|  |  |  |
| --- | --- | --- |
| **Texto  Descripción generada automáticamente con confianza media** | **TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS**  **FACULTAD DE INGENIERÍA**  **Universidad Nacional de Jujuy** |  |

Trabajo Práctico

N°1

Ortega Josué Daniel – LU / 421

*Profesores:*

*Mg. Ing. Ariel Alejandro Vega*

*Año 2024*

**FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS**

INDICE

[**Sección Expresiones aritméticas y lógicas 3**](#_Toc164201239)

[**Ejercicio 1 3**](#_Toc164201240)

[**Ejercicio 2 3**](#_Toc164201241)

[**Ejercicio 4 4**](#_Toc164201242)

[**Ejercicio 5 5**](#_Toc164201243)

[**Ejercicio 6 6**](#_Toc164201244)

[**Ejercicio 7 7**](#_Toc164201245)

[**Ejercicio 8 7**](#_Toc164201246)

[**Ejercicio 9 8**](#_Toc164201247)

[**Ejercicio 10 8**](#_Toc164201248)

[**Ejercicio 11 8**](#_Toc164201249)

[**Sección Análisis – Diseño y Codificación de algoritmos – Aplicación de estructuras de control 9**](#_Toc164201250)

[**Ejercicio 12 9**](#_Toc164201251)

[**Ejercicio 13 10**](#_Toc164201252)

[**Ejercicio 14 11**](#_Toc164201253)

[**Ejercicio 15 12**](#_Toc164201254)

[**Ejercicio 16 14**](#_Toc164201255)

[**Ejercicio 17 15**](#_Toc164201256)

[**Ejercicio 18 18**](#_Toc164201257)

[**Ejercicio 19 20**](#_Toc164201260)

[**Ejercicio 20 22**](#_Toc164201261)

[**Ejercicio 21 24**](#_Toc164201262)

[**Ejercicio 22 26**](#_Toc164201263)

# Sección Expresiones aritméticas y lógicas

## Ejercicio 1

**Evaluar (obtener resultado) la siguiente expresión para A = 2 y B = 5**

**3\* A - 4 \* B / A ^ 2**

**(3\*A)-(4\*B/(A^2))**

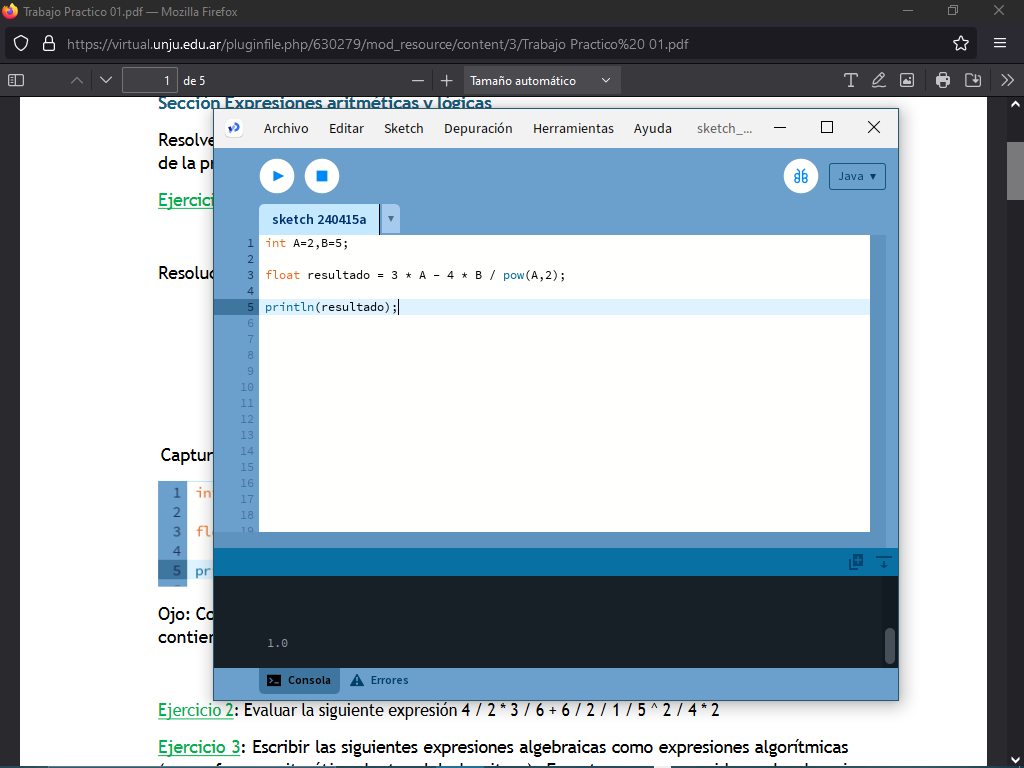
**(3\*2)-(4\*5/(2^2))**

**(3\*2)-(4\*5/4)**

**6-(20/4)**

**6-5**

**1**

****

## Ejercicio 2

**Evaluar la siguiente expresión:**

**4 / 2 \* 3 / 6 + 6 / 2 / 1 / 5 ^ 2 / 4 \* 2**

**4 / 2 \* 3 / 6 + 6 / 2 / 1 / (5 ^ 2) / 4 \* 2**

**(4 / 2) \* 3 / 6 + (6 / 2) / 1 / 25 / 4 \* 2**

**(2 \* 3) / 6 + (3 / 1) / 25 / 4 \* 2**

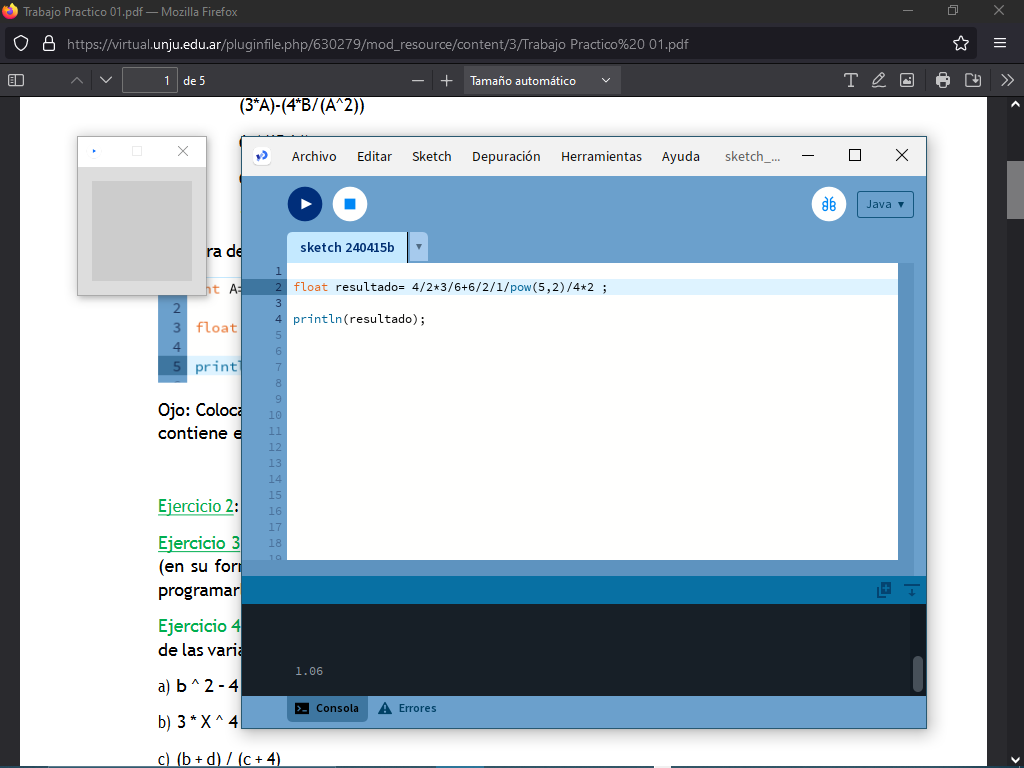
**(6 / 6) + (3 / 25) / 4 \* 2**

**1 + (0,12 / 4) \* 2**

**1 + (0,03 \* 2)**

**1 + 0,06**

**1,06**

****

## Ejercicio 4

**Evaluar las siguientes expresiones aritméticas, para lo cual indicar en el caso de las variables, el valor indicado. Luego escribirlas como expresiones algebraicas.**

