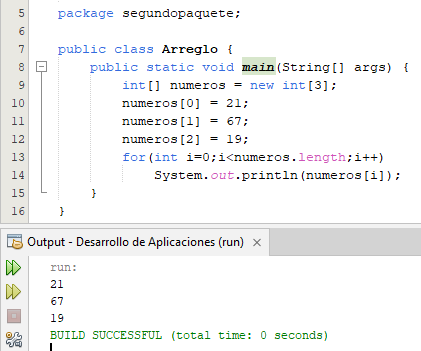
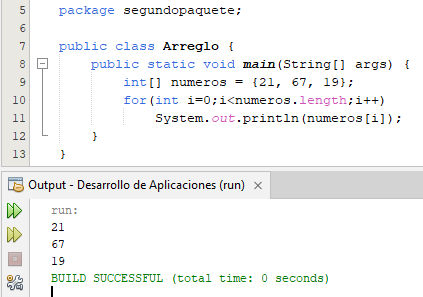
**Desarrollo Avanzado de Aplicaciones I**

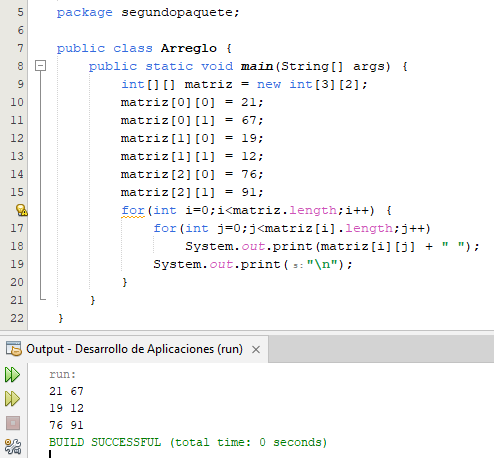
**Ejercicio 1**. Desarrollar una aplicación que declare, cree y luego inicialice un arreglo unidimensional de enteros y los muestre en pantalla usando una sentencia repetitiva y usando el atributo “length”.



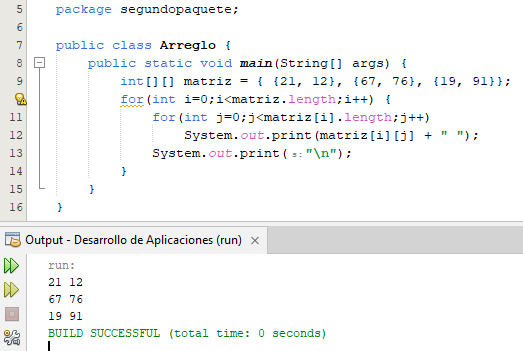
**Ejercicio 2**. Desarrollar una aplicación que declare e inicialice un arreglo unidimensional de enteros y los muestre en pantalla usando una sentencia repetitiva y usando el atributo “length”.



**Ejercicio 3**. Desarrollar una aplicación que declare, cree y luego inicialice un arreglo multidimensional de enteros y los muestre en pantalla usando una sentencia repetitiva anidada y usando el atributo “length”.

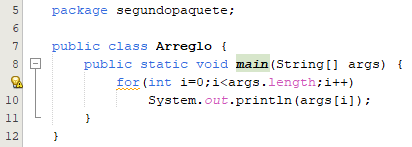


**Ejercicio 4**. Desarrollar una aplicación que declare e inicialice un arreglo multidimensional de enteros y los muestre en pantalla usando una sentencia repetitiva anidada y usando el atributo “length”.

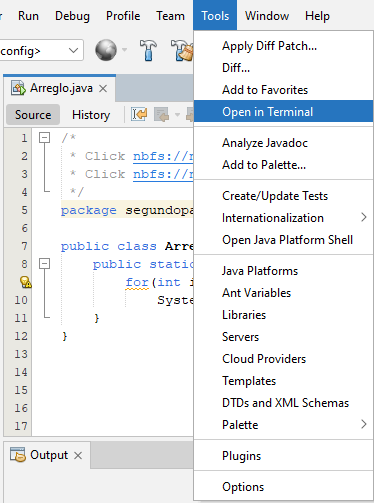


**Ejercicio 5**. Desarrollar una aplicación que imprima en pantalla argumentos pasados al método main.

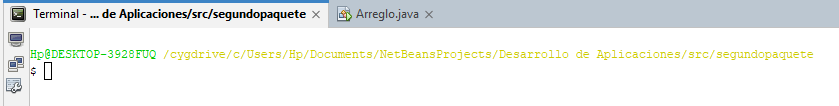
1. Crear una clase “Arreglo” dentro del paquete “segundopaquete” con el siguiente código:



1. Hacer click en el menú “Tools” opción “Open in Terminal”.



Le debe mostrar la siguiente pantalla dentro del mismo NetBeans:



1. Si le sale un mensaje de error que le falta el componente “cygwin”, descargarlo de la siguiente ruta: <https://www.cygwin.com/install.html>. Luego proceder con su instalación por defecto.



