



INSTITUTO POLITÉCNICO NACIONAL

Escuela Superior de Cómputo



Unidad de Aprendizaje:

Programación Orientada a Objetos

Grupo:

2CV8

Profesor:

Daniel Cruz García

Práctica 3

“Sobrecarga de constructores y métodos”

Alumnos:

Luciano Espina Melisa

Dávila García Rivas Emiliano

Ramos Mesas Edgar Alaín

Fecha de entrega:

19/03/2018

Introducción

Cuando nosotros realizamos una aplicación donde abstraemos objetos para poder cumplir la funcionalidad de esta, se hacen diversas acciones, tales como saber las propiedades del objeto, lo que hace, lo que se puede hacer con él, etc.

Se ha aplicado los constructores y métodos para hacer aplicaciones, con el objetivo de poder hacer métodos con el mismo nombre pero que tengan diferentes argumentos, en esta práctica se pondrá a prueba las llamados sobrecargas de métodos y constructores.

Marco teórico

Un método sobrecargado se utiliza para reutilizar el nombre de un método, pero con diferentes argumentos. [1]

No puede haber dos métodos que se llamen igual con la misma lista de argumentos, aunque devuelvan datos de distinto tipo. El compilador sabría a cuál de todas las sobrecargas nos referimos por los argumentos que se le pasen en la llamada, pero no sería capaz de determinar cuál de ellas debe ejecutar si tienen la misma lista de argumentos. [2]

Lo que diferencia las listas de argumentos de las diferentes sobrecargas no es el nombre de las variables, sino el tipo de cada una de ellas. [2]

Las reglas para sobrecargar un método:

- ☒ Los métodos sobrecargados deben de cambiar la lista de argumentos
- ☒ Pueden cambiar el tipo de retorno
- ☒ Un método puede ser sobrecargado en la misma clase o en una subclase. [1]

Constructores

Un constructor es una función, método, etc de las clases, la cual es llamada automáticamente cuando se crea un objeto en esta clase.

Por ser métodos, los constructores también aceptan parámetros. Cuando en una clase no especificamos ningún tipo de constructor, el compilador añade uno público por omisión sin parámetros, el cual NO hace nada. [3]

Características:

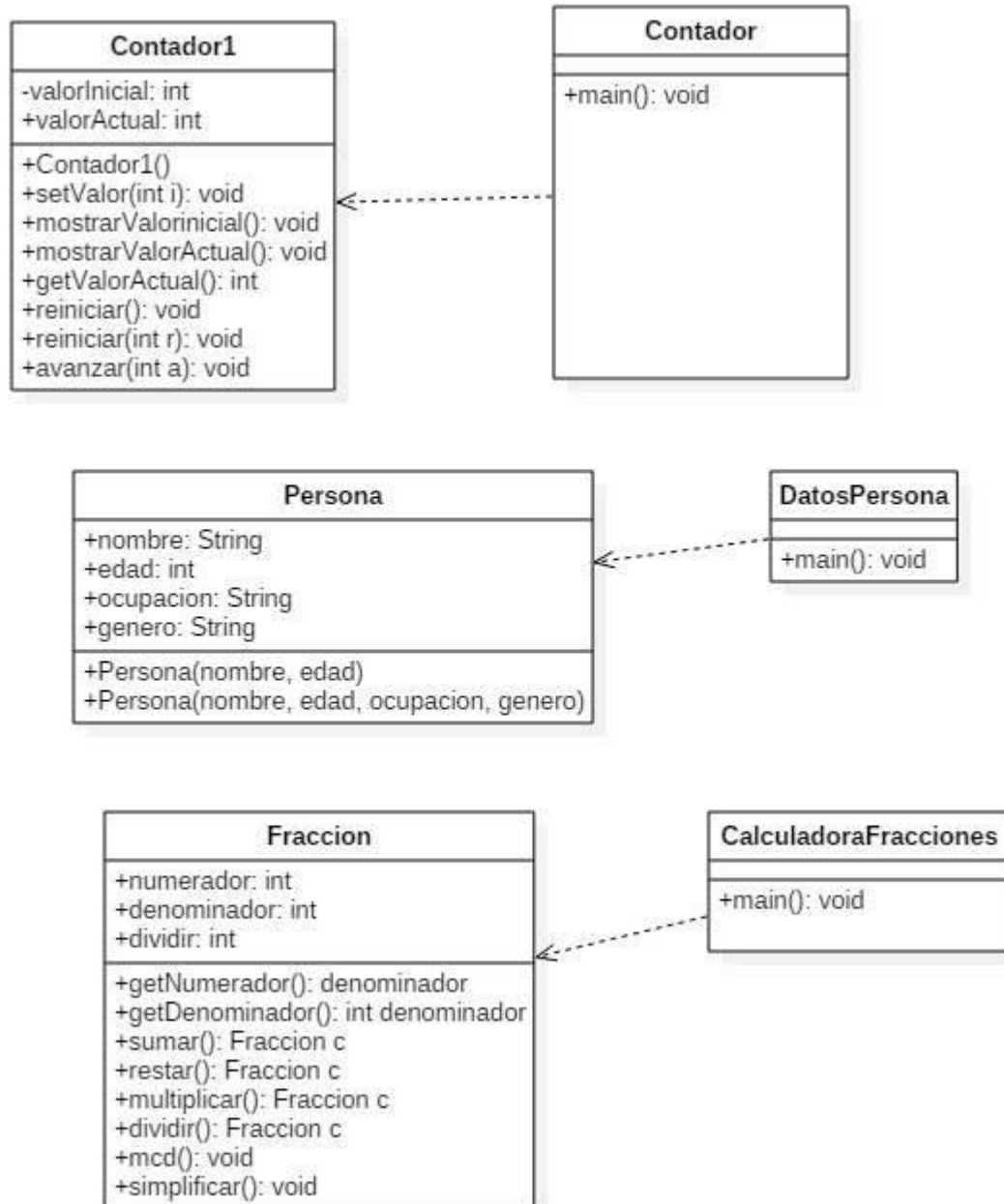
- ☒ Un constructor, tiene el mismo nombre de la clase a la cual pertenece.
- ☒ No puede ser heredado
- ☒ No retorna ningún valor (ni Void), por lo cual no debe especificarse ningún tipo de dato.
- ☒ Debe declararse como "public", en casos extraordinarios será de otro tipo. [3]

Los constructores no se heredan, aunque las subclases heredan todos los métodos y variables de superclases, no heredan sus constructores.

Las clases solo pueden tener dos formas: debe escribirse o, si éste no lo escribe, debe usar el constructor predeterminado.

- ☒ Si se usa, es preciso que "super" o "this" sean la primera línea del constructor.
- ☒ Si un constructor no contiene ninguna llamada a super (...) ni a this (...) el compilador introduce una llamada al constructor de la superclase sin argumentos.

Análisis del problema

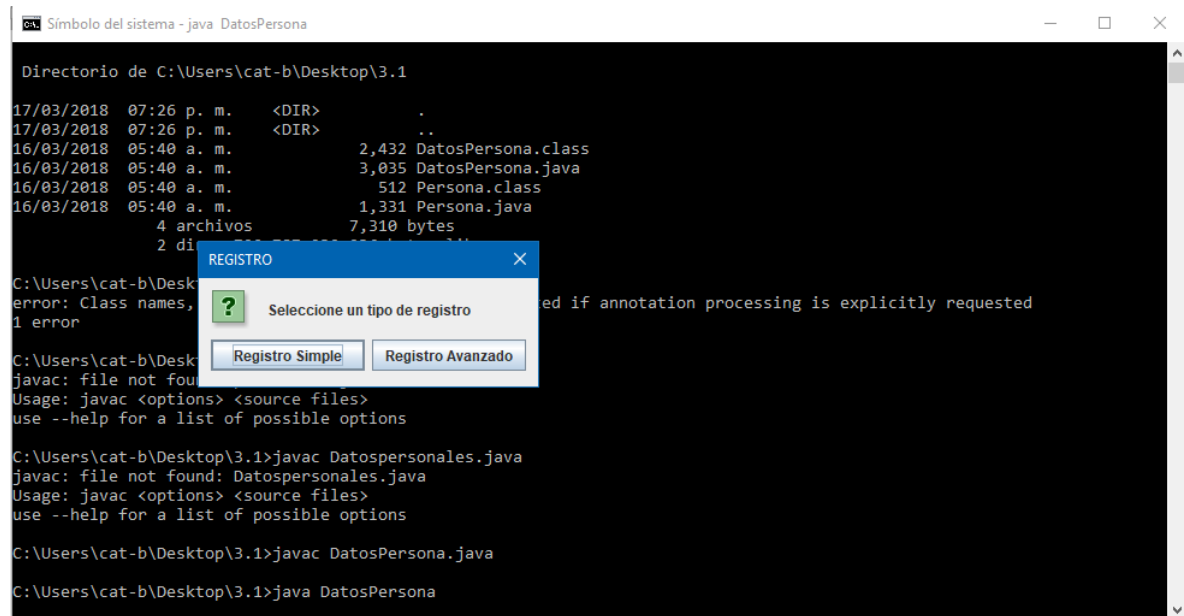


Pruebas y resultados

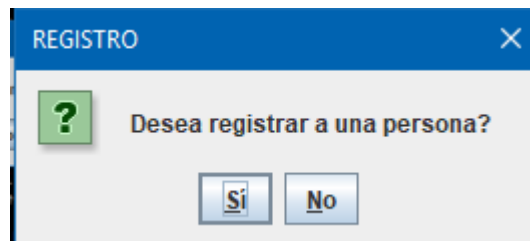
Primer programa

Aplicación que permite capturar y mostrar los datos de una persona.

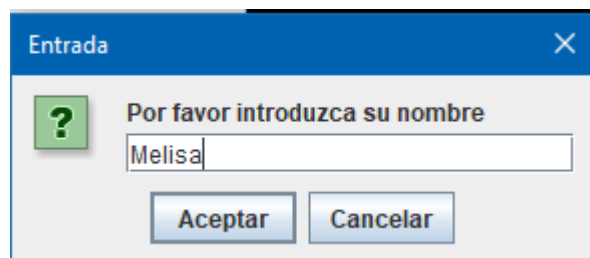
Se muestra la captura de pantalla de la ejecución del programa.