**a) b ^ 2 – 4 \* a \* c a= 2, b=1, c=4, d=5, x=3, y=4**

**(1 ^ 2) – (4 \* 2) \* 4**

**1 – (8 \* 4)**

**1-32**

**-31**

**Expresión algebraica:**

**b) 3 \* X ^ 4 – 5 \* X ^ 3 + X 12 – 17**

**3 \* (3 ^ 4) – 5 \* (3 ^ 3) + 12 \* 3 – 17**

**(3 \* 81) – (5 \* 27) + (12 \* 3) – 17**

**(243 – 135) + 36 – 17**

**(108 + 36) - 17**

**144 – 17**

**127**

**Expresión algebraica:**

**c) (b + d) / (c + 4)**

**(1 + 5) / (4 + 4)**

**6 / 8**

**0,75**

**Expresión algebraica:**

**d) (x ^ 2 + y ^ 2) ^ (1 / 2)**

**((3 ^ 2) + (4 ^ 2)) ^ (1 / 2)**

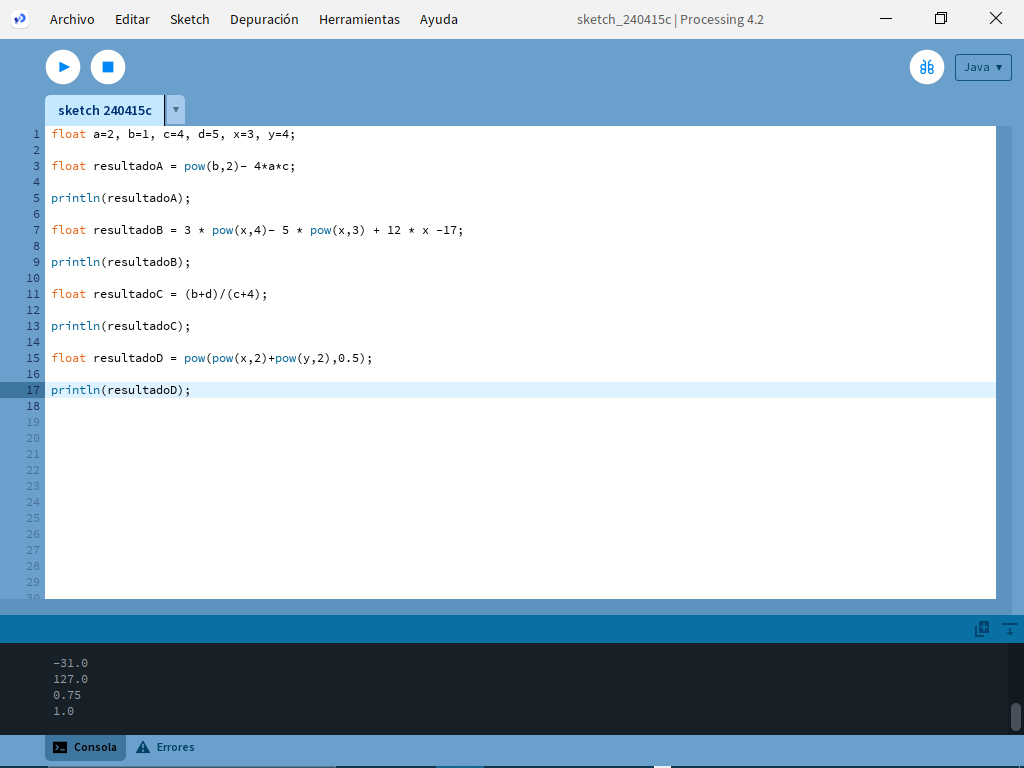
**(9 + 16) ^ (1/2)**

**25 ^ (1/2)**

**25 ^ 0,5**

**5**

**Expresión algebraica:**

****

## Ejercicio 5

**Si el valor de A es 4, el valor de B es 5 y el valor de C es 1, evaluar las siguientes expresiones:**

**a) B \* A – B ^ 2 / 4 \* C**

**5 \* 4 – (5 ^ 2) / 4 \* 1**

**(5 \* 4) – (25 / 4) \* 1**

**20 – (6,25 \* 1)**

**20 – 6,25**