Por último, volver a probar el abrir el Terminal.

1. En la Terminal ejecutar las siguientes instrucciones:
2. Compilar el archivo Arreglo.java:

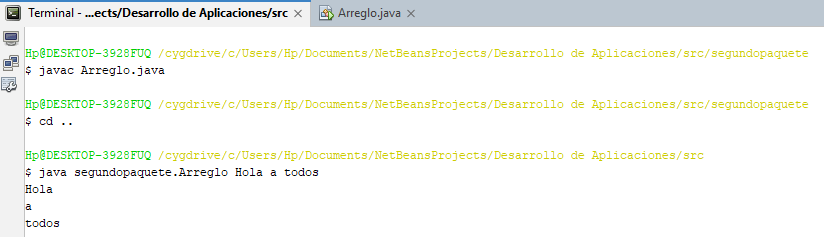
$ javac Arreglo.java

1. Debemos salir del directorio del paquete para que el interprete encuentre la clase:

$ cd ..

1. Ejecutar el programa pasando los argumentos:

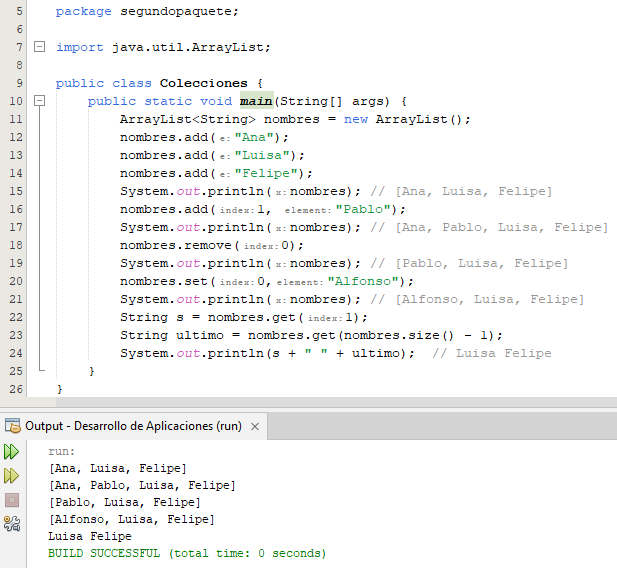
$ java segundopaquete.Arreglo Hola a todos



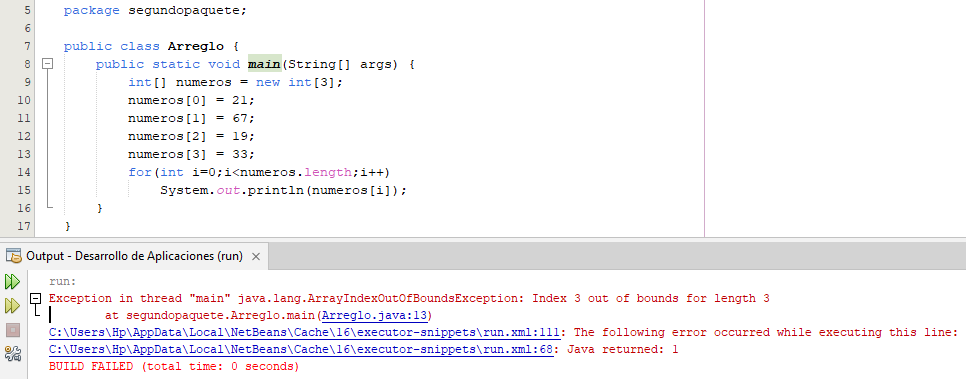
**Ejercicio 6**. Desarrollar una aplicación que defina una clase “Mascota”, que tenga un atributo privado “nombre”, un constructor de un parámetro que le asigne valor al atributo y el método “toString” heredado de la clase “Object” sobrescrito que devuelva la siguiente cadena: “Mascota{nombre=’nombreMascota’}”. Definir un método main que cree un arreglo de instancias de la clase “Mascota” y le asigne su respectivo valor al atributo “nombre” usando el constructor. Luego a través de una sentencia repetitiva recorrer el arreglo para mostrar la información de cada instancia.