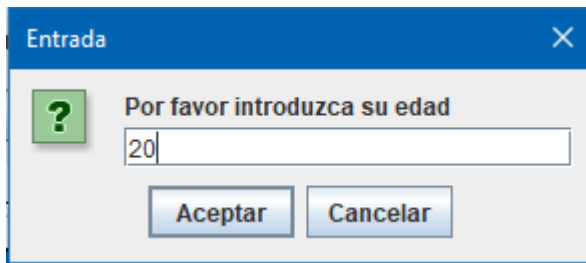
Seleccionando "registro simple"



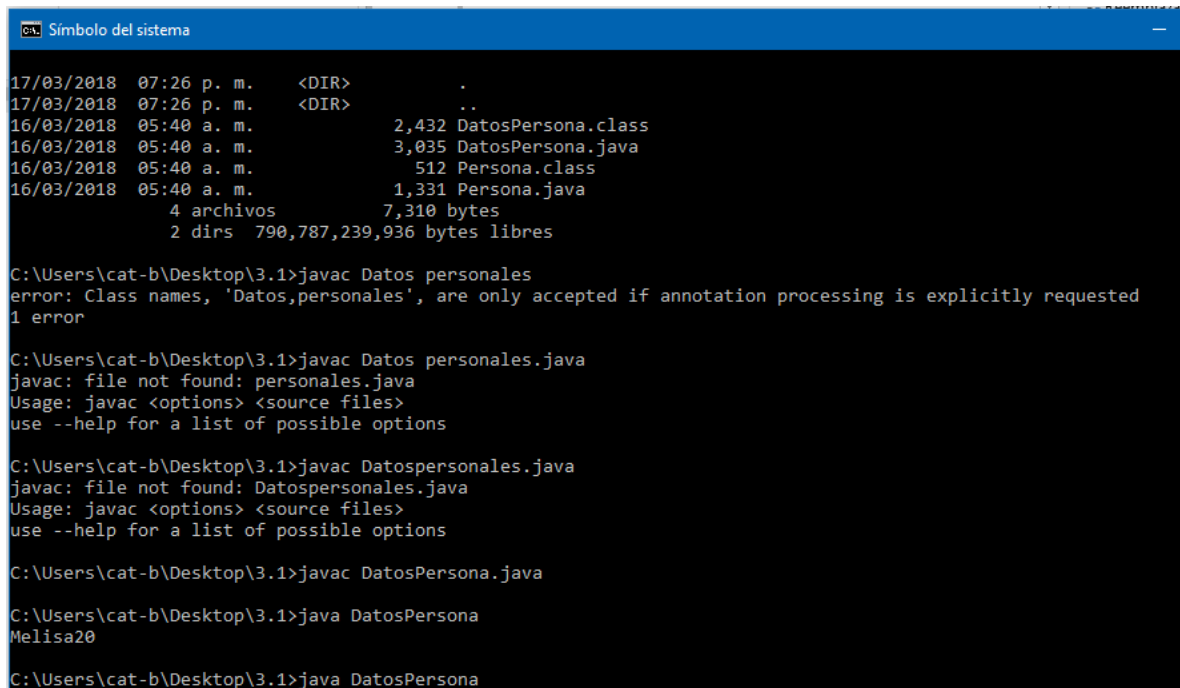
Seleccionando "sí" e ingresando el nombre



Seleccionando "Aceptar" e introduciendo la edad



Se guardan los datos capturados



```
C:\> Símbolo del sistema

17/03/2018 07:26 p. m. <DIR> .
17/03/2018 07:26 p. m. <DIR> ..
16/03/2018 05:40 a. m. 2,432 DatosPersona.class
16/03/2018 05:40 a. m. 3,035 DatosPersona.java
16/03/2018 05:40 a. m. 512 Persona.class
16/03/2018 05:40 a. m. 1,331 Persona.java
4 archivos 7,310 bytes
2 dirs 790,787,239,936 bytes libres

C:\Users\cat-b\Desktop\3.1>javac Datos personales
error: Class names, 'Datos,personales', are only accepted if annotation processing is explicitly requested
1 error

C:\Users\cat-b\Desktop\3.1>javac Datos personales.java
javac: file not found: personales.java
Usage: javac <options> <source files>
use --help for a list of possible options

C:\Users\cat-b\Desktop\3.1>javac Datospersonales.java
javac: file not found: Datospersonales.java
Usage: javac <options> <source files>
use --help for a list of possible options

C:\Users\cat-b\Desktop\3.1>javac DatosPersona.java

C:\Users\cat-b\Desktop\3.1>java DatosPersona
Melisa20

C:\Users\cat-b\Desktop\3.1>java DatosPersona
```

Ahora ejecutando de nuevo, pero seleccionando registro avanzado

```
Símbolo del sistema - java DatosPersona
16/03/2018 05:40 a. m.      2,432 DatosPersona.class
16/03/2018 05:40 a. m.      3,035 DatosPersona.java
16/03/2018 05:40 a. m.      512 Persona.class
16/03/2018 05:40 a. m.      1,331 Persona.java
4 archivos      7,310 bytes
2 dirs 790,787,239,936 bytes libres

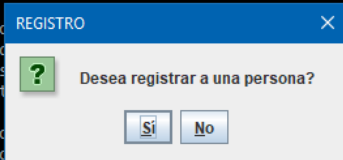
C:\Users\cat-b\Desktop\3.1>javac Datos personales
error: Class names, 'Datos,personales', are only accepted if annotation processing is explicitly requested
1 error

C:\Users\cat-b\Desktop\3.1>javac DatosPersona.java
javac: file not found
Usage: javac <options> <source files>
use --help for a list of possible options

C:\Users\cat-b\Desktop\3.1>javac DatosPersona.java
javac: file not found
Usage: javac <options> <source files>
use --help for a list of possible options

C:\Users\cat-b\Desktop\3.1>java DatosPersona
Melisa20

C:\Users\cat-b\Desktop\3.1>javac DatosPersona.java
C:\Users\cat-b\Desktop\3.1>java DatosPersona
```



Opción “sí” introduciendo nombre y edad

Entrada

Por favor introduzca su nombre

Edgar

Aceptar Cancelar

Entrada

Por favor introduzca su edad

20

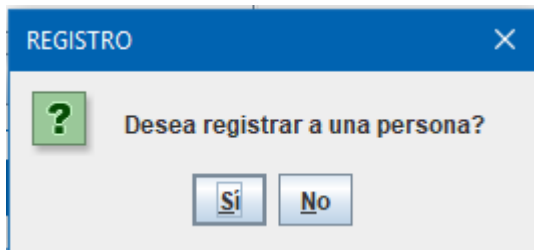
Aceptar Cancelar

Entrada

Por favor Introduzca su ocupacion

Estudiante

Aceptar Cancelar



Aparece en la consola

```
C:\Users\cat-b\Desktop\3.1>java DatosPersona
Exception in thread "main" java.lang.NumberFormatException: null
    at java.base/java.lang.Integer.parseInt(Unknown Source)
    at java.base/java.lang.Integer.parseInt(Unknown Source)
    at DatosPersona.main(DatosPersona.java:75)

C:\Users\cat-b\Desktop\3.1>java DatosPersona
Edgar20EstudianteHombre

C:\Users\cat-b\Desktop\3.1>
C:\Users\cat-b\Desktop\3.1>
```

Segundo programa

Hacer un contador con un valor inicial o que lo de el usuario

Se muestra la ejecución del programa


```
Símbolo del sistema - java Contador

at java.base/java.lang.Integer.parseInt(Unknown Source)
at DatosPersona.main(DatosPersona.java:75)

C:\Users\cat-b\Desktop\3.1>java DatosPersona
Edgar20EstudianteHombre

C:\Users\cat-b\Desktop\3.1>
C:\Users\cat-b\Desktop\3.1>cd..

C:\Users\cat-b\Desktop>cd 3.2

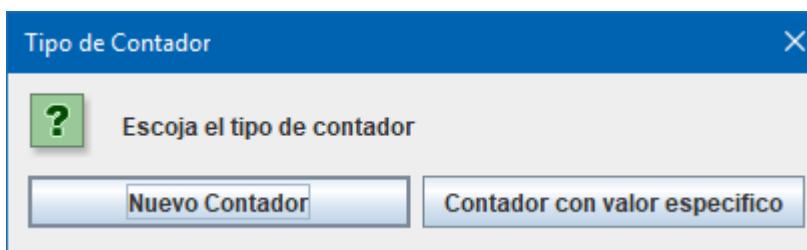
C:\Users\cat-b\Desktop\3.2>dir
El volumen de la unidad C: tiene el sistema de archivos NTFS.
El número de serie de la unidad es {F8B73831-6F9F-436D-8C9A-F64BF645F618}.

Directorio de C:\Users\cat-b\Desktop\3.2:

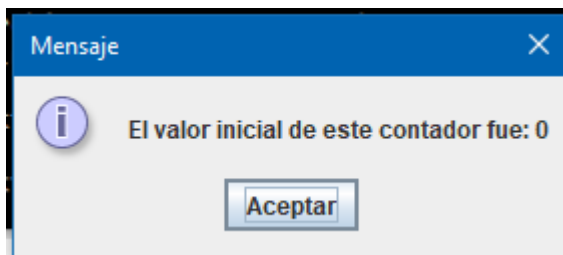
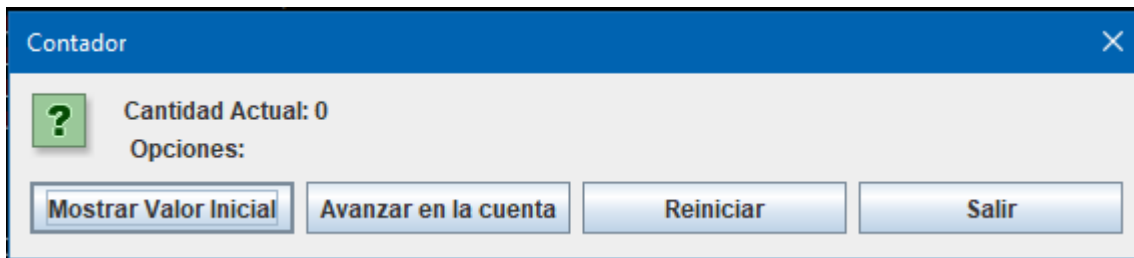
17/03/2018  07:26 p. m.      <DIR>          ..
17/03/2018  07:26 p. m.      <DIR>          .
16/03/2018  05:40 a. m.      2,384 Contador.class
16/03/2018  05:40 a. m.      4,676 Contador.java
16/03/2018  05:40 a. m.      1,395 Contador1.class
16/03/2018  05:40 a. m.      3,023 Contador1.java
               4 archivos      11,478 bytes
               2 dirs  790,813,032,448 bytes libres

C:\Users\cat-b\Desktop\3.2>java Contador
C:\Users\cat-b\Desktop\3.2>java Contador
```