**13,75**

**b) (A \* B) / 3 ^ 2**

**(4 \* 5) / 3 ^ 2**

**20 / 9**

**2,2222**

**c) (((B + C) / 2 \* A + 10) \* 3 \* B) – 6**

**(((5 + 1) / 2 \* 4 + 10) \* 3 \* 5) – 6**

**((6 / 2 \* 4 + 10) \* 3 \* 5) – 6**

**((3 \* 4 + 10) \* 3 \* 5) – 6**

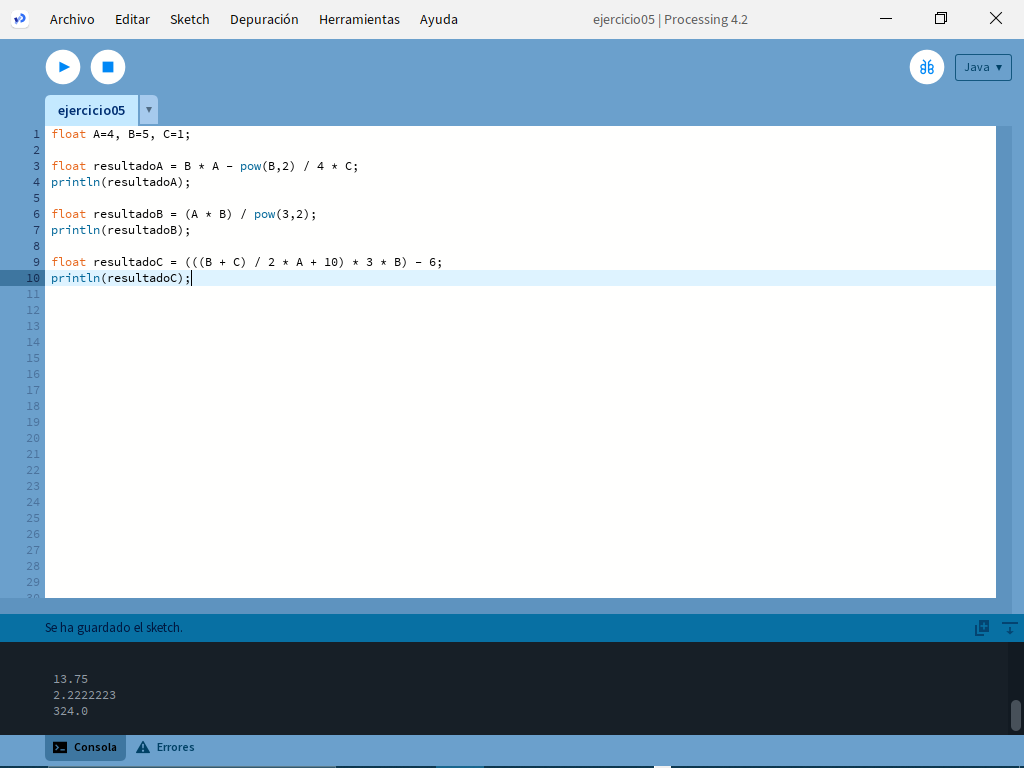
**((12 + 10) \* 3 \* 5) – 6**

**(22 \* 3 \* 5) – 6**

**(66 \* 5) – 6**

**330 - 6**

**324**

****

## Ejercicio 6

**Para x=3, y=4; z=1, evaluar el resultado de**

**R1 = y + z**

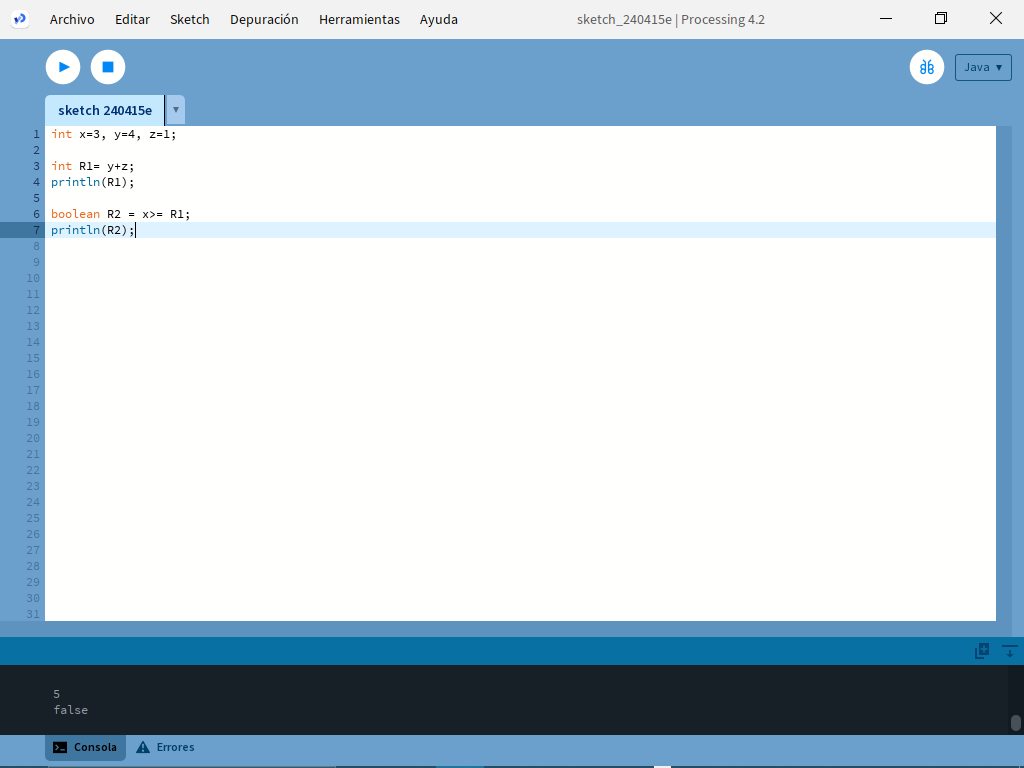
**R2 = x >= R1**

**R1 = 4 + 1**

**R1= 5**

**R2 = 3 >= 5**

**R2 = False**

****

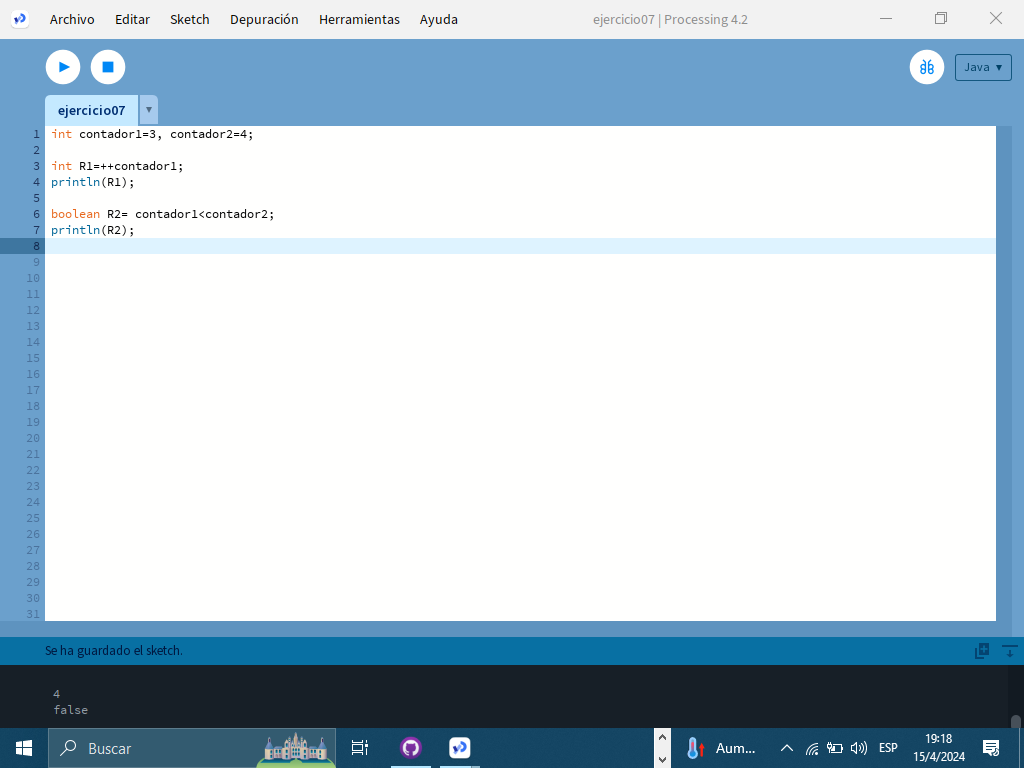
## Ejercicio 7

**Para contador1=3, contador2=4, evaluar el resultado de**

**R1 = ++contador1**

**R2 = contador1 < contador2**

**R1=++3  
R1= 4  
R2= 4 < 4  
R2= False**

****

## Ejercicio 8

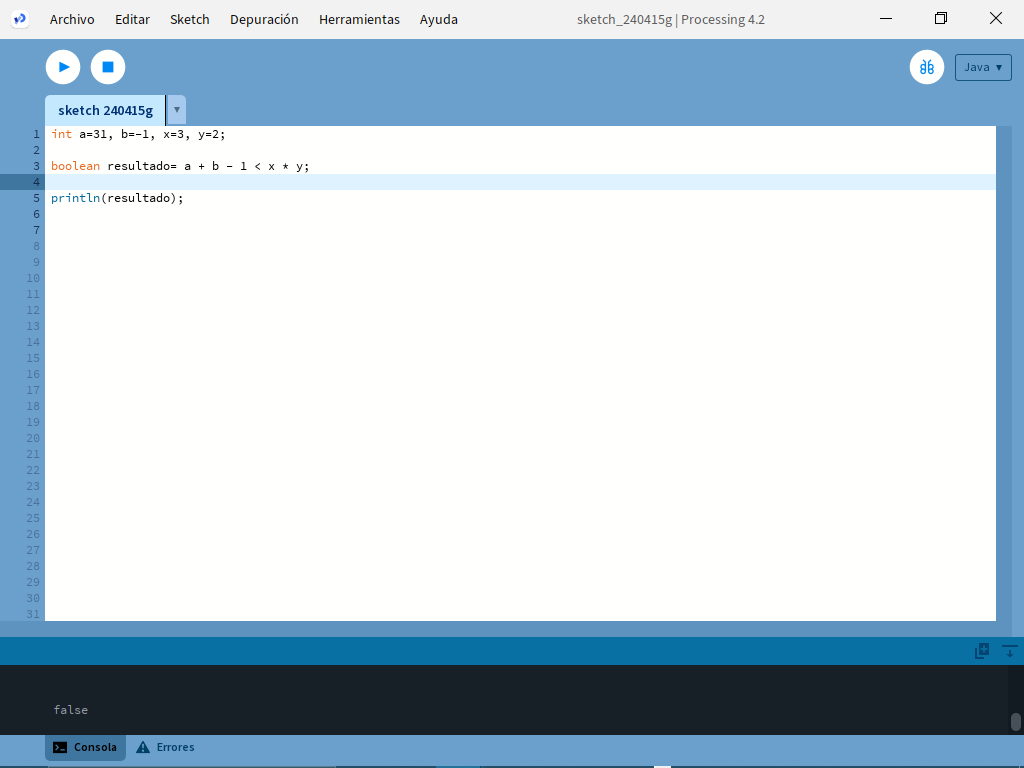
**Para a=31, b=-1; x=3, y=2, evaluar el resultado de**

