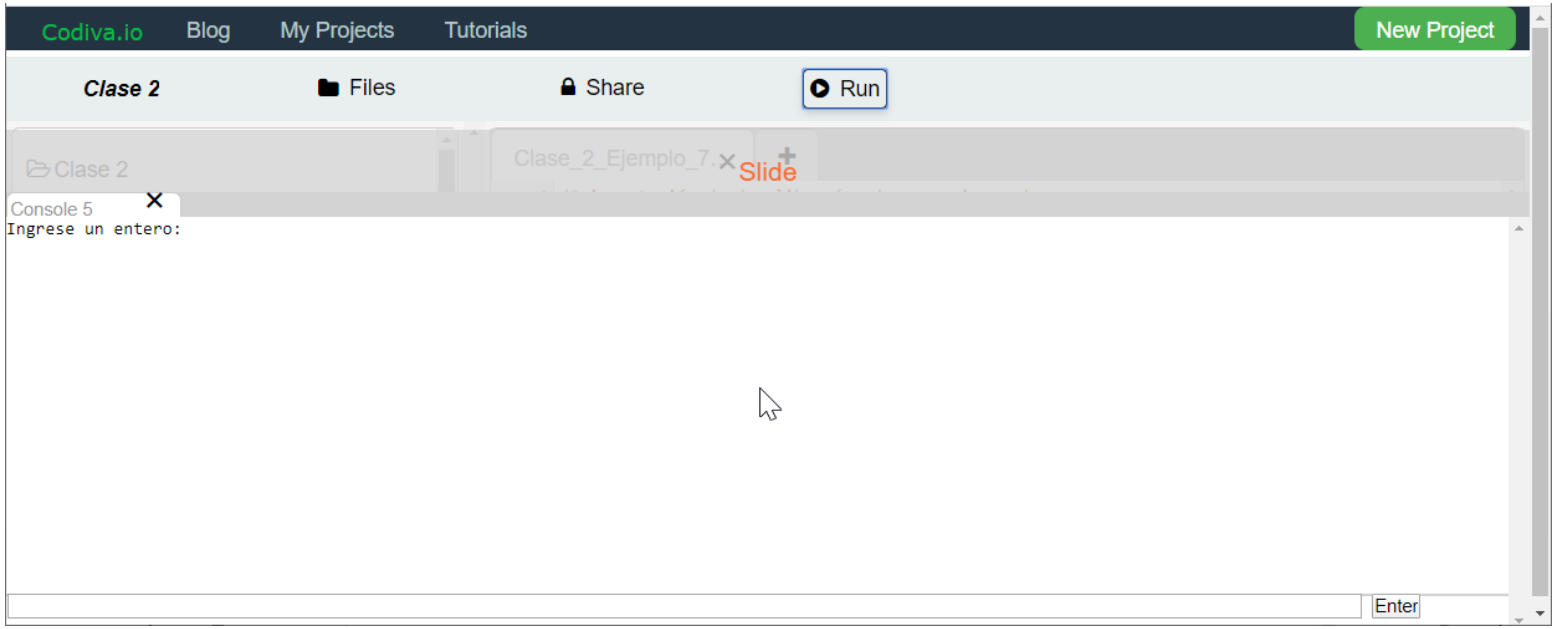
Ejemplo 7

```
/* importación de dos librerías de operaciones de
*  entrada/salida y carga de una variable de tipo entero desde consola
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;

public class Clase_2_Ejemplo_7{
   public static void main(String[] args){
           //declaro la variable donde quiero cargar el valor ingresado
           int entero;
           try {
                       BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
                       System.out.println ("Ingrese un entero: ");
                       entero = Integer.valueOf(entrada.readLine());
                       //imprimo por consola un mensaje con el valor de la variable entero
                       System.out.println("El valor ingresado es: "    + entero);
           }
           catch (Exception exc) {
                       //imprimo por consola un mensaje con el valor de la variable exc (tipo de error)
                        System.out.println(exc);
           }
   }
}
```

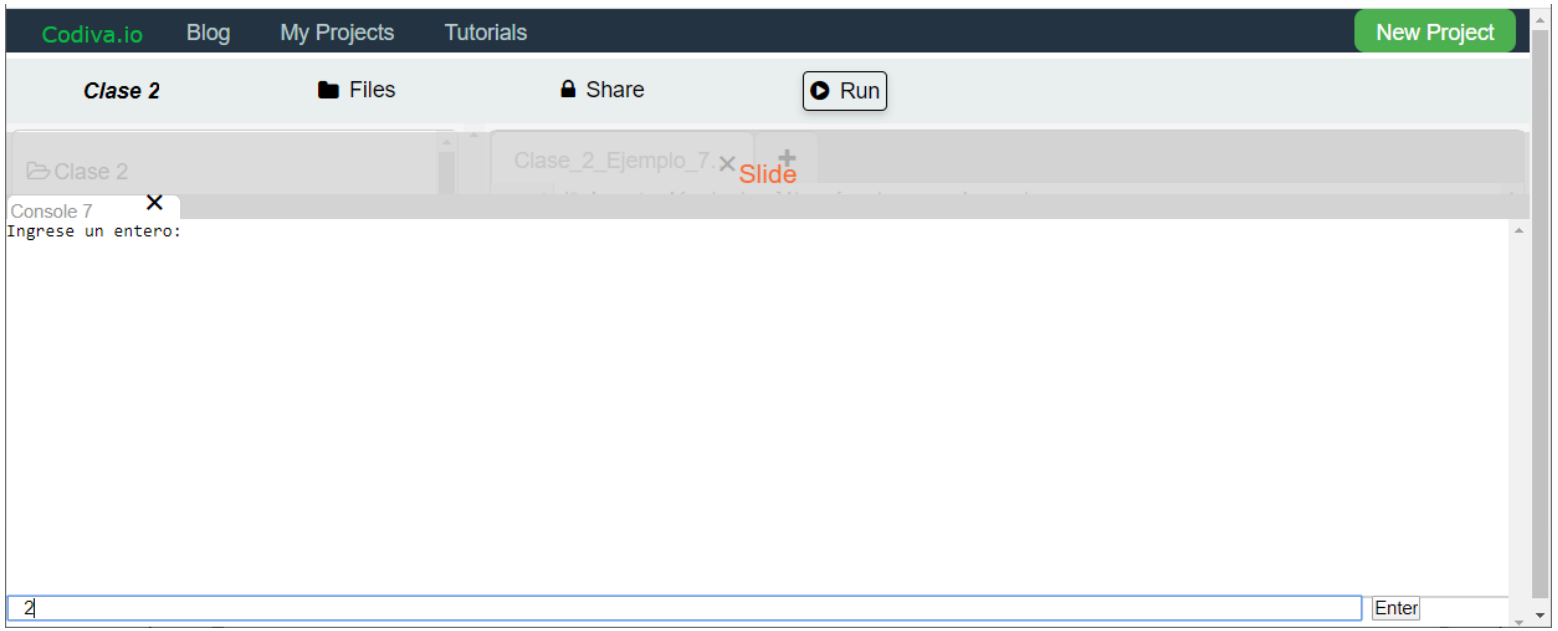
Programando el ejemplo 7 en el entorno Codiva

- Cuando ejecuto Clase_2_Ejemplo_7.java aparece el mensaje Ingrese un entero:, y el programa queda a la espera de que se cargue un valor.



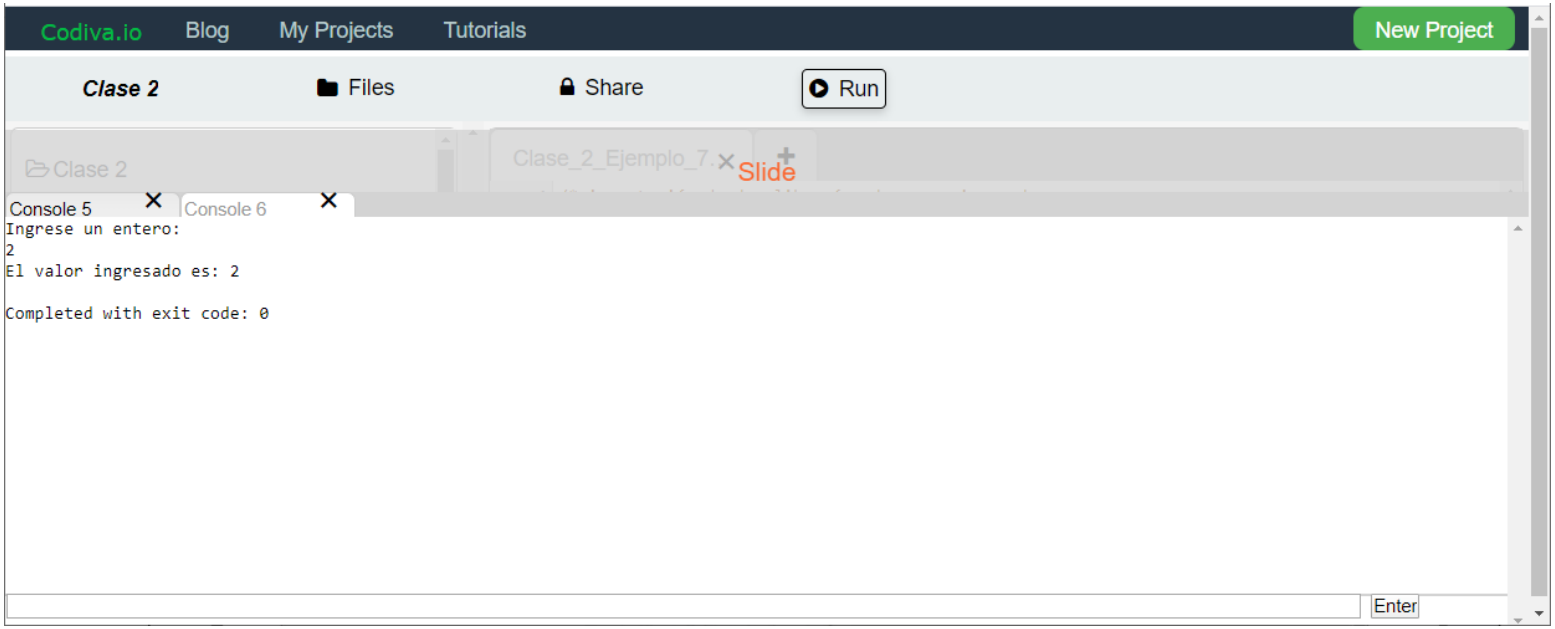
Programando el ejemplo 7 en el entorno Codiva

- Para cargar un valor en este entorno se ingresa un valor en el cuadro de texto de la parte inferior y se oprime enter (parte inferior lado derecho). En este ejemplo se ingresa un 2.



Programando el ejemplo 7 en el entorno Codiva

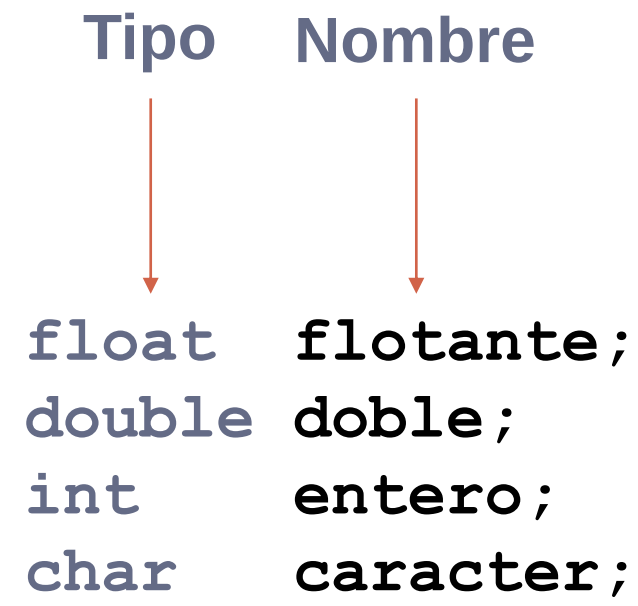
- Luego de presionar enter se puede ver en la consola el valor ingresado y el mensaje de impresión de la salida.



Programando el ejemplo 7 en el entorno Codiva

- En el caso de que el usuario ingrese un valor que no es entero (la variable a la que se le quiere asignar el valor es de tipo entero) se genera un error o excepción. En ese punto se ejecuta el bloque dentro del catch (en el ejemplo hay una sentencia que imprime el error por consola).

Cargar y visualizar más variables y de distintos tipos primitivos



Ejemplo 8

```
/* Carga de variables de tipos primitivos desde consola
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_2_Ejemplo_8 {
 public static void main(String[] args){
   float  flotante;
   double doble;
   int    entero;
   char   caracter;
   try {
        //Puedo utilizar el mismo buffer entrada mas de una vez o por cada carga desde consola
        BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
        System.out.println ("Ingrese un float: ");
        flotante = Float.valueOf(entrada.readLine());//la sentencia Float.valueOf() convierte a float
        System.out.println ("Ingrese un double: ");
        doble = Double.valueOf(entrada.readLine());//la sentencia Double.valueOf() convierte a double
        System.out.println ("Ingrese un entero: ");
        entero = Integer.valueOf(entrada.readLine());//la sentencia Integer.valueOf() convierte a int
        System.out.println ("Ingrese char: ");
        //Para convertir/acceder al caracter la sentencia es diferente
        caracter = entrada.readLine().charAt(0);
        System.out.println("El float es: " + flotante);
        System.out.println("El double es: " + doble);
        System.out.println("El entero es: " + entero);
        System.out.println("El char es: " + caracter);
   }
   catch (Exception exc) {
        System.out.println(exc);
   }
 }
}
```

Tipo de datos para representar un texto corto

- El tipo String permite representar un texto corto o cadena de caracteres. La declaración de una variable de tipo es:

```
//String es un tipo especial para almacenar texto
//si solo se necesita un caracter deberia utilizar char
String texto;
```

- El valor es una cadena de caracteres entre doble comillas.

```
texto = "una frase corta";
```

- La lectura e impresión por consola se realiza de la siguiente forma:

```
//para leer textos desde consola no necesita convertirlo a
//String
texto = entrada.readLine();
…
System.out.println(texto);
System.out.println("El texto es " + texto)
```

Ejemplo 9

```
/* Carga de variable de tipo string desde consola
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_2_Ejemplo_9 {
 public static void main(String[] args){
   String  texto;
   try {
       BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
       System.out.println ("Ingrese un texto: ");
       texto = entrada.readLine();
       System.out.println("El texto es: " + texto);
   }
   catch (Exception exc) {
       System.out.println(exc);
   }
 }
}
```

Práctico segunda parte

1. Escribir un programa que solicite y luego muestre por consola los valores necesarios para dibujar un círculo y un triángulo. Hay que determinar en cada caso que constantes (que no se cargan por consola) y variables con tipos son necesarias declarar.

2. Escribir un programa que solicite los siguientes datos de una persona (nombre, apellido, edad, altura, ocupación, dirección) y los imprima por pantalla.

3. Escribir un programa que pida que se ingresen datos necesarios para emitir una factura por la compra de dos artículos de librería (tipo factura, número, nombre cliente, producto 1, importe 1, producto 2, importe 2, importe total). Como salida debe imprimir por pantalla la factura en un formato similar al siguiente (utilizar literales):

Factura	**C**	**201**
Nombre y Apellido	**Jorge Rodríguez**	
Producto		**Importe**
Lápices		**12.2**
Cuadernos		**20.0**
Importe Total		**30.2**