Se escoge el contador “Nuevo contador”



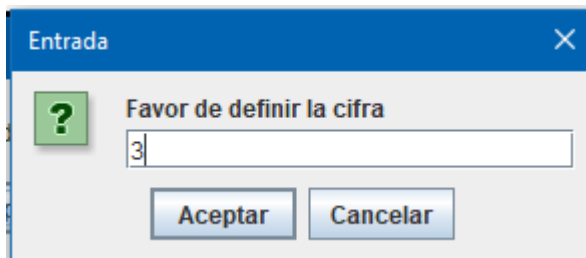
La cantidad que nos muestra es cero, si se elige “Mostrar valor inicial” Se muestra el valor de cero



Se elige “Avanzar una unidad” y le suma al valor que tenía “1” y se muestra en la salida



Si se elige “Avanzar una cifra especifica” muestra un InputDialog donde escribimos la cifra “3” y se le suma al valor que ya se tenía

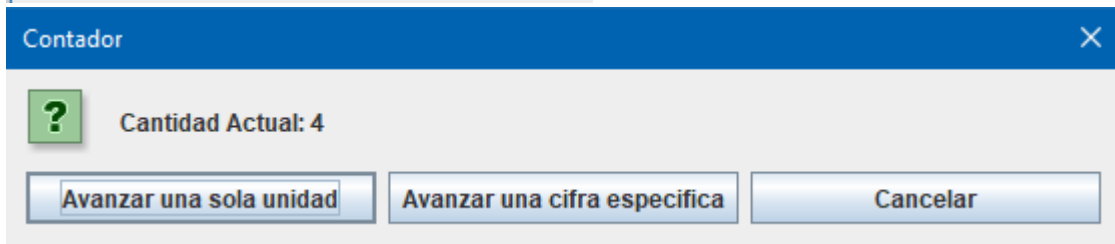


Entrada

Favor de definir la cifra

3

Aceptar Cancelar

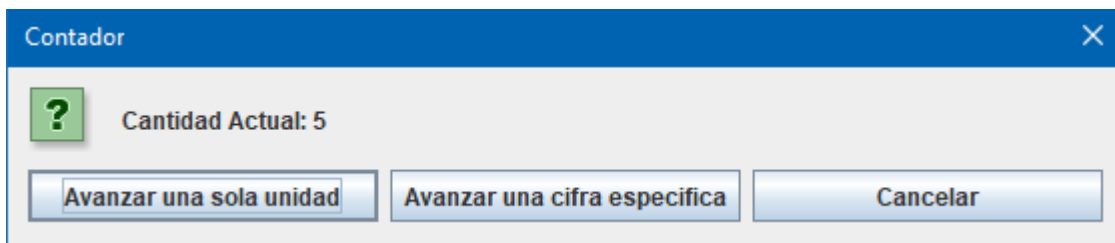


Contador

Cantidad Actual: 4

Avanzar una sola unidad Avanzar una cifra especifica Cancelar

Si se elige "Avanzar una sola unidad" Se muestra el valor guardado anterior más uno.

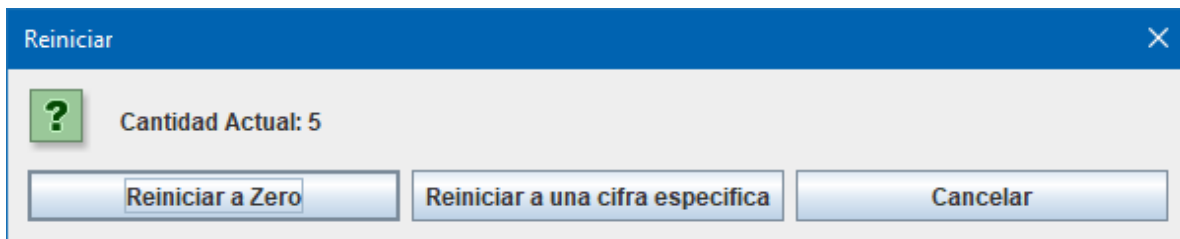


Contador

Cantidad Actual: 5

Avanzar una sola unidad Avanzar una cifra especifica Cancelar

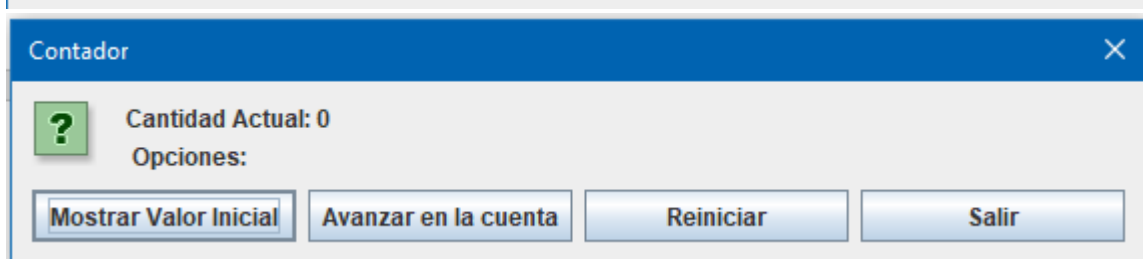
Al reiniciar el contador se puede elegir si desde cero o una cifra en específico, en este caso se eligió desde cero.



Reiniciar

Cantidad Actual: 5

Reiniciar a Zero Reiniciar a una cifra especifica Cancelar



Contador

Cantidad Actual: 0

Opciones:

Mostrar Valor Inicial Avanzar en la cuenta Reiniciar Salir

Ahora eligiendo la opción de "Contador con valor específico"

```
C:\> Símbolo del sistema - java Contador

C:\Users\cat-b\Desktop>cd 3.2

C:\Users\cat-b\Desktop\3.2>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 5CD2-57CE

Directorio de C:\Users\cat-b\Desktop\3.2

17/03/2018  07:26 p. m.    <DIR>          .
17/03/2018  07:26 p. m.    <DIR>          ..
16/03/2018  07:26 p. m.    <DIR>          .
16/03/2018  07:26 p. m.    <DIR>          ..
16/03/2018  07:26 p. m.    <DIR>          .
16/03/2018  07:26 p. m.    <DIR>          ..

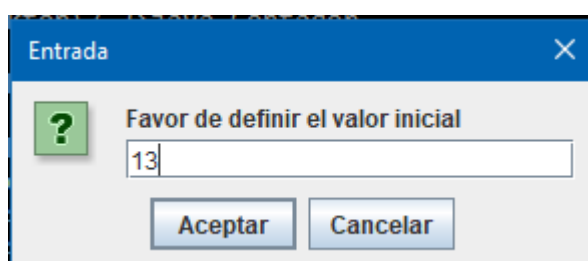
C:\Users\cat-b\Desktop\3.2>java Contador

C:\Users\cat-b\Desktop\3.2>java Contador

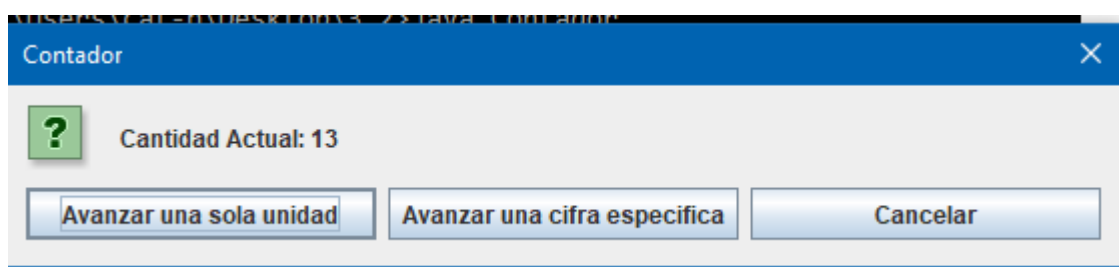
C:\Users\cat-b\Desktop\3.2>java Contador
Exception in thread "main" java.lang.NumberFormatException: null
    at java.base/java.lang.Integer.parseInt(Unknown Source)
    at java.base/java.lang.Integer.parseInt(Unknown Source)
    at Contador.main(Contador.java:23)

C:\Users\cat-b\Desktop\3.2>java Contador
```

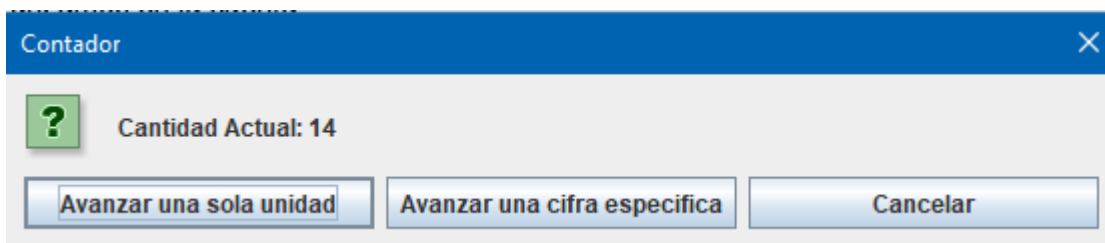
Muestra un InputPane



Avanzamos en la cuenta



Avanzando solo una unidad tres veces

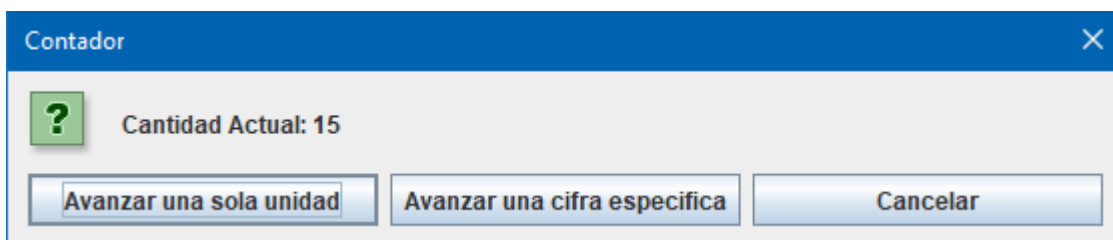


Contador

?