**a + b - 1 < x \* y**

**31 + (-1) – 1 < 3 \* 2**

**29 < 6**

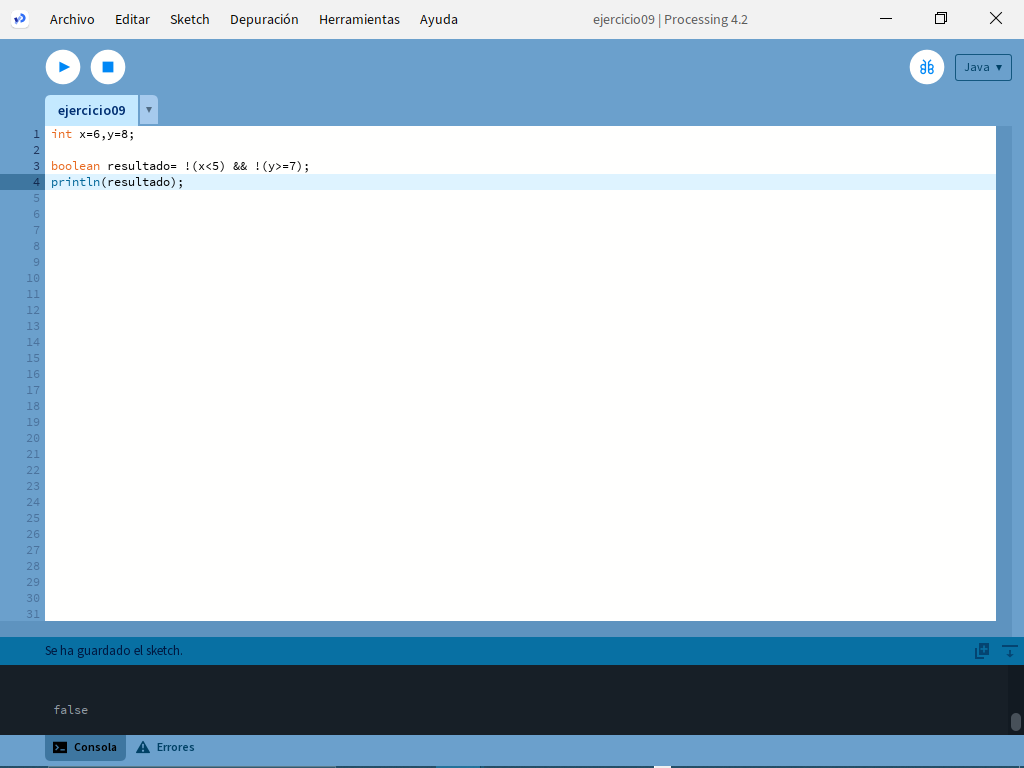
**False**

****

## Ejercicio 9

**Para x=6, y=8, evaluar el resultado de**

**!(x<5) && !(y>=7)  
!(6<5) && !(8 >=7)  
!(false) && !(true)  
true && false  
false**

****

## Ejercicio 10

**Para i=22, j=3, evaluar el resultado de**

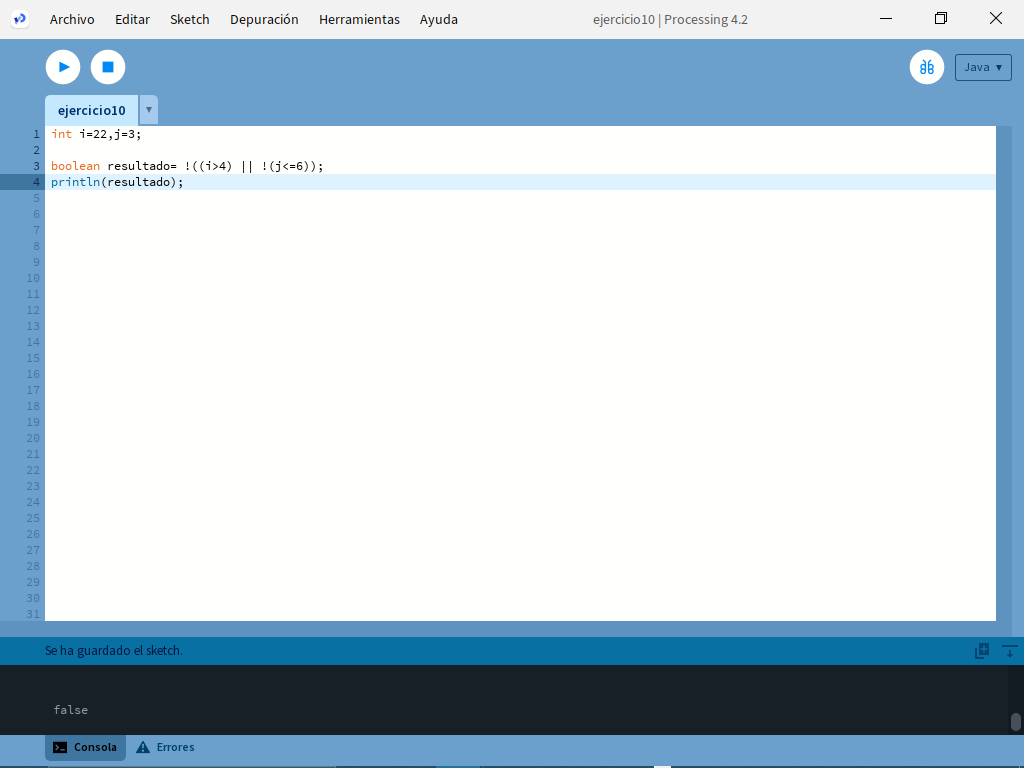
**!((i>4) || !(j<=6))**

**!((22>4) || !(3<=6))**

**!(true) || !(true)**

**false || false**

**false**

****

## Ejercicio 11

**Para a=34, b=12,c=8, evaluar el resultado de:**

**!(a+b==c) || (c!=0)&&(b-c>=19)**

**!(34+12==8) || (8!=0)&&(12-8>=19)**

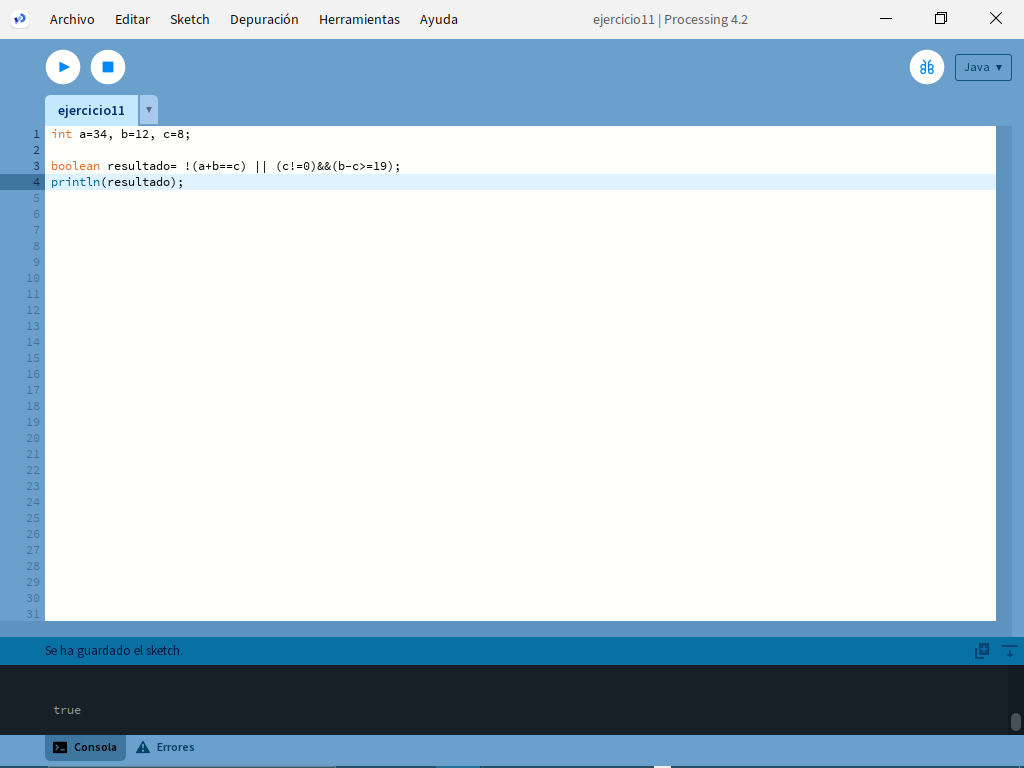
**!(46==8) || true && (4>=19)**

**!false || true && false**

**True || true && false**

**True || false**

**true**

****

# Sección Análisis – Diseño y Codificación de algoritmos – Aplicación de estructuras de control

## Ejercicio 12

**Un problema sencillo. Deberá pedir por teclado al usuario un nombre y posteriormente realizará la presentación en pantalla de un saludo con el nombre indicado.**