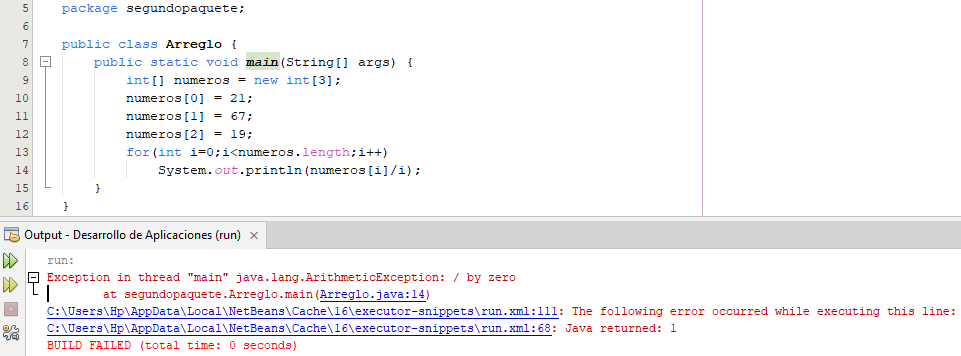
**Ejercicio 7**. Desarrollar una aplicación que cree una ArrayList de tipo String y que utilice sus diferentes métodos para agregar, modificar y eliminar elementos.



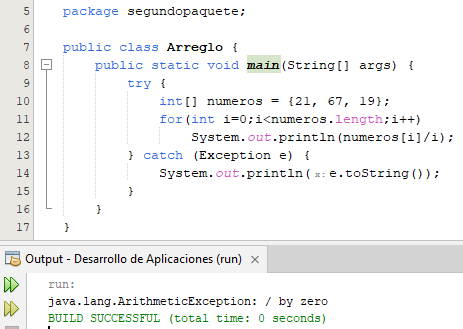
**Ejercicio 8**. Desarrollar una aplicación que genere una excepción por el uso de un índice fuera del límite de un arreglo.



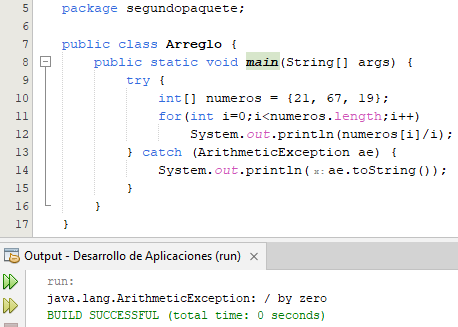
**Ejercicio 9**. Desarrollar una aplicación que genere una excepción por la división por cero entre dos números.



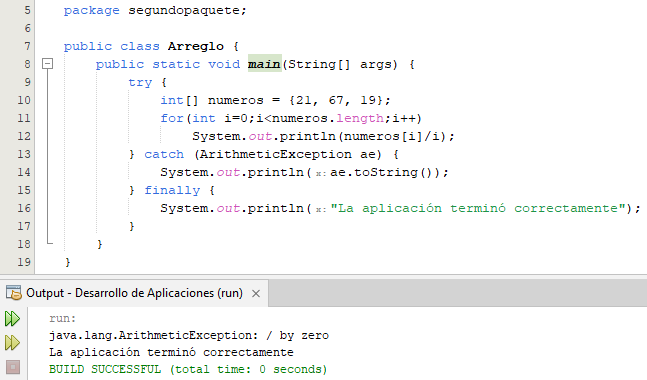
**Ejercicio 10**. Desarrollar una aplicación que genere una excepción por la división por cero entre dos números y que se capture y gestione la excepción a través de try … catch.