Cantidad Actual: 14

Avanzar una sola unidad Avanzar una cifra especifica Cancelar

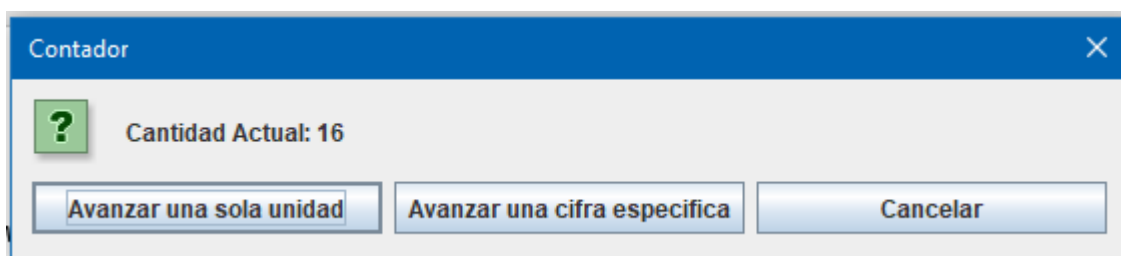


Contador

?

Cantidad Actual: 15

Avanzar una sola unidad Avanzar una cifra especifica Cancelar



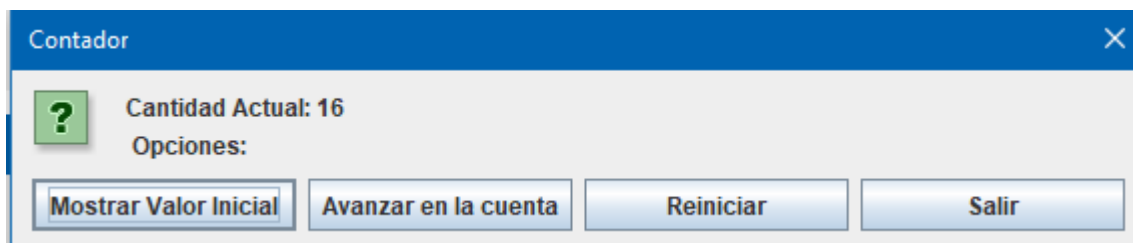
Contador

?

Cantidad Actual: 16

Avanzar una sola unidad Avanzar una cifra especifica Cancelar

Cancelando



Contador

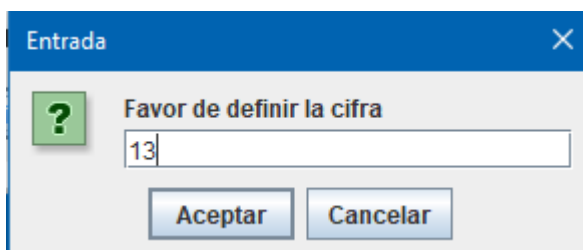
?

Cantidad Actual: 16

Opciones:

Mostrar Valor Inicial Avanzar en la cuenta Reiniciar Salir

Avanzando cifra especifica



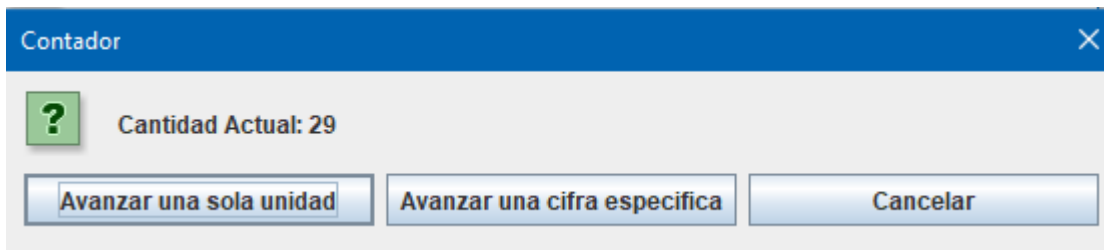
Entrada

?

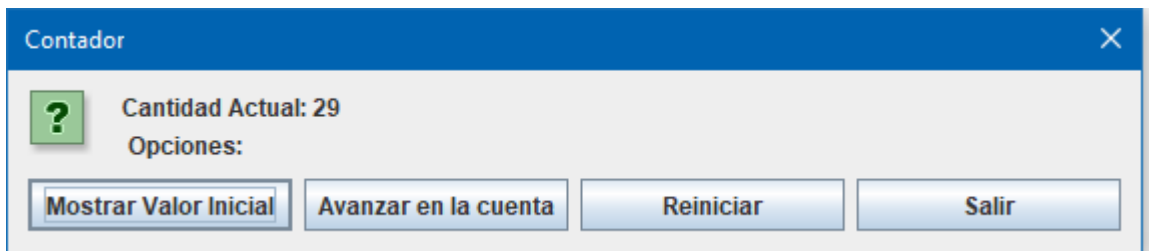
Favor de definir la cifra

13

Aceptar Cancelar



Cancelando y saliendo



```
04% Símbolo del sistema
C:\Users\cat-b\Desktop>cd 3.2

C:\Users\cat-b\Desktop\3.2>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 5CD2-57CE

Directorio de C:\Users\cat-b\Desktop\3.2

17/03/2018  07:26 p. m.      <DIR>          .
17/03/2018  07:26 p. m.      <DIR>          ..
16/03/2018  05:40 a. m.           2,384 Contador.class
16/03/2018  05:40 a. m.           4,676 Contador.java
16/03/2018  05:40 a. m.           1,395 Contador1.class
16/03/2018  05:40 a. m.           3,023 Contador1.java
               4 archivos             11,478 bytes
               2 dirs  790,813,032,448 bytes libres

C:\Users\cat-b\Desktop\3.2>java Contador

C:\Users\cat-b\Desktop\3.2>java Contador

C:\Users\cat-b\Desktop\3.2>java Contador
Exception in thread "main" java.lang.NumberFormatException: null
    at java.base/java.lang.Integer.parseInt(Unknown Source)
    at java.base/java.lang.Integer.parseInt(Unknown Source)
    at Contador.main(Contador.java:23)

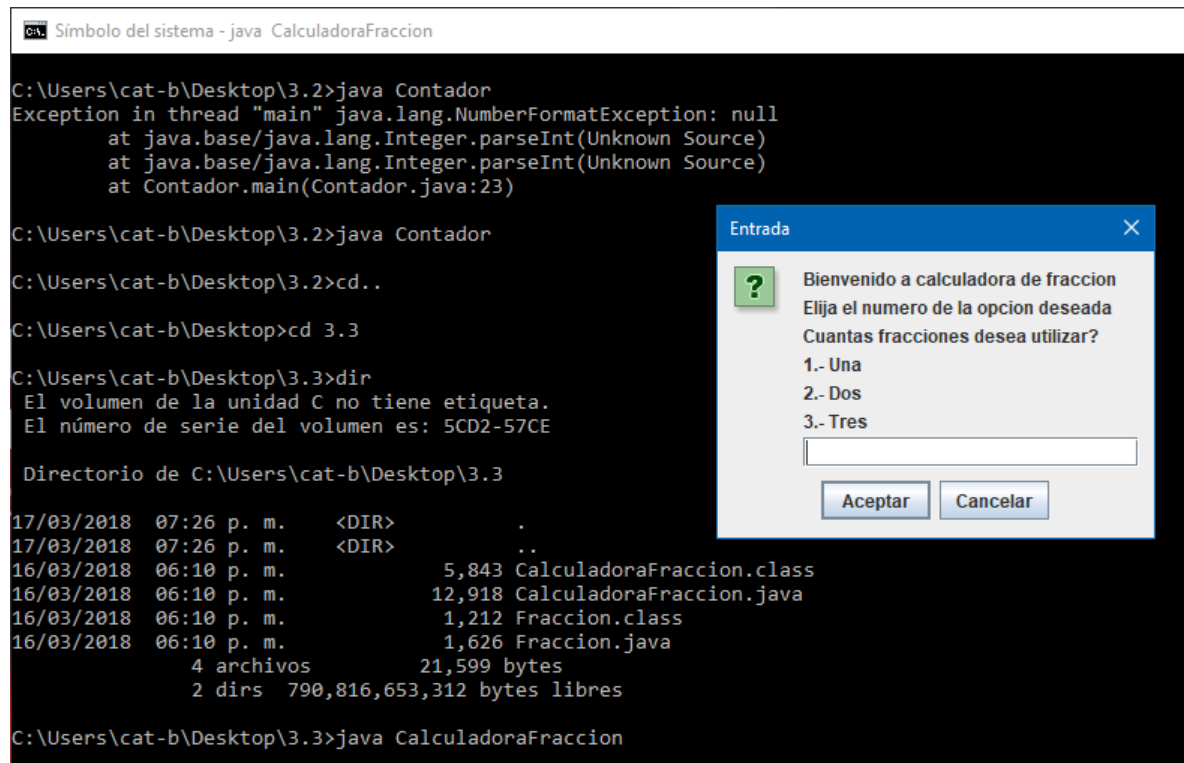
C:\Users\cat-b\Desktop\3.2>java Contador

C:\Users\cat-b\Desktop\3.2>
```

Tercer programa

Aplicación que realice operaciones básicas con una, dos o tres fracciones.

Se muestra la ejecución del programa



The screenshot shows a Windows command prompt window titled "Símbolo del sistema - java CalculadoraFraccion" and a Java application window titled "Entrada".