**Definición del Problema: Pedir al usuario un nombre y mostrarlo por pantalla en un saludo.**

**Análisis:**

**• Datos de Entrada:**

**nombre: String**

**• Datos de Salida:**

**mensaje: String**

**• Proceso:**

**¿Quién debe realizar el proceso?: Usuario**

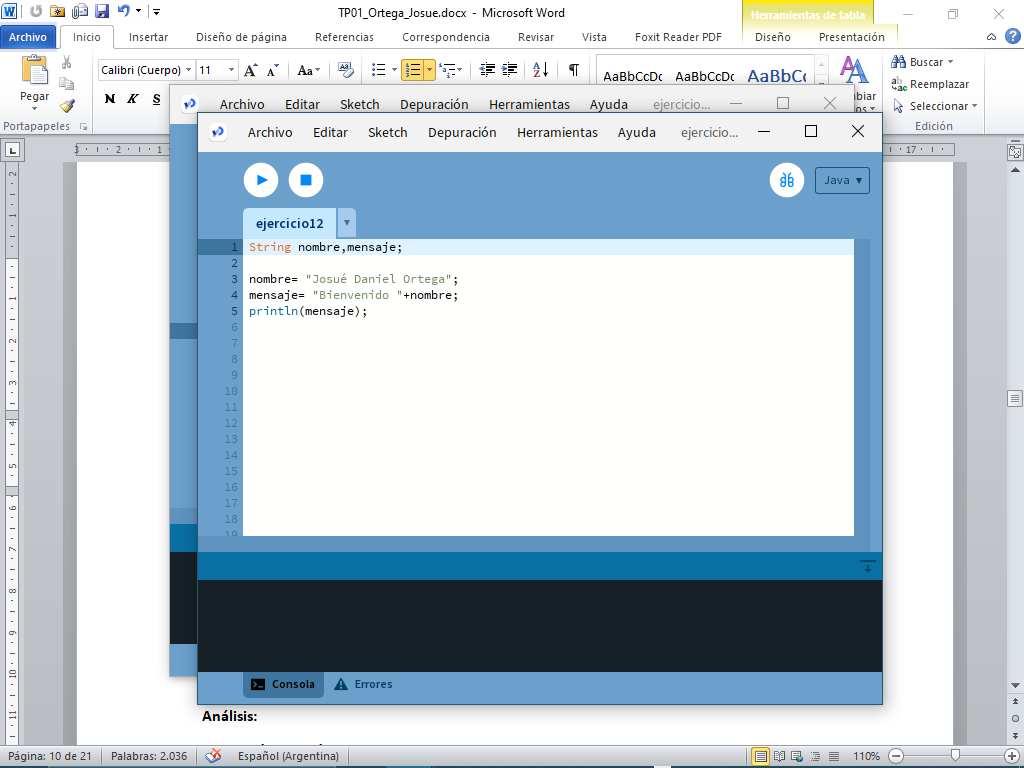
**¿Cuál es el proceso que realiza ?**

**mensaje = “Bienvenido ” + nombre**

**Diseño**

|  |
| --- |
| **Entidad que resuelve el problema: USUARIO** |
| **Variables:**   * **nombre: String//almacena el nombre ingresado** * **mensaje: String// almacena el mensaje con el nombre** |

|  |
| --- |
| **Nombre del Algoritmo: mostrar\_mensaje\_de\_bienvenida** |
| **Proceso del algoritmo:**   1. **Leer nombre** 2. **mensaje🡨”Bienvenido “+ nombre** 3. **Mostrar mensaje** |

****

## Ejercicio 13

**Será común resolver problemas utilizando variables. Calcule el perímetro y área de un rectángulo dada su base y su altura.**

**Definición del Problema: Calcular el perímetro y área de un rectángulo con su base y altura**

**Análisis:**

**• Datos de Entrada:**

**base: float**

**altura: float**

**• Datos de Salida:**

**area: float**

**perímetro: float**

**• Proceso:**

**¿Quién debe realizar el proceso?: Usuario**

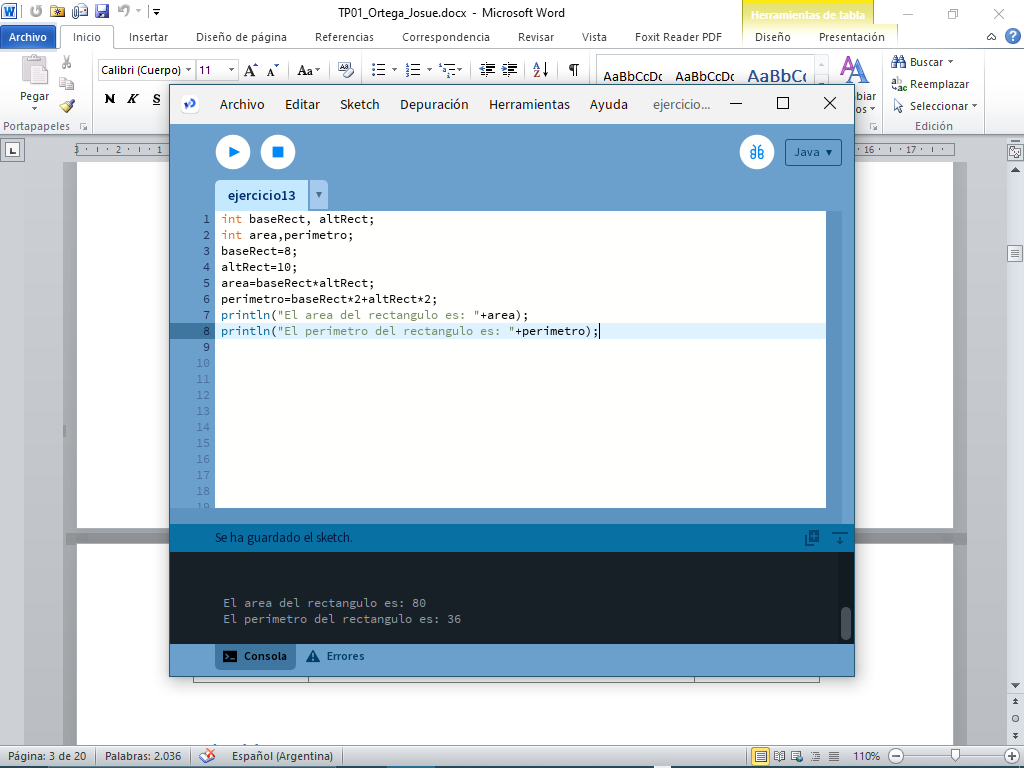
**¿Cuál es el proceso que realiza ?**

**Área= base x altura Perimetro: base\*2+altura\*2**

**Diseño**

|  |
| --- |
| **Entidad que resuelve el problema: USUARIO** |
| **Variables:**   * **base,altura : float// números ingresados por el usuario** * **perímetro,área: float// resultado de las operaciones** |

|  |
| --- |
| **Nombre del Algoritmo: calcular\_perimetro\_y\_area** |
| **Proceso del algoritmo:**   1. **Leer base** 2. **Leer altura** 3. **perímetro <- base\*2+altura\*2** 4. **área <- base\*altura** 5. **Mostrar perimetro** 6. **Mostrar area** |

****

## Ejercicio 14

**Una ayuda importante al momento de resolver problemas con algoritmos es asumir que su gran amigo son las matemáticas. Obtenga la hipotenusa de un triángulo rectángulo conociendo sus catetos.**

