Nombre y apellidos:

1) El método computacional más común para calcular raíces cuadradas es el método de Newton de aproximaciones sucesivas. Cada vez que tenemos una estimación **y** del valor de la raíz cuadrada de un número **x**, podemos hacer una pequeña manipulación para obtener una mejor aproximación promediando **y** con **x/y.** (3pto)

   Por ejemplo, calculemos la raíz cuadrada de 2 usando la aproximación inicial $\sqrt{2} \approx 1$:

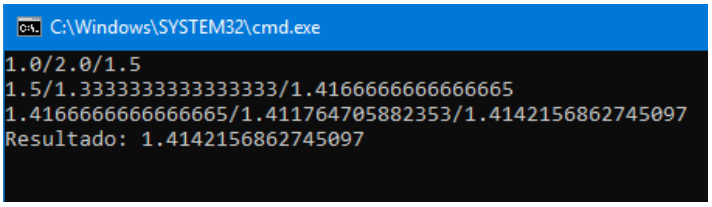
| Estimación $y$ | Cuociente $x/y$ | Promedio |
|---|---|---|
| $1$ | $2/1 = 2$ | $(2+1)/2 = 1.5$ |
| $1.5$ | $2/1.5 = 1.3333$ | $(1.3333+1.5)/2 = 1.4167$ |
| $1.4167$ | $2/1.4167 = 1.4118$ | $(1.4118+1.4167)/2 = 1.4142$ |
| $1.4142$ | ... | ... |

   Al continuar este proceso, obtenemos cada vez mejores estimaciones de la raíz cuadrada.

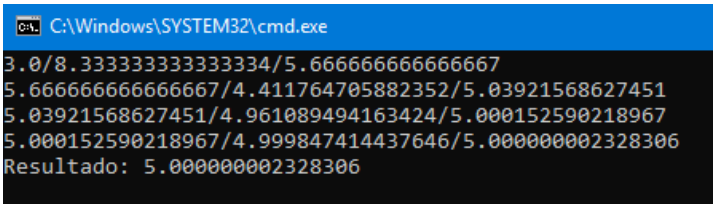
   El algoritmo debe detenerse cuando la estimación es "suficientemente buena", que debe ser un criterio bien definido.

   Escriba un programa que reciba como entrada un número real **x** y un número entero **aprox** (número del cual queremos calcular la raíz cuadrada y su aproximación inicial) y calcule su raíz cuadrada usando el método de Newton. El algoritmo debe detenerse cuando **aprox²** difiera de **x** en menos de 0,0001.

   El programa mostrará por pantalla los resultados de cada paso y la aproximación final:




   Ejemplo de raíz de 25 con aproximación de 3:




2) Hacer un programa que cree un vector de n elementos y lo rellene con números aleatorios del 1 al 500. Estos números podrán ser positivos o negativos (el 0 no está incluido). Una vez lleno, mostrarlo por pantalla. Mostrar por pantalla, separado por comas, todos los números primos entre 3 y 400, incluidos, del vector. El número n lo pedirá por pantalla. Hay que controlar que el número introducido sea mayor de 50, si no lo seguirá pidiendo. (Los repetidos salen. El orden da igual)

   Un número primo es un número natural mayor que 1 que tiene únicamente dos divisores positivos distintos: él mismo y el 1. Solo se puede dividir por uno y el mismo (5ptos)

Ejemplo de ejecución:
Introduce un número entero: 20
Introduce un número entero: 59
-34 3 77 -98 31, 8, 401 …

Resultado: 3, 31

3) Dado el siguiente código indica si compila o no. En caso de que creas que sí, indica cual sería la salida por pantalla. En caso de que no, razona tu respuesta. Se valorará la concreción. (2pto)

```
import java.util.Scanner;
public class Prueba {

	public static void main (String[] args) {
		Scanner in=new Scanner(System.in);
		int [] vec={1,2,3};
		int suma=0;
		for(int i=0;i<vec.length;i++)
			suma=suma+vec[i];
			System.out.println("La suma es: "+suma);

	}
}
```

**Se valorará la claridad, estructuras usadas, así como ceñirse a lo que se pide.**

# API

- **Scanner**
  - Instanciación: Scanner sc=new Scanner(System.in)
  - Uso:
    - nextInt()----->enteros;
    - nextDouble->decimales
    - nextLine()--->String
- **Random**
  - Instanciación: Random rnd=new Random();
  - Uso:
    - nextInt(a): genera números aleatorios entre 0 y a-1
- **String**
  - lentgth(): devuelve la longitus del String
  - charAt(i): devuelve el carácter en la posición i
  - String.valueOf(x): devuelve el número x como String
- **Integer.parseInt(s):** devuelve el String s como int

**Algoritmo de la burbuja**

```
for(int i = 1; i < vec.length; i++) {
	for(int j = 0; j < (vec.length - i); j++) {
		if(vec[j] > vec[j+1]){
			aux = vec[j];
			vec[j] = vec[j+1];
			vec[j+1] = aux;
```

```
            }
        }
}
```

**Math**

`int abs(int a), double abs(double a)`

Returns the absolute value of an `int` value. If the argument is not negative, the argument is returned. If the argument is negative, the negation of the argument is returned.

Note that if the argument is equal to the value of `Integer.MIN_VALUE`, the most negative representable `int` value, the result is that same value, which is negative.

**Parameters:**

`a - the argument whose absolute value is to be determined`

**Returns:**

`the absolute value of the argument.`

`double sqrt(double a)`

Returns the correctly rounded positive square root of a `double` value. Special cases:

- If the argument is NaN or less than zero, then the result is NaN.
- If the argument is positive infinity, then the result is positive infinity.
- If the argument is positive zero or negative zero, then the result is the same as the argument.

Otherwise, the result is the `double` value closest to the true mathematical square root of the argument value.

**Parameters:**

`a - a value.`

**Returns:**

`the positive square root of a. If the argument is NaN or less than zero, the result is NaN.`

`double pow(double a, double b)`

Returns the value of the first argument raised to the power of the second argument.

**Parameters:**

`a - the base.`

`b - the exponent.`

**Returns:**

`the value` $a^b$.