The command prompt shows the following commands and output:

```
C:\Users\cat-b\Desktop\3.2>java Contador
Exception in thread "main" java.lang.NumberFormatException: null
    at java.base/java.lang.Integer.parseInt(Unknown Source)
    at java.base/java.lang.Integer.parseInt(Unknown Source)
    at Contador.main(Contador.java:23)

C:\Users\cat-b\Desktop\3.2>java Contador

C:\Users\cat-b\Desktop\3.2>cd..

C:\Users\cat-b\Desktop>cd 3.3

C:\Users\cat-b\Desktop\3.3>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 5CD2-57CE

Directorio de C:\Users\cat-b\Desktop\3.3

17/03/2018  07:26 p. m.    <DIR>          .
17/03/2018  07:26 p. m.    <DIR>          ..
16/03/2018  06:10 p. m.             5,843 CalculadoraFraccion.class
16/03/2018  06:10 p. m.            12,918 CalculadoraFraccion.java
16/03/2018  06:10 p. m.             1,212 Fraccion.class
16/03/2018  06:10 p. m.             1,626 Fraccion.java
                4 archivos             21,599 bytes
                2 dirs  790,816,653,312 bytes libres

C:\Users\cat-b\Desktop\3.3>java CalculadoraFraccion
```

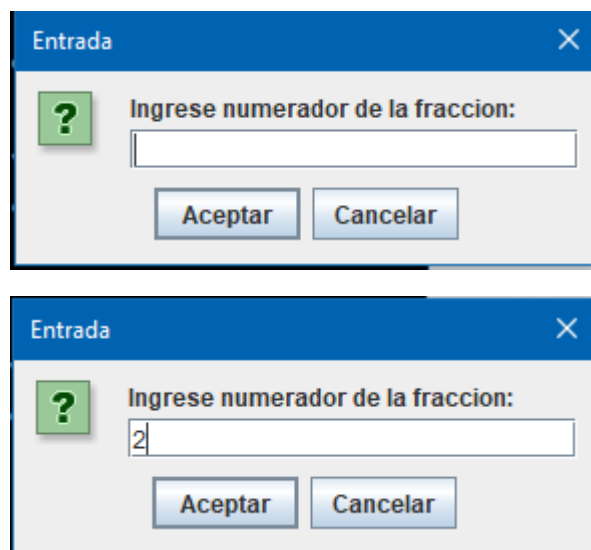
The Java application window "Entrada" displays a green question mark icon and the following text:

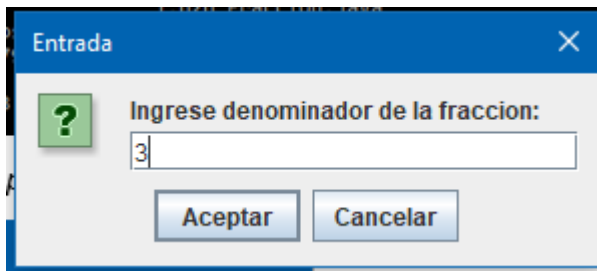
Bienvenido a calculadora de fraccion
Elija el numero de la opcion deseada
Cuantas fracciones desea utilizar?

1.- Una
2.- Dos
3.- Tres


Below the text is an empty text input field and two buttons: "Aceptar" and "Cancelar".

Eligiendo la primera opción





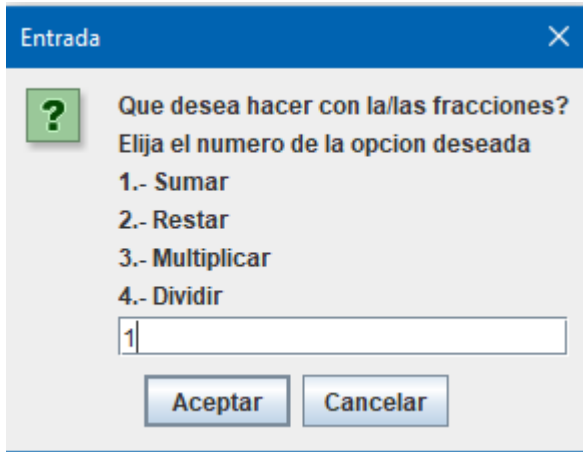
Entrada

 Ingrese denominador de la fraccion:


3

Aceptar Cancelar

Se tiene la primera fracción



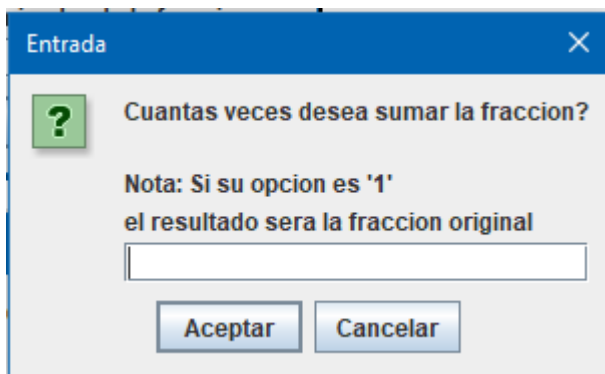
Entrada

 Que desea hacer con la/las fracciones?
Elija el numero de la opcion deseada


- 1.- Sumar
- 2.- Restar
- 3.- Multiplicar
- 4.- Dividir

1

Aceptar Cancelar



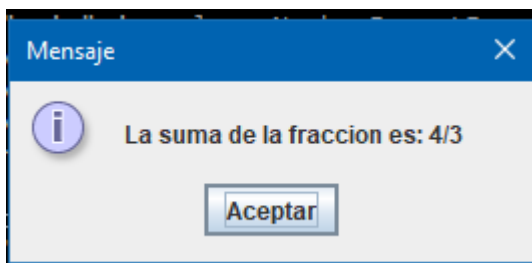
Entrada

 Cuantas veces desea sumar la fraccion?


Nota: Si su opcion es '1'
el resultado sera la fraccion original

Aceptar Cancelar

Le sumamos dos

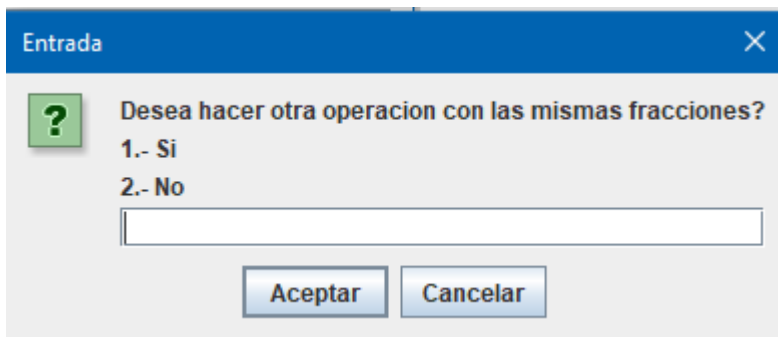


Mensaje


 La suma de la fraccion es: $\frac{4}{3}$

Aceptar

Después del resultado al aceptar se muestra otra pantalla donde pregunta si se desea hacer más operaciones, eligiendo si



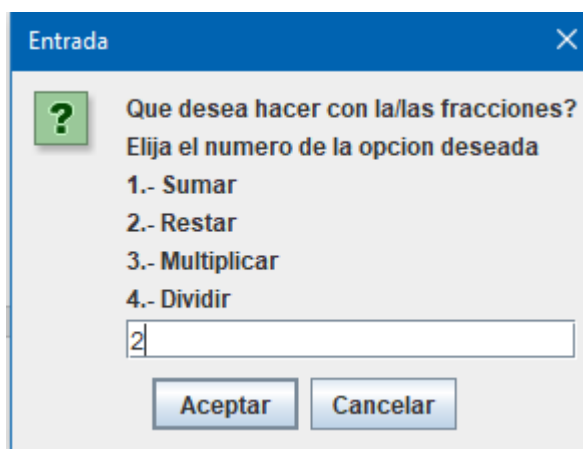
Entrada

 Desea hacer otra operacion con las mismas fracciones?


1.- Si
2.- No

Aceptar Cancelar

Ahora división



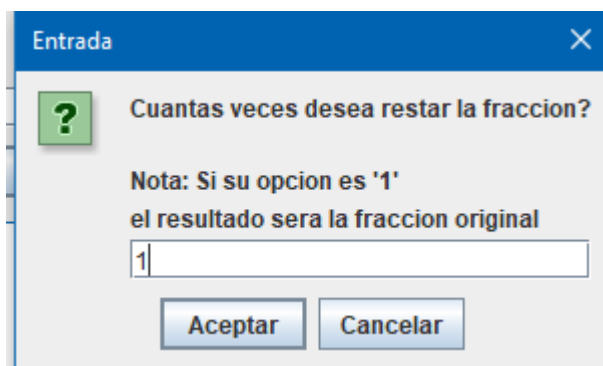
Entrada

 Que desea hacer con la/las fracciones?
Elija el numero de la opcion deseada


1.- Sumar
2.- Restar
3.- Multiplicar
4.- Dividir

Aceptar Cancelar

Le restamos uno

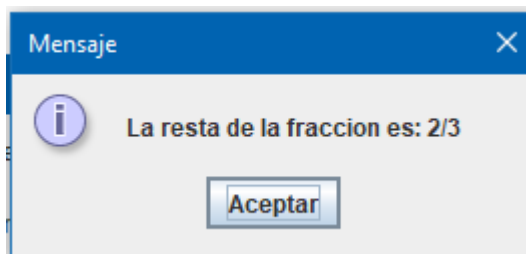


Entrada


 Cuantas veces desea restar la fraccion?

Nota: Si su opcion es '1'
el resultado sera la fraccion original

Aceptar Cancelar




Mensaje

 La resta de la fraccion es: $\frac{2}{3}$

Aceptar


Pregunta nuevamente si queremos realizar otra operación

Entrada ✕

 **Que desea hacer con la/las fracciones?**
Elija el numero de la opcion deseada


- 1.- Sumar
- 2.- Restar
- 3.- Multiplicar
- 4.- Dividir

Entrada ✕

 **Cuantas veces desea multiplicar la fraccion?**


Nota: Si su opcion es '1'
el resultado sera la fraccion original

Mensaje ✕

 **La multiplicacion de la fraccion es: 4/9**

Nuevamente pregunta y se elige si

Entrada ✕

 **Desea hacer otra operacion con las mismas fracciones?**

- 1.- Si
- 2.- No

Se elige la última operación

Entrada

?

Que desea hacer con la/las fracciones?
Elija el numero de la opcion deseada

- 1.- Sumar
- 2.- Restar
- 3.- Multiplicar
- 4.- Dividir

4

Aceptar Cancelar

Entrada

?

Cuantas veces desea dividir la fraccion?

Nota: Si su opcion es '1'
el resultado sera la fraccion original

2

Aceptar Cancelar

Mensaje

i

La division de la fraccion es: 1/1

Aceptar

Ahora se elige la segunda opción

Entrada

?

Desea hacer otra operacion con las mismas fracciones?

- 1.- Si
- 2.- No

2

Aceptar Cancelar

Se escoge uno para realizar diferentes pruebas

Entrada

?

Desea nuevas fracciones?

1.- Si

2.- No (Terminara el programa)

1

Aceptar Cancelar

Se muestra la pantalla de inicio del programa, ahora eligiendo dos fracciones

Entrada

?

Bienvenido a calculadora de fraccion

Elija el numero de la opcion deseada

Cuantas fracciones desea utilizar?

1.- Una

2.- Dos

3.- Tres

2

Aceptar Cancelar

Para la primera fracción

Entrada

?

Ingrese numerador de la primera fraccion:

1

Aceptar Cancelar

Entrada

?


Ingrese denominador de la primera fraccion:

2

Aceptar Cancelar


Para la segunda fracción

Entrada

 Ingrese numerador de la segunda fraccion:

Aceptar Cancelar


Entrada

 Ingrese denominador de la segunda fraccion:

Aceptar Cancelar

Se muestra la pantalla donde se pueden hacer las operaciones con las fracciones dadas, se elige la primera opción.