**Definición del Problema: Calcular hipotenusa de un triángulo rectángulo.**

**Análisis:**

**• Datos de Entrada:**

**catetoA: float**

**catetoB: float**

**• Datos de Salida:**

**hipotenusa: float**

**• Proceso:**

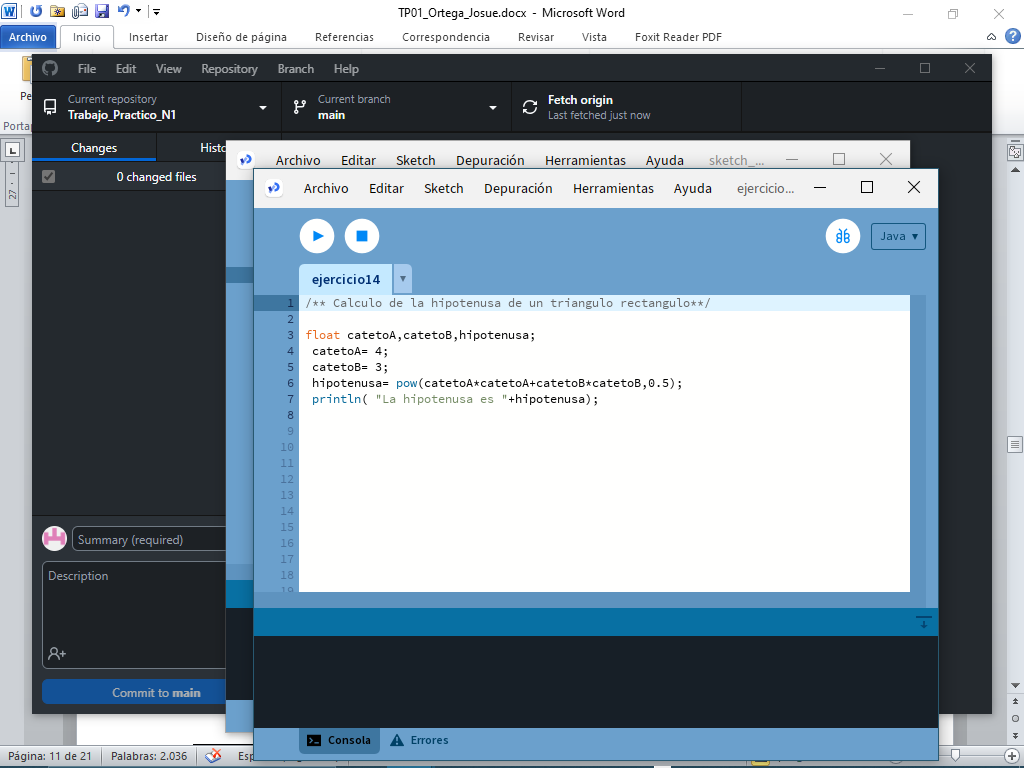
**¿Quién debe realizar el proceso?: Usuario**

**¿Cuál es el proceso que realiza ?**

**hipotunusa=**

|  |
| --- |
| **Entidad que resuelve el problema: USUARIO** |
| **Variables:**   * **catetoA,catetoB : float// números ingresados por el usuario** * **hipotenusa: float// hipotenusa del triangulo rectangulo** |

|  |
| --- |
| **Nombre del Algoritmo: calcular\_hipotenusa** |
| **Proceso del algoritmo:**   1. **Leer catetoA** 2. **Leer catetoB** 3. **hipotenusa <- ( catetoA\*catetoA+catetoB\*catetoB)\*\*(1/2)** 4. **Mostrar hipotenusa** |

****

## Ejercicio 15

**Si viste algo de los apuntes y vídeos, esto debería ser muy fácil de resolver.**

**Dados dos números permita calcular la suma, resta, multiplicación y división de estos.**

**Considere que cada una de estas operaciones es un algoritmo cuando realice el diseño.**

**Obviamente muestre los resultados.**

**Definición del Problema: Calcular operación suma, resta, multiplicación y división.**

**Análisis:**

**• Datos de Entrada:**

**a: float**

**b: float**

**• Datos de Salida:**

**suma: float**

**resta: float**

**producto: float**

**división: float**

**• Proceso:**

**¿Quién debe realizar el proceso?: Usuario**

**¿Cuál es el proceso que realiza ?**

**suma= a+b, resta=a-b, multiplicación=a\*b, división=a/b**

**Diseño**

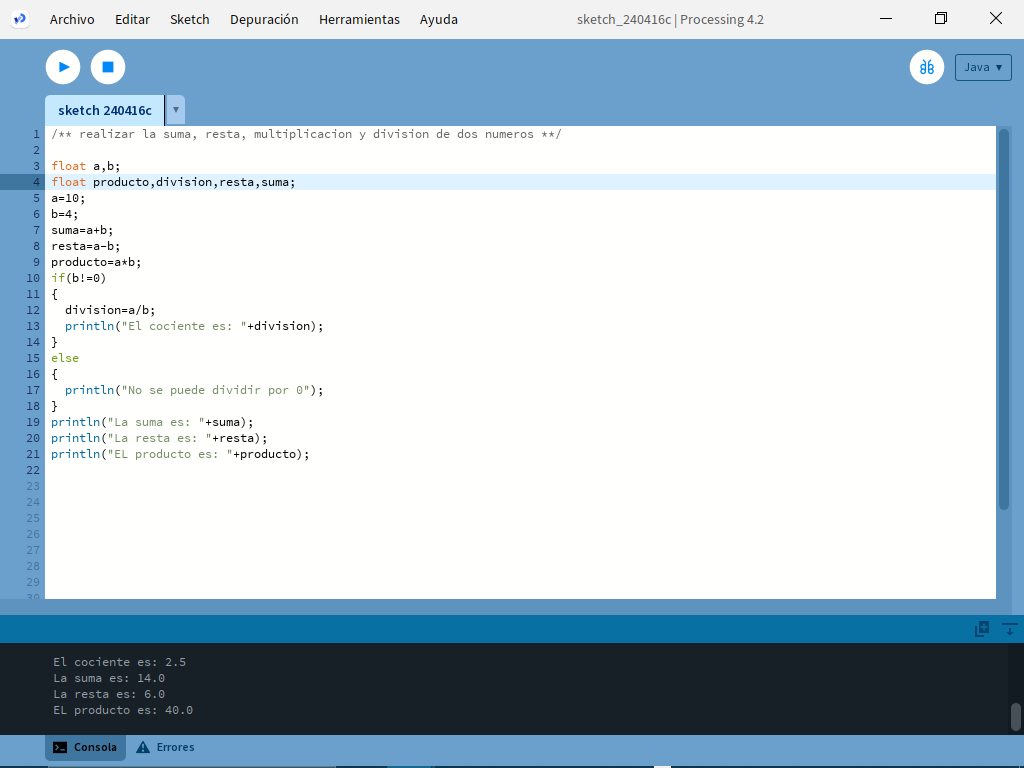
|  |
| --- |
| **Entidad que resuelve el problema: Usuario** |
| **Variables:**   * **a,b : float// números ingresados por el usuario para realizar operaciones** * **suma,resta,producto,division: float// guardan el resultado de las operaciones** |

|  |
| --- |
| **Nombre del Algoritmo: sumar\_numeros** |
| **Proceso del algoritmo:**   1. **Leer a** 2. **Leer b** 3. **suma <- a+b** 4. **Mostrar suma** |

|  |
| --- |
| **Nombre del Algoritmo: restar\_numeros** |
| **Proceso del algoritmo:**   1. **Leer a** 2. **Leer b** 3. **resta <- a-b** 4. **Mostrar resta** |

|  |
| --- |
| **Nombre del Algoritmo: sumar\_numeros** |
| **Proceso del algoritmo:**   1. **Leer a** 2. **Leer b** 3. **producto <- a\*b** 4. **Mostrar producto** |

|  |
| --- |
| **Nombre del Algoritmo: dividir\_numeros** |
| **Proceso del algoritmo:**   1. **Leer a** 2. **Leer b** 3. **Si (b != 0) entonces** 4. **division <- a/b** 5. **Mostrar división** 6. **Si\_no** 7. **Mostrar “No se puede dividir por 0”** 8. **Fin\_SI** |

****

## Ejercicio 16

**Necesitamos convertir una temperatura Fahrenheit en grados Celsius. Si no conoce la forma en la que se realiza esta conversión, debería investigarlo; para eso sirve la etapa de análisis. Pero como somos buenos, daremos una ayuda.**