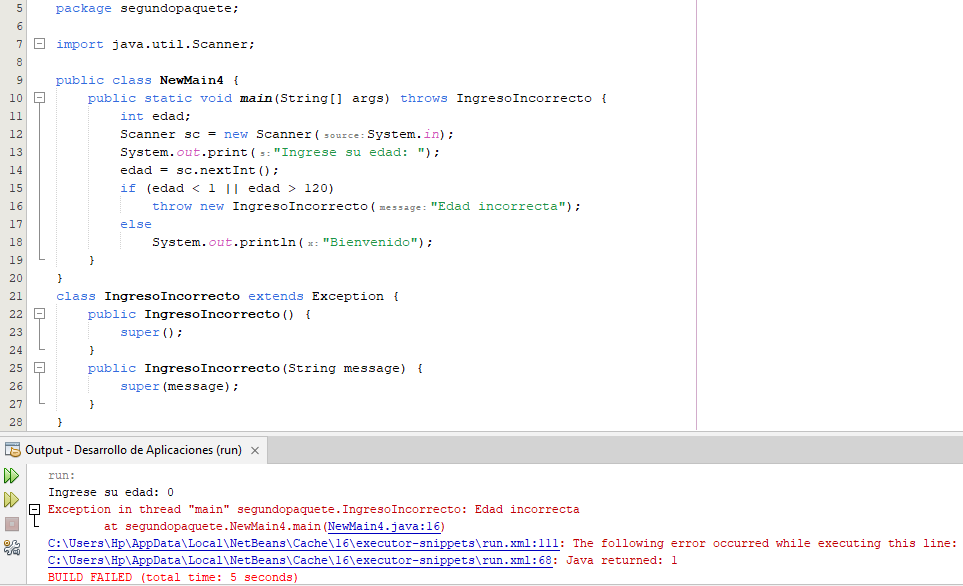
**Ejercicio 11**. Modificar el ejercicio anterior utilizando la clase “ArithmeticException”.



**Ejercicio 12**. Modificar el ejercicio anterior utilizando la sección “finally” que muestre un mensaje de que la aplicación terminó correctamente.

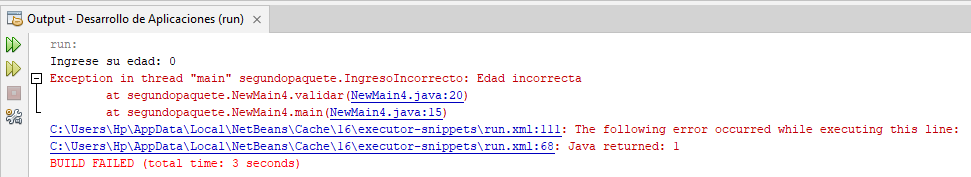


**Ejercicio 13**. Desarrollar una aplicación que declare una nueva excepción de nombre “IngresoIncorrecto”. Lanzar la excepción en un método “main” que solicite ingresar su edad y valide si no está en el rango de 1 a 120.

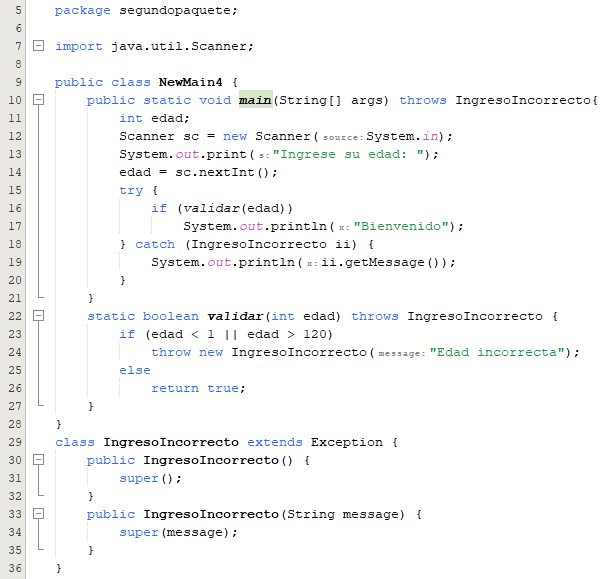


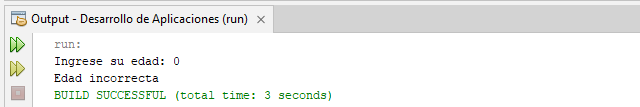
**Ejercicio 14**. Modificar la aplicación anterior, creando un método “validar” que debe ser llamado en el método “main” para validar la edad ingresada. En el método “validar” se debe lanzar la excepción “IngresoIncorrecto”.





**Ejercicio 15**. Modificar la aplicación anterior, para que en el método “main” se cature y gestione la excepción usando la sentencia “try … catch” y la aplicación termine correctamente.





**Ejercicio 16**. Desarrollar una aplicación que solicite el ingreso del nombre de 2 alumnos y la nota de cada uno de ellos como valor numérico. El resultado que debe mostrarse es el nombre de cada alumno, su nota y su calificacion como texto (Sobresaliente, Notable, Bien o A Mejorar).

Para ello crear un arreglo numérico y otro de tipo String (ambos unidimensionales). En el arreglo numérico se insertarán las notas ingresadas entre 0 y 20, pudiendo ser decimal. En el arreglo de Strings se insertarán los nombres de los alumnos.

