```
/*Sin tratar excepciones
   Al introducir un dato erróneo el programa se cuelga y no finaliza correctamente, las
sentencias después del error no se ejecutan*/
    public static void main(String[] args) {
      int a,b,c=0;
      Scanner tec=new Scanner(System.in);
      a=tec.nextInt();
      b=tec.nextInt();
      System.out.println(a/b);

      System.out.println("FIN");
   }
```

```
run:
p
Exception in thread "main" java.util.InputMismatchException
        at java.util.Scanner.throwFor(Scanner.java:864)
        at java.util.Scanner.next(Scanner.java:1485)
        at java.util.Scanner.nextInt(Scanner.java:2117)
        at java.util.Scanner.nextInt(Scanner.java:2076)
        at Ejercicios1.Ejemplo1.main(Ejemplo1.java:23)
C:\Users\maria\AppData\Local\NetBeans\Cache\8.1\executor-snippets\run.xml:53: Java returned: 1
BUILD FAILED (total time: 3 seconds)
```

```
/* Tratando las excepciones y mostrando el mensaje genérico*/
   public static void main(String[] args) {
      int a,b,c=0;
      Scanner tec=new Scanner(System.in);
      try{
      a=tec.nextInt();
      b=tec.nextInt();
      System.out.println(a/b);
      }
      catch (Exception d){
         System.out.println(d);
         System.out.println(d.getMessage());
        d.printStackTrace();
      }

      System.out.println("FIN");
   }
```

```
run:
p
java.util.InputMismatchException
java.util.InputMismatchException
null
FIN
        at java.util.Scanner.throwFor(Scanner.java:864)
        at java.util.Scanner.next(Scanner.java:1485)
        at java.util.Scanner.nextInt(Scanner.java:2117)
        at java.util.Scanner.nextInt(Scanner.java:2076)
        at Ejercicios1.Ejemplo1.main(Ejemplo1.java:34)
D SUCCESSFUL (total time: 6 seconds)
```

System.out.println(d);

System.out.println(d.getMessage());

d.printStackTrace();

Las sentencias después del tratamiento del error sí se ejecutan

```
/* Tratando las excepciones y mostrando los mensajes particulares */
public static void main(String[] args) {
   int a,b,c=0;
   Scanner tec=new Scanner(System.in);
   try{
   a=tec.nextInt();
   b=tec.nextInt();
   System.out.println(a/b);
   }
   catch (InputMismatchException d){
      System.out.println("Tipo de entrada incorrecta");
     tec.next();
   }catch (ArithmeticException d){
      System.out.println("División por 0");

   }catch (Exception d){
      System.out.println("Error desconocido");

   }
   System.out.println("FIN");
}


 /* Pide los datos de entrada hasta que la ejecución sea correcta*/
public static void main(String[] args) {
   int a,b,c=0;
   boolean correcto=false;
   Scanner tec=new Scanner(System.in);
  do{
     try{
   a=tec.nextInt();
   b=tec.nextInt();
   System.out.println(a/b);
   correcto=true;
   }
    catch (InputMismatchException d){
      System.out.println("Tipo de entrada incorrecta");
     tec.next();

   }catch (ArithmeticException d){
      System.out.println("División por 0");

   }catch (Exception d){
      System.out.println("Error desconocido");

   }
  }while(!correcto);
   System.out.println("FIN");
}
```

Pero si no, continúa pidiéndonos los 2 datos
Mejor separamos el programa principal de la entrada de datos y sus posibles errores:

```
   public static void main(String[] args) {
      int a,b,c=0;

      a=leerEntero();
      b=leerEntero();
       try{
          System.out.println(a/b);


      }catch (ArithmeticException d){
         System.out.println("División por 0");

      }catch (Exception d){
         System.out.println("Error desconocido");

      }

      System.out.println("FIN");
   }


public static int leerEntero(){
    Scanner tec=new Scanner(System.in);
    boolean correcto=false;
    int a=0;
     do{
        try{
       a=tec.nextInt();
       correcto=true;
       }catch (InputMismatchException d){
          System.out.println("Teclee un entero");
         tec.next();
       }
      }while(!correcto);
      return a;
}
```

Ejercicio 2:

```
public class Persona {
String nombre;
int edad;
public Persona(String nom){
   nombre=nom;
}
public void ponerEdad(int e) throws Exception{
   if(e>0 && e<100)
      edad=e;
   else
      throw new Exception("Edad incorrecta");
}
public String toString(){
```

```
   return nombre+":"+edad+"\n";
}
}

public class PruebaPersona {
   public static void main(String[] args) {
      Scanner tec=new Scanner(System.in);
      String nom;
      int edad;
      Persona p;
      System.out.println("Nombre:");
      nom=tec.nextLine();
      p=new Persona(nom);
      System.out.println("Edad:");
      edad=tec.nextInt();
      try{
         p.ponerEdad(edad);
      }catch(Exception e){
         System.out.println(e.getMessage());
      }
      System.out.println(p);
   }
}
```

El método de la clase persona comprueba los datos y si no son correctos lanza la excepción, como la va a lanzar y no la trata, tiene que aparecer en la cabecera **throws Exception**
En el método donde se llama al método ponerEdad, nos obliga a tratar la excepción o a propagarla