**Definición del Problema: Transformar Fahrenheit a Celsius.**

**Análisis:**

**• Datos de Entrada:**

**tempFahrenheit: float**

**• Datos de Salida:**

**tempCelsius: float**

**• Proceso:**

**¿Quién debe realizar el proceso?: Usuario**

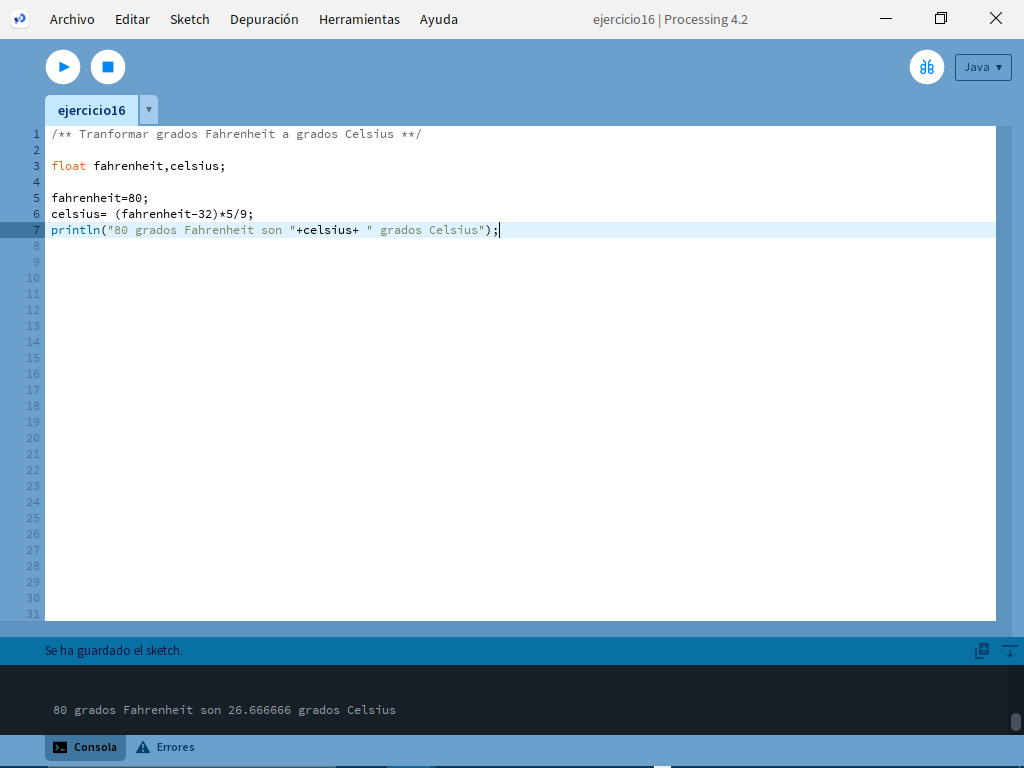
**¿Cuál es el proceso que realiza ?**

**tempCelsius=(tempFahrenheit-32)\*5/9**

**Diseño**

|  |
| --- |
| **Entidad que resuelve el problema: Usuario** |
| **Variables:**   * **tempFahrenheit: float// temperatura en grados F** * **tempCelsius: float// valor de salida en grados Celsius** |

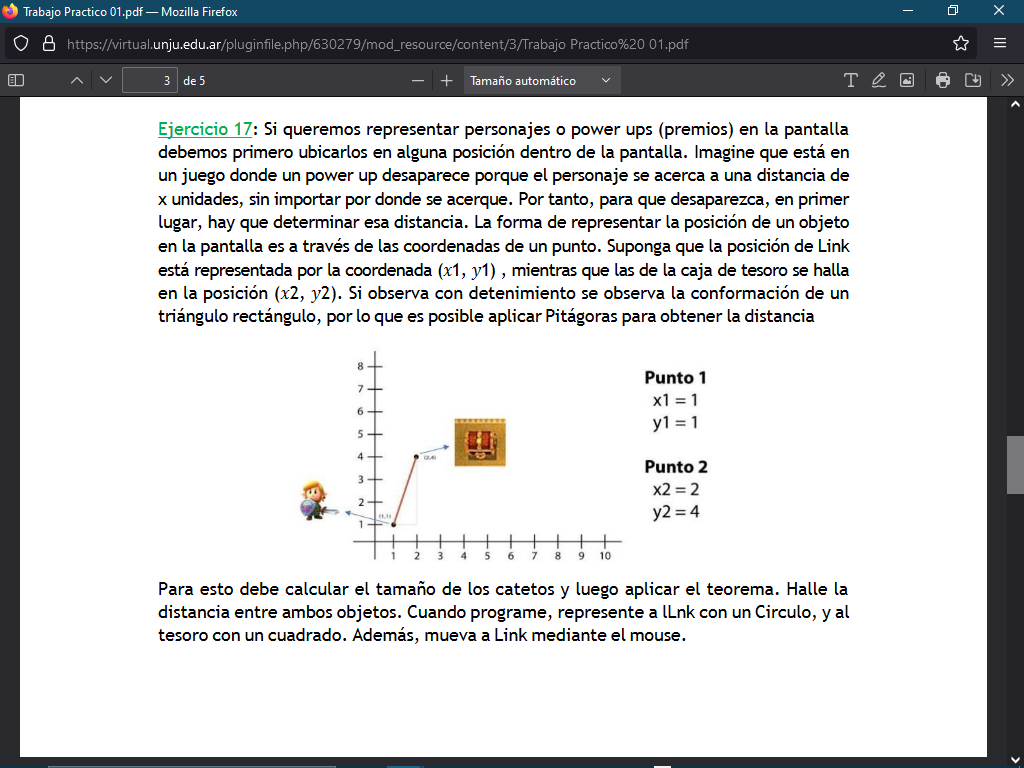
|  |
| --- |
| **Nombre del Algoritmo: tranformar\_grados\_Fahrenheit** |
| **Proceso del algoritmo:**   1. **Leer tempFahrenheit** 2. **tempCelsius <-( tempFahrenheit-32)\*5/9** 3. **Mostrar tempCelsius** |

****

## Ejercicio 17

**Si queremos representar personajes o power ups (premios) en la pantalla debemos primero ubicarlos en alguna posición dentro de la pantalla. Imagine que está en un juego donde un power up desaparece porque el personaje se acerca a una distancia de x unidades, sin importar por donde se acerque. Por tanto, para que desaparezca, en primer lugar, hay que determinar esa distancia. La forma de representar la posición de un objeto en la pantalla es a través de las coordenadas de un punto. Suponga que la posición de Link está representada por la coordenada (𝑥1, 𝑦1) , mientras que las de la caja de tesoro se halla en la posición (𝑥2, 𝑦2). Si observa con detenimiento se observa la conformación de un triángulo rectángulo, por lo que es posible aplicar Pitágoras para obtener la distancia**

**Para esto debe calcular el tamaño de los catetos y luego aplicar el teorema. Halle la distancia entre ambos objetos. Cuando programe, represente a Link con un Círculo, y al tesoro con un cuadrado. Además, mueva a Link mediante el mouse.**