Se debe controlar que se inserte en el arreglo notas válidas, para lo cual se debe usar el manejo de excepciones. Usar la excepción creada en los ejercicios anteriores “IngresoIncorrecto”.

Crear también un arreglo de String donde insertaremos el resultado de la nota con palabras admitiéndose estos valores: Sobresaliente, Notable, Bien o A Mejorar.

El programa debe comprobar la nota de cada alumno teniendo en cuenta esta equivalencia:

* + Si la nota está entre 0 y 12.99 será un A Mejorar.
  + Si la nota está entre 13 y 15.99 será un Bien.
  + Si la nota está entre 16 y 17.99 será un Notable.
  + Si la nota está entre 18 y 20 será un Sobresaliente.

Se debe mostrar en pantalla los resultados: el alumno su nota y su resultado en palabras utilizando bucles. Crea los métodos que creas convenientes.

**Ejercicio 17**. Desarrollar una aplicación que solicite el ingreso de su correo y defina una excepción propia llamada “Checked”. Crear una clase con el método “main” que invoque a un método adicional llamado “validarCorreo”. El método “main” debe controlar con “try-catch” la excepción lanzada desde el método “validarCorreo”. El método “validarCorreo” debe recibir como parámetro una cadena de texto y debe validar que sea el formato de un correo válido. La validación del correo debe tener las siguientes consideraciones:

* + Debe contener un caracter “@”.
  + Al lado izquierdo del carácter “@” solo debe permitir letras, números y guiones.
  + Al lado derecho del caracter “@” debe tener al menos un punto “.” Y después del último punto debe tener solo 2 o 3 caracteres.

**Ejercicio 18**. Desarrollar una aplicación que declare e inicialice un arreglo de arreglos numéricos con el siguiente contenido:

1 2 0 3 0 7

0 0 0 0

0 6 9 8 0 9 6

0 0 0 0 7 9 0 0

0 0 0 0 0

8 7 0 8 9 0 8

Se debe de eliminar todos los ceros que hay en los arreglos, y si un arreglo está lleno de puros ceros, hay que eliminar ese arreglo. Usar la clase “ArrayList”. El resultado debe ser el siguiente:

1 2 3 7

6 9 8 9 6

7 9

8 7 8 9 8

**Ejercicio 19**. Desarrollar una aplicación que declare una clase “PartidoFutbol” que tenga los siguientes atributos: equipoLocal, equipoVisitante, golesLocal, golesVisitante. La clase debe estar encapculada y debe tener sus métodos constructores, getters y setters. Desarrollar una clase con un método “main” donde cree un “ArrayList” del tipo “PartidoFutbol” y almacene los resultados de un campeonato, usando un método constructor. Una vez terminado el campeonato, se debe mostrar como resultado el equipo campeón.

Campeona el equipo que obtuvo mas puntos durante todo el campeonato. Los puntajes se asignan de la siguiente manera:

* Partido ganado equivale a 3 puntos.
* Partido empatado equivale a 1 punto.
* Partido perdido equivale a 0 puntos.

Esta validación se debe realizar a través de un método adicional que reciba como parámetro el ArrayList y verifique quien es el campeón. El método devolverá el equipo campeón.

Antes se debe verificar que todos los equipos que participan tengan la misma cantidad de partidos. Esta validación se debe hacer a través de un método adicional que reciba como parámetro el “ArrayList” y debe lanzar una excepción de no cumplir con lo indicado. La excepción puede ser propia, es decir creada por usted. La excepción debe ser capturada en el método “main” para que no se interrumpa la ejecución de la aplicación. Y al regresar al método “main” debe solicitar que se ingresen los partidos que faltan.

Si dos o mas equipos obtuvieran la misma cantidad de puntos se debe realizar partidos adicionales entre los involucrados y ver quien obtiene mas puntos para ser el campeón. En este repechaje se debe crear otro ArrayList de la misma clase “PartidoFutbol” y se debe usar los setters y getters para crear los partidos (OJO: son partidos de los equipos que han empatado en puntos, no son de nuevos equipos). Esto se debe realizar en otro método y debe devolver el equipo campeón.

**Ejercicio 20**. Desarrollar una aplicación que declare un arreglo de números sin ningún orden pre establecido y elaborar el código que ordene el arreglo sin usar métodos de ordenamiento propios de Java. Elaborar el ejercicio para arreglos de enteros, decimales y ArrayList. Gestionar las excepciones que se puedan presentar en sus pruebas, para que la aplicación siempre termine correctamente y usted sepa que error ha cometido en su código.