Entrada


 Que desea hacer con la/las fracciones?
Elija el numero de la opcion deseada

- 1.- Sumar
- 2.- Restar
- 3.- Multiplicar
- 4.- Dividir

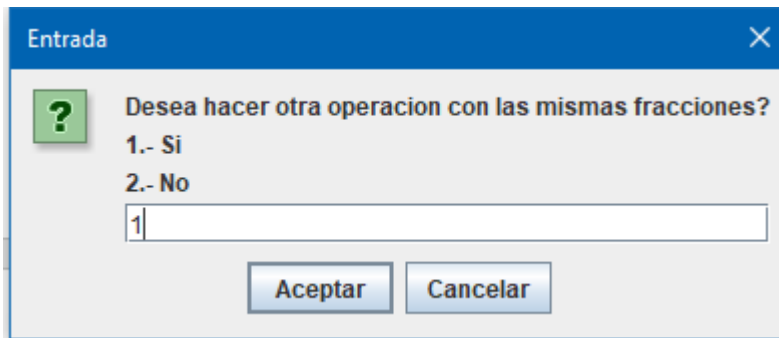
Aceptar Cancelar

Se muestra el resultado.


Mensaje

 La suma de las fracciones es: 9/10

Aceptar



Entrada

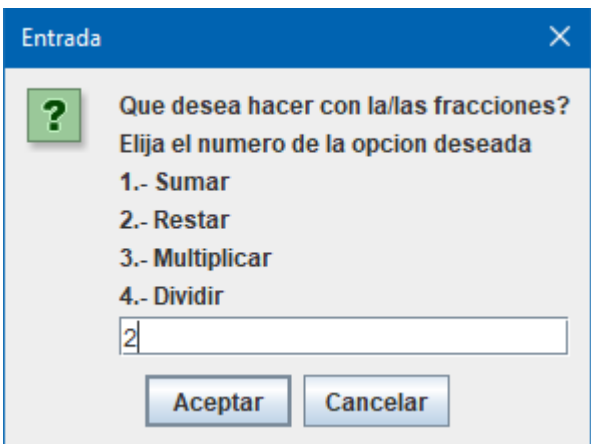
 Desea hacer otra operacion con las mismas fracciones?

1.- Si
2.- No


1

Aceptar Cancelar

Después de elegir esa opción, se muestra el menú principal de la aplicación, donde ahora se elige la segunda opción.



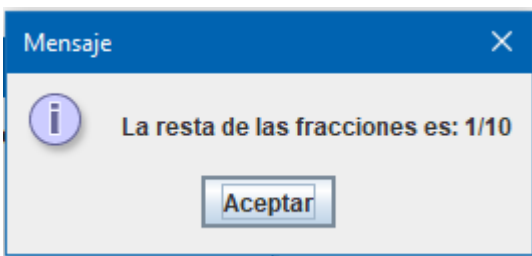
Entrada

 Que desea hacer con la/las fracciones?
Elija el numero de la opcion deseada


1.- Sumar
2.- Restar
3.- Multiplicar
4.- Dividir

2

Aceptar Cancelar

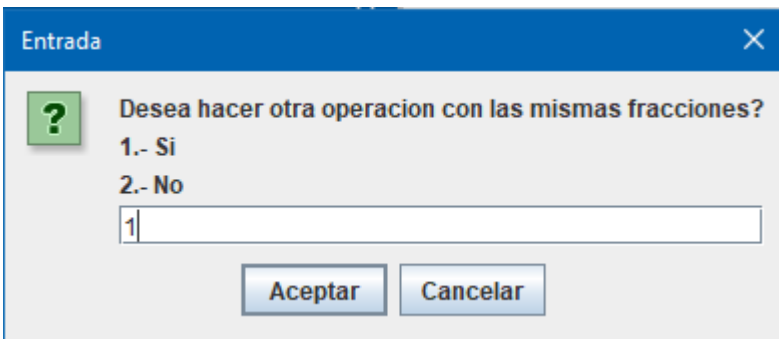


Mensaje


 La resta de las fracciones es: $\frac{1}{10}$

Aceptar

Después de mostrar el resultado, aparece nuevamente la entrada



Entrada

 Desea hacer otra operacion con las mismas fracciones?

1.- Si
2.- No

1

Aceptar Cancelar

Ahora con la tercera opción

Entrada

?

Que desea hacer con la/las fracciones?
Elija el numero de la opcion deseada

- 1.- Sumar
- 2.- Restar
- 3.- Multiplicar
- 4.- Dividir

3

Aceptar Cancelar

Se muestra el resultado

Mensaje

i

La multiplicacion de las fracciones es: $\frac{1}{5}$

Aceptar

Aceptando y eligiendo nuevamente uno

Entrada

?

Desea hacer otra operacion con las mismas fracciones?

- 1.- Si
- 2.- No

1

Aceptar Cancelar

Entrada

?

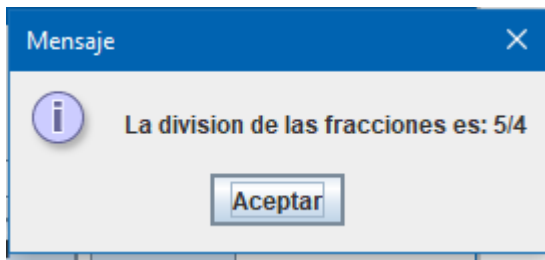
Que desea hacer con la/las fracciones?
Elija el numero de la opcion deseada

- 1.- Sumar
- 2.- Restar
- 3.- Multiplicar
- 4.- Dividir

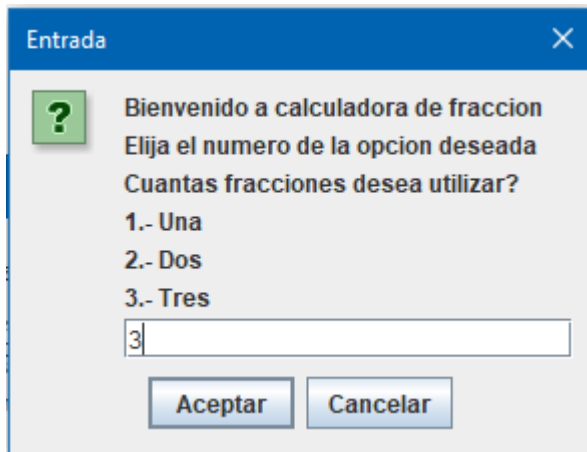
4

Aceptar Cancelar

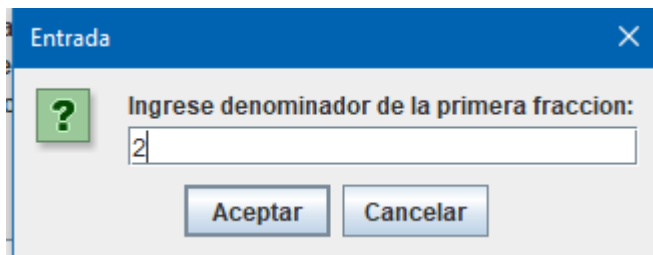
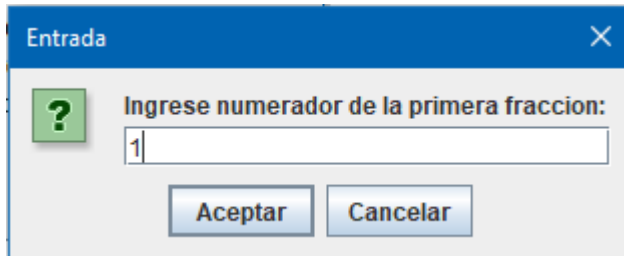
Se muestra el resultado



Ahora con tres fracciones



Para la primera fracción



Para la segunda fracción

Entrada

? Ingrese numerador de la segunda fraccion:

2

Aceptar Cancelar

Entrada

? Ingrese denominador de la segunda fraccion:

3

Aceptar Cancelar

Para la tercera fracción

Entrada

? Ingrese numerador de la tercera fraccion:

4

Aceptar Cancelar

Entrada

? Ingrese denominador de la tercera fraccion:

5

Aceptar Cancelar

Se muestra las opciones que se puede realizar con las fracciones dadas, eligiendo la primera.

Entrada

?

Que desea hacer con la/las fracciones?
Elija el numero de la opcion deseada

- 1.- Sumar
- 2.- Restar
- 3.- Multiplicar
- 4.- Dividir

1

Aceptar Cancelar

Mensaje

i

La suma de las fracciones es: 59/30

Aceptar

Después se muestra una pantalla preguntando lo siguiente

Entrada

?

Desea hacer otra operacion con las mismas fracciones?

- 1.- Si
- 2.- No

1

Aceptar Cancelar

Eliendo uno, y ahora seleccionando dos

Entrada

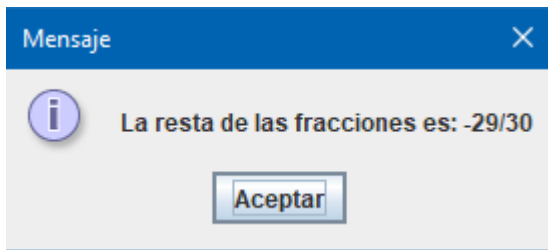
?

Que desea hacer con la/las fracciones?
Elija el numero de la opcion deseada

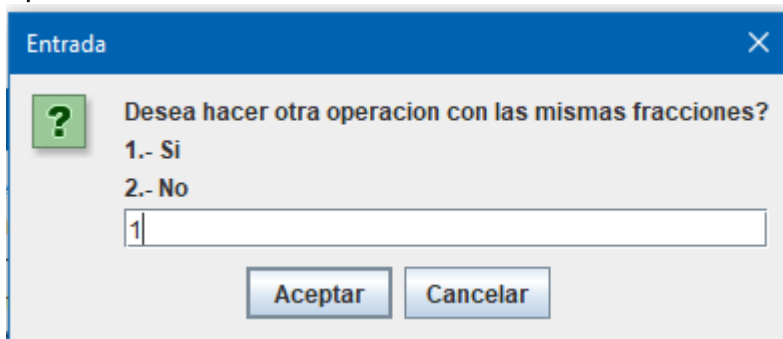
- 1.- Sumar
- 2.- Restar
- 3.- Multiplicar
- 4.- Dividir

2

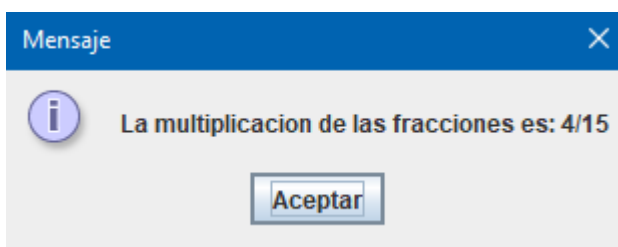
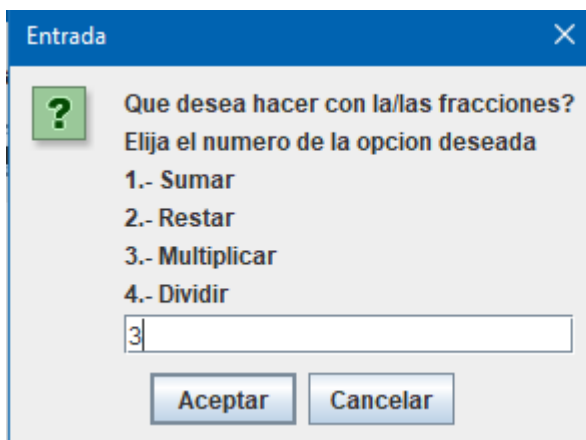
Aceptar Cancelar



Aparece nuevamente




Se elige la tercera opción



Nuevamente nos pregunta si continuaremos

Entrada


 Desea hacer otra operacion con las mismas fracciones?

1.- Si
2.- No

Aceptar Cancelar

Eligiendo la tercera opcion


Entrada

 Que desea hacer con la/las fracciones?
Elija el numero de la opcion deseada

1.- Sumar
2.- Restar
3.- Multiplicar
4.- Dividir

Aceptar Cancelar


Mensaje

 La division de las fracciones es: 15/16

Aceptar

Eligiendo no

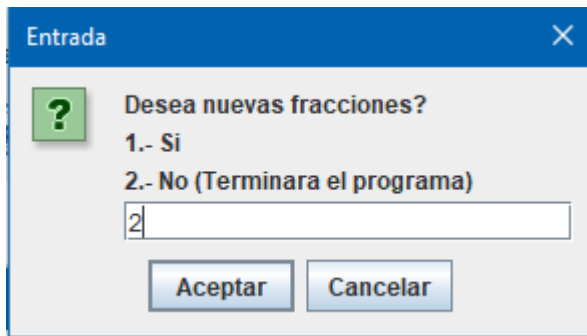
Entrada

 Desea hacer otra operacion con las mismas fracciones?

1.- Si
2.- No

Aceptar Cancelar

Y nuevamente no



Se termina el programa.

Conclusiones

Dávila García Rivas Emiliano

En mi caso, tuve la oportunidad de practicar el uso de JOptionPane con el contador; a medida que abstraía el objeto para diseñar las clases que servirían para resolver el problema, se presentaron algunos problemas particulares. Estos causados tanto por mi inexperiencia con el uso de JOptionPane, como con la sobrecarga de los métodos y constructores. Un claro ejemplo del primer tipo de circunstancia que salió a luz durante el diseño fue la falta de conocimiento en cuanto a los parámetros de esta clase, ya que mi idea original fue la utilización de un arreglo dinámico con ArrayList para el uso de múltiples contadores, sin embargo, esto no me fue posible.

Además, los problemas en cuanto al sobrecargo de constructores resulto en errores extraños que no logre obtener una explicación de estos errores, sin embargo, por medio de un cambio del diseño de la clase Contador1 logre obtener el mismo resultado con sobrecarga de métodos.

Luciano Espina Melisa

Se aplico el paradigma orientado a objetos y con eso se uso los ciclos lógicos, se pudo implementar la sobrecarga de métodos y aunque faltó la sobrecarga de constructores, pero se puede implementar y mejorar.

Además, ahora se puede implementar la función de JOptionPane para que los programas sean vistos mejor estéticamente, se le da una presentación para que el usuario pueda sentirse más cómodo y así también que pueda entender mejor lo que se hace con ese programa.

Se cambiaron los modificadores de acceso y se implemento "get" esto para poder acceder a las variables privadas.

Se abstrajeron los objetos para crear las clases y utilizar.

Ramos Mesas Edgar Alaín

Cuando hablamos de sobrecargar, ya sea de métodos o constructores, nos referimos al hecho de tener diferentes posibles soluciones a un mismo problema. En cada uno de los métodos podemos notar, por ejemplo, que, aunque la metodología puede parecer la misma, lo cierto es que se toman en cuenta muchas más cosas que solo eso, pues en la sobrecarga de métodos, como ya es sabido, se toman en cuenta el número de variables de entrada a la hora de "proceder". De forma similar, con los constructores, sabemos que es posible asignar diferentes valores (o estados) a objetos de la misma clase mediante la sobrecarga de dichos constructores. Quizá el ejemplo más claro de esto es la calculadora elaborada durante el desarrollo de esta práctica, pues en ella es más fácil poder apreciar los diferentes casos a tratar, y la forma en la que la sobrecarga de métodos y constructores, hacen su trabajo. Esto quiere decir que, al sobrecargar métodos podemos especificar, de alguna manera, la forma en la

que se trataran las diferentes situaciones que pueden presentarse en nuestro planteamiento original.

De este modo, podemos concluir que la sobrecarga, tanto de métodos como de constructores, nos permite contemplar varios casos a tratar en un mismo problema, lo cual es realmente practico ya que de este modo ampliamos el campo de acción, y a su vez, podemos particularizar cada uno de los diferentes casos a tratar.

“Por cada problema siempre habrá, al menos, una posible solución”

Bibliografía

- ☞ [1] S/A Curso Online [Online] Available: <http://www.aulafacil.com/cursos/l13573/informatica/programacion/java-basico/sobrecargar-metodos-y-constructores>
- ☞ [2] S/A Métodos, sobrecarga de métodos, Argumentos pasados por valor y por referencia | Curso de iniciación [Online] Available: <http://www.mundonet.es/metodos-sobrecarga-de-metodos-argumentos-pasados-por-valor-y-por-referencia-metodos-static.html?Pg=Entrega6.htm>
- ☞ [3] Luis E. Aponte I. Constructores en java [Online] Available: <http://programandoenjava.over-blog.es/article-32829724.html>
- ☞ [4] Virginia Hernández, Orientación a objetos (III) – sobrecarga constructores [Online] Available: <https://preparandoscjp.wordpress.com/2011/05/09/orientacion-a-objetos-iii/>

Anexos

Práctica 3.1

Persona

/**

* M.A:Publico

* Tipo: Clase

* Esta clase representa los datos de una persona

*/

public class Persona{

 /**

 * M.A: Publico

 * Tipo:entero

 * Atributo correspondiente a la edad de la persona

 */

 public int edad;

 /**

 * M.A: Publico

 * Tipo:String

 * Atributo correspondiente al nombre de la persona

 */

 public String nombre;

 /**

 * M.A: Publico

 * Tipo:String

 * Atributo correspondiente a la ocupacion de la persona

 */

 public String ocupacion;

```
/**
 * M.A: Privado
 * Tipo:String
 * Atributo correspondiente al genero de la persona
 */
    public String sexo;
```

```
/**
 * Constructor por default que provoca en el momento de creacion del objeto que los valores
 sean
```

```
 * puestos en sus valores por defecto.
 */
```

```
    public Persona()
    {

    }
}
```

```
/**
 * Constructor de modo de registro basico
 * parametros de entrada simples
 * nombre y edad
 */
```

```
    public Persona(String n, int e)
    {
        nombre = n;
        edad = e;
    }
}
```

```
/**
 * Constructor de modo de registro avanzado
 * parametros de entrada completos
```

```

        * nombre, edad, ocupacion, genero
    */

    public Persona(String n, int e, String o, String s)
    {
        nombre = n;
        edad = e;
        ocupacion = o;
        sexo = s;
    }

}

DatosPersona
import javax.swing.JOptionPane;///Importacion del paquete que permite el uso de JOptionPane

/**
 * Programa disenado para pedir y mostrar los datos de una persona
 * con difernetes tipos de registros y sobrecarga de constryctores
 */
public class DatosPersona
{
    public static void main(String args[])
    {
        /**
         * Variables que RECIBIRAN valores a traves de JOptionPane
         */
        String name;
        String age;
        String occupation;
        String gen;
        /**

```

```

        *Variables SETEABLES con JOptionPane
        */
        int resp;
        int i;
        /**
        * Segmento que permite al usuario elegir si quiere realizar un registro simple(nombre y
        edad)
        * o si prefiere un registro mas detallado(nombre, edad, ocupacion y genero)
        */

        String[] options = {"Registro Simple", "Registro Avanzado"};

        int seleccion = JOptionPane.showOptionDialog(null, "Seleccione un tipo de
        registro", "REGISTRO",
        JOptionPane.DEFAULT_OPTION, JOptionPane.QUESTION_MESSAGE, null, options,
        options[0]);

        /**
        * Verificacion de seleccion, seleccion de tipo de registro a traves de un
        JOptionPane

        * caso 0 registro simple
        */
        if(seleccion==0)
        {

            resp = JOptionPane.showConfirmDialog(null, "Desea registrar a una
            persona?", "REGISTRO", JOptionPane.YES_NO_OPTION);

            /**
            * Verificacion de proceso a realizar(Confirmacion)
            * realizado a traves de JOptionPane
            */
            if(resp==0)
            {

                name = JOptionPane.showInputDialog("Por favor introduzca su
                nombre");

```

```

edad");

        age = JOptionPane.showInputDialog("Por favor introduzca su

        int ag = Integer.parseInt(age);

        Persona p1 = new Persona(name, ag);

        System.out.println(p1.nombre + p1.edad);

    }

    else

        return;

}

/**
 * Verificacion de seleccion, seleccion de tipo de registro a traves de un
JOptionPane

 * caso 1 registro avanzado

 */

else if(seleccion==1)

{

    personas

        Persona p[] = null; //Creacion de un array de objetos, registro multiple de

        p = new Persona [5];

        for(i=0; i<p.length;i++)

        {

            /**

            personas

                * Confirmacion de registro nuevo, puede terminar el registro de

                */

                resp = JOptionPane.showConfirmDialog(null, "Desea registrar a

                una persona?", "REGISTRO", JOptionPane.YES_NO_OPTION);

                if(resp==0)

                {

                    /**

                    * Obtencion de datos a traves de JOptionPane

```

```

metodos                                * asignacion de valores y conversion de datos para uso de

                                        */

introduzca su nombre");               name = JOptionPane.showInputDialog("Por favor

su edad");                           age = JOptionPane.showInputDialog("Por favor introduzca

Introduzca su ocupacion");           ocupation = JOptionPane.showInputDialog("Por favor

genero");                             gen = JOptionPane.showInputDialog("Introduzca su

                                        int ed = Integer.parseInt(age);
                                        p[i] = new Persona(name, ed, ocupation, gen);
                                        System.out.println(p[i].nombre + p[i].edad + p[i].ocupacion
+ p[i].sexo);

                                        }
                                        else
                                        System.exit(0);//Termina el proceso si se desea detener el registro

de personas                           }

                                    }

                                }

}

```

Práctica 3.2

Contador1

import javax.swing.*; /// Importacion del paquete javax.swing para el uso de JOptionPane

/**

* M.A:Publico

* Tipo: Clase

* Esta clase representa el ente de un contador que tiene ciertos tipos de habilidades tales como
Mostrar Valores, Avanzar y Reiniciar

*/

public class Contador1

{

/**

* M.A: Privado

* Tipo:entero

* Atributo correspondiente a el valor inicial del contador

*/

private int valorInicial;

/**

* M.A: Provado

* Tipo: entero

* Atributo correspondiente a el valor actual del contador

*/

private int valorActual;

/**

* Constructor por default que provoca en el momento de creacion del objeto que los valores
sean

* puestos en sus valores basicos, es decir, en ceros.

*/

Contador1()

{

valorInicial=0;///Atributos del Objeto

valorActual=0;

}

/**

* M.A:Publico

* Retorno: void

* Argumentos: Un solo entero llamado "i"

* Metodo que permite la modificacion a los atributos del objeto, cambia los dos atributos del objeto a cualquiera

* especifique el usuario

*/

public void setValor(int i)

{

valorInicial=i;

valorActual=i;

}

/**

* M.A: Publico

* Retorno: void

* Argumentos: N/A

* Metodo que activa una ventana JOptionPane, tipo Message Dialog, la cual muestra el valor inicial del contador

*/

public void mostrarValorInicial()


```
{
    JOptionPane.showMessageDialog(null, "El valor inicial de este contador fue: " + valorInicial);
}

/**
 * M.A: Publico
 * Retorno: void
 * Argumentos: N/A
 * Metodo que activa una ventana JOptionPane, tipo Message Dialog, la cual muestra el valor
actual del contador
 */
public void mostrarValorActual()
{
    JOptionPane.showMessageDialog(null, "El valor actual de este contador es: " + valorActual);
}

/**
 * M.A: Publico
 * Retorno: entero
 * Argumentos: N/A
 * Metodo que esta disenado para retornar el valor del atributo para su utilizacion por parte de
otras clases
 */
public int getValorActual()

{
    return valorActual;
}

/**
```

```
* M.A: Publico
* Retorno: void
* Argumentos: N/A
* Metodo que regresa los valores a zero.
*/
```

```
public void reiniciar()
{
    valorActual=0;
    valorInicial=0;
}
```

```
/**
```

```
* M.A: Publico
* Retorno: void
* Argumentos: N/A
* Metodo sobrecargado que reinicia los valores a valores definidos por el usuario
*/
```

```
public void reiniciar(int r)
{
    valorActual=r;
    valorInicial=r;
}
```

```
/**
```

```
* M.A: Publico
* Retorno: void
* Argumentos: Un solo entero
* Metodo que le suma a valorActual la cantidad recibida por parte del entero "a"
*/
```

```

    public void avanzar(int a)
    {
        valorActual+=a;
    }

}

```

Contador

import javax.swing.*;///Importacion del paquete que permite el uso de JOptionPane

```

/**
 * Programa disenado para demostrar la utilizacion de la clase Contador, ademas de servir como
 * una clase que pone en prueba los metodos de ella y sus atributos
 */
public class Contador
{
    public static void main( String args[] )
    {
        Contador1 c=new Contador1();///Creacion del objeto c tipo Contador1

        JOptionPane.showMessageDialog(null, "Contador",
"Contador",JOptionPane.INFORMATION_MESSAGE); ///ventana de JOptionPane que indica que el
objeto es un contador

        /**
         * Segmento que permite al usuario elegir si quiere que el contador sea un Contador Nuevo,
         es decir que los valores esten en zeros, o uno
         * que el punto de comienzo sea especificado por es usuario
         */
        Object[] op = {"Nuevo Contador","Contador con valor especifico"};

        int n = JOptionPane.showOptionDialog(null,"Escoja el tipo de contador","Tipo de
Contador",JOptionPane.YES_NO_OPTION,JOptionPane.QUESTION_MESSAGE,null,op,op[0]);

```

```

if (n==1)
{
    String inputValue = JOptionPane.showInputDialog("Favor de definir el valor inicial");
    int i = Integer.parseInt(inputValue);
    c.setValor(i);
}

/**
 * Segmento que permite al usuario elegir una de las cuatro opciones diferentes que le
    permiten utilizar diferentes atributos y metodos
 * del objeto, estas opciones son:
 * Mostrar Valor Inicial: Muestra valorInicial ya sea que haya sido por default o definido por el
    usuario
 */
Object[] op2 = {"Mostrar Valor Inicial", "Avanzar en la cuenta", "Reiniciar", "Salir"};
do
{
    n=-2;

    n = JOptionPane.showOptionDialog(null, "Cantidad Actual: "+ c.getValorActual()+"\n
    Opciones:", "Contador", JOptionPane.YES_NO_OPTION, JOptionPane.QUESTION_MESSAGE, null, op2,
    op2[0]);

    switch(n)
    {
        /**
         * Caso 0: Caso que aplica a Mostrar Valor inicial.
         * Llama al metodo mostrarValorInicial que utiliza JOptionPane para mostrar este valor.
         */
        case 0:
            c.mostrarValorInicial();

```

```

        break;

    /**
     * Caso 1: Caso que aplica para Avanzar el contador
     * Contiene un switch anidado que permite al usuario decidir si quiere avanzar por unidad
     o una cantidad definida por el usuario
     */
    case 1:
        Object[] op3 = {"Avanzar una sola unidad","Avanzar una cifra especifica","Cancelar"};
        int j;
        do{

            j = JOptionPane.showOptionDialog(null,"Cantidad Actual: "+ c.getValorActual()
            ,"Contador",JOptionPane.YES_NO_OPTION,JOptionPane.QUESTION_MESSAGE,null,op3,op3[0]);

            switch(j)
            {
                case 0:
                    int a=1;
                    c.avanzar(a);
                    break;
                case 1:
                    int i;
                    String inputValue = JOptionPane.showInputDialog("Favor de definir la cifra");
                    i = Integer.parseInt(inputValue);
                    c.avanzar(i);
                    break;
            }

        }while(j!=2);

```

```

        break;

    /**
     * Caso 2: Caso que aplica los metodos para reiniciar los valores del contador.
     */
    case 2:
        Object[] op4 = {"Reiniciar a Zero","Reiniciar a una cifra especifica","Cancelar"};
        int k;
        k = JOptionPane.showOptionDialog(null,"Cantidad Actual: "+ c.getValorActual()
        ,"Reiniciar",JOptionPane.YES_NO_OPTION,JOptionPane.QUESTION_MESSAGE,null,op4,op4[0]);
        if(k==0)
        {
            c.reiniciar();
        }
        else if(k==1)
        {
            int r;
            String inputValue = JOptionPane.showInputDialog("Favor de definir la cifra");
            r = Integer.parseInt(inputValue);
            c.reiniciar(r);
        }
        else
            break;
    }
}while(n!=3);

}

}

```

Práctica 3.3

Fracción

```
import java.lang.Math;
```

```
public class Fraccion{

    private int numerador;

    private int denominador;


    public Fraccion(){

        numerador = 0;

        denominador = 1;

    }


    public Fraccion(int x, int y){

        numerador = x;

        denominador = y;

    }


    public int getNumerador(){

        return this.numerador;

    }


    public int getDenominador(){

        return this.denominador;

    }


    public Fraccion sumar(Fraccion a, Fraccion b){

        Fraccion c = new Fraccion();

        c.numerador = a.numerador * b.denominador + b.numerador * a.denominador;
```

```
c.denominador = a.denominador * b.denominador;  
return c;  
}
```

```
public Fraccion restar(Fraccion a, Fraccion b){  
    Fraccion c = new Fraccion();  
    c.numerador = a.numerador * b.denominador - b.numerador * a.denominador;  
    c.denominador = a.denominador * b.denominador;  
    return c;  
}
```

```
public Fraccion multiplicar(Fraccion a, Fraccion b){  
    Fraccion c = new Fraccion();  
    c.numerador = a.numerador * b.numerador;  
    c.denominador = a.denominador * b.denominador;  
    return c;  
}
```

```
public Fraccion dividir(Fraccion a, Fraccion b){  
    return new Fraccion(a.numerador * b.denominador, a.denominador *  
b.numerador);  
}
```

```
private int mcd(){  
    int u = Math.abs(numerador);  
    int v = Math.abs(denominador);  
    if(v == 0){  
        return u;  
    }  
}
```



```
int r;  
while(v != 0){  
    r = u % v;  
    u = v;  
    v = r;  
}  
return u;  
}
```

```
public Fraccion simplificar(){  
    int dividir = mcd();  
    numerador /= dividir;  
    denominador /= dividir;  
    return this;  
}
```

```
}
```

CalculadoraFracciones

```
import javax.swing.JOptionPane;
```

```
public class CalculadoraFracciones{
```

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

```
//objetos de tipo Fraccion
```

```
        Fraccion f1;
```

```
        Fraccion f2;
```

```

        String a1 = JOptionPane.showInputDialog("Ingrese el numerador de la primera
fraccion: ");
        int aa1 = Integer.parseInt ( a1 );

        String b1 = JOptionPane.showInputDialog("Ingrese el denominador de la primera
fraccion: ");
        int bb1 = Integer.parseInt ( b1 );

        String a2 = JOptionPane.showInputDialog("Ingrese el numerador de la segunda
fraccion: ");
        int aa2= Integer.parseInt ( a2 );

        String b2 = JOptionPane.showInputDialog("Ingrese el denominador de la segunda
fraccion: ");
        int bb2 = Integer.parseInt ( b2 );

        f1 = new Fraccion(aa1, bb1);
        f2 = new Fraccion(aa2, bb2);

        Fraccion resul = new Fraccion();
        resultado.suma(f1,f2);

        int aux = f1.getnumerador();
        int auxresult = resul.getnumerador();

        System.out.println ( auxresult);
        System.out.println (aux);
    }

}

```