****

**Definición del Problema: calcular la distancia entre Link y el tesoro**

**Análisis:**

**• Datos de Entrada:**

**x1,y1:float**

**x2,y2:float**

**• Datos de Salida:**

**catA:float**

**catB:float**

**Distancia:float**

**• Proceso:**

**¿Quién debe realizar el proceso?: Link**

**¿Cuál es el proceso que realiza ?**

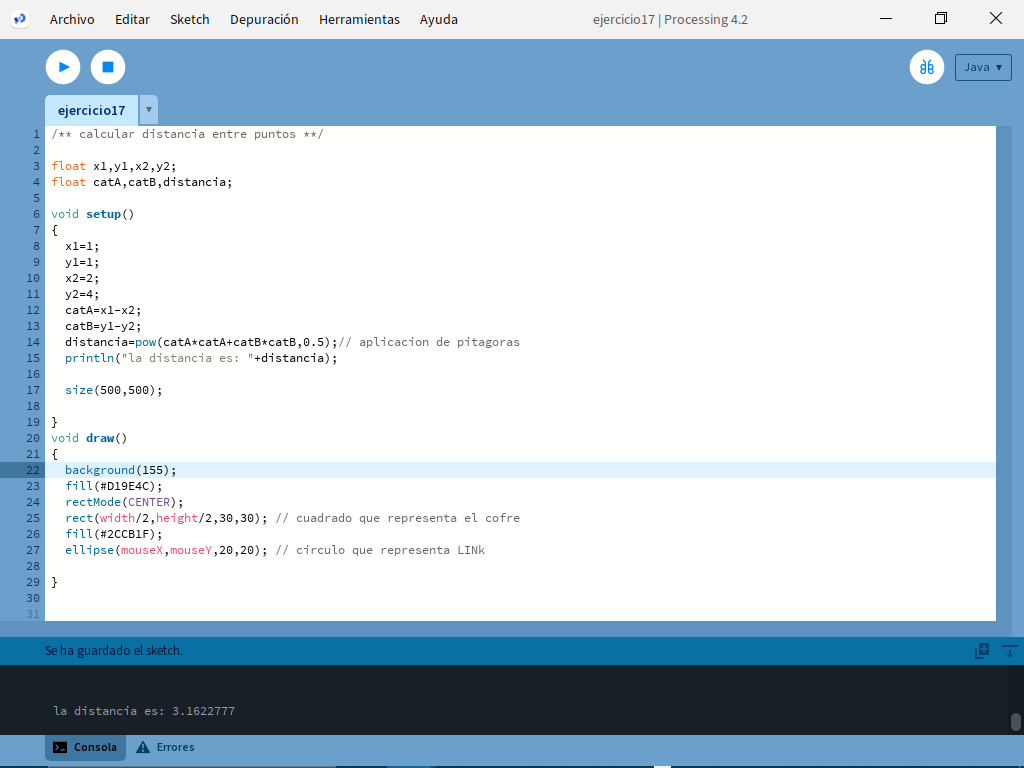
**Aplicación del teorema de Pitágoras**

**Hipotenusa=**

**Diseño**

|  |
| --- |
| **Entidad que resuelve el problema: Link** |
| **Variables:**   * **x1,y1,x2,y2: float// coordenadas de Link y el cofre** * **catA,catB: float// resultado de la resta de las coordenadas** * **distancia: float// distancia entre Link y el cofre** |

|  |
| --- |
| **Nombre del Algoritmo: calcular\_distancia\_entre\_puntos** |
| **Proceso del algoritmo:**   1. **x1<- 1** 2. **y1<- 1** 3. **x2<- 2** 4. **y2<- 4** 5. **catA<- x1 - x2** 6. **catB<- y1 – y2** 7. **distancia<- (catA\*catA+catB\*catB)\*\*(1/2)** 8. **Mostrar distancia** |

****

## Ejercicio 18

**Desarrolle el análisis y diseño de un algoritmo que permita obtener las raíces de una ecuación de segundo grado. Además, utilice la estructura según para el análisis de la discriminante de la ecuación cuadrática. Obviamente codifique en Processing.**

**Definición del Problema: Calcular las raíces de una ecuación de segundo grado**

**Análisis:**

**• Datos de Entrada:**

**a: float**

**b: float**

**c: float**

**• Datos de Salida:**

**x1: float**

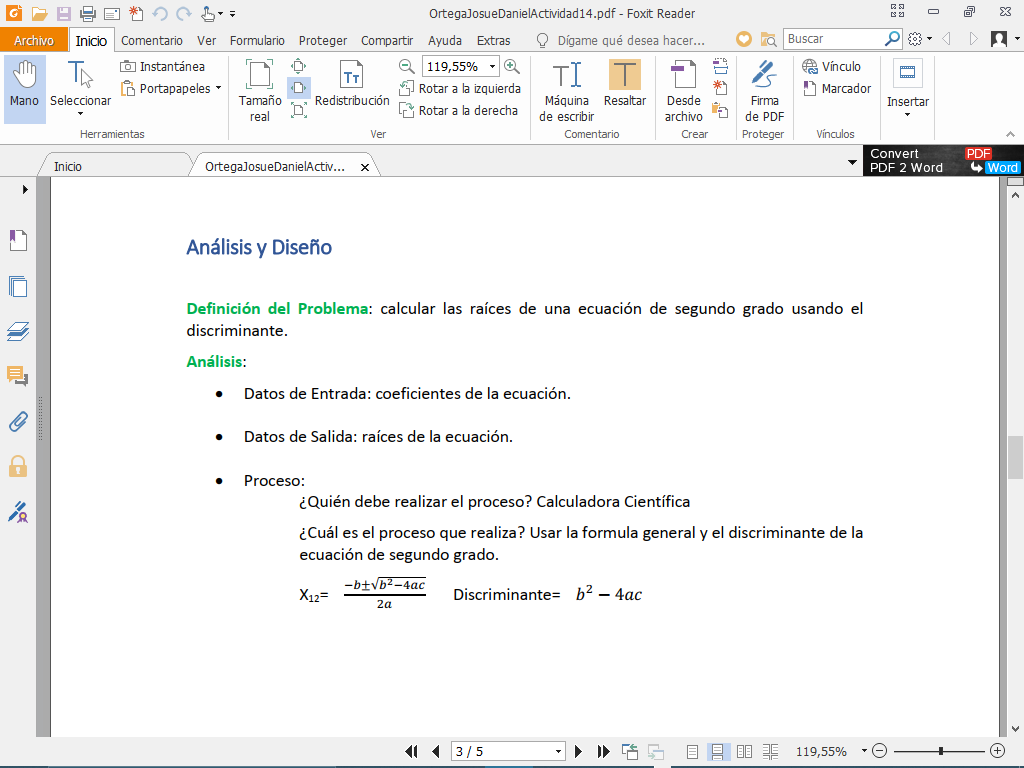
**x2: float**

**x: float**

**• Proceso:**

**¿Quién debe realizar el proceso?: Calculadora cientifica**

**¿Cuál es el proceso que realiza ?**

****

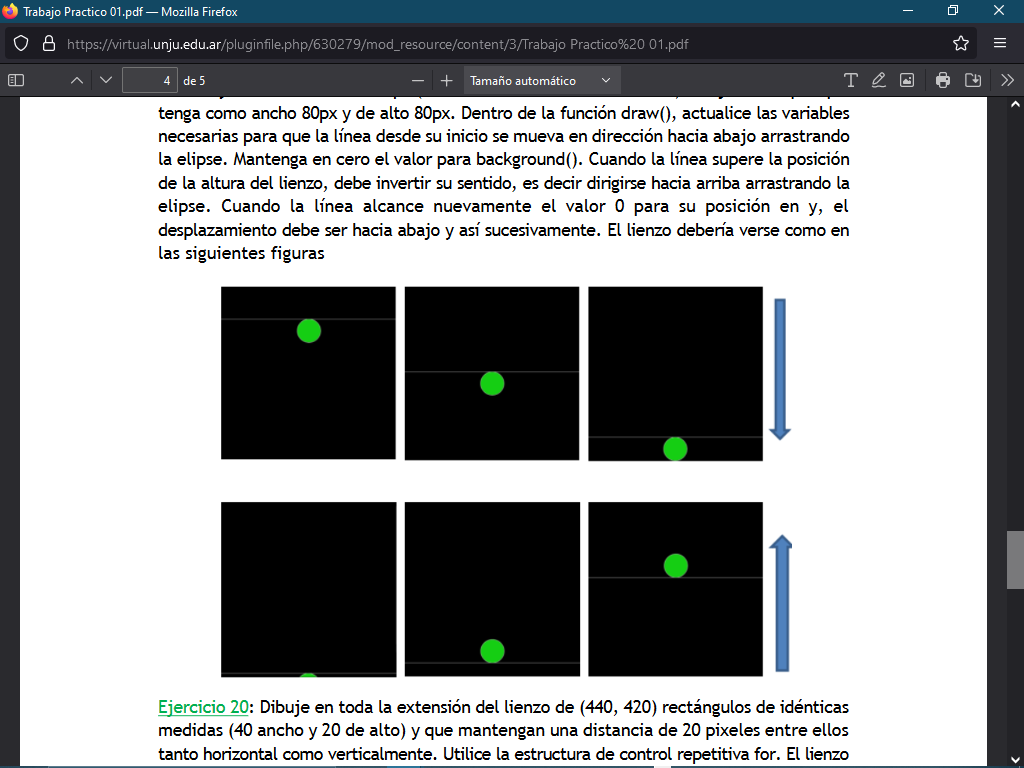
|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA: Calculadora\_Cientifica** |
| **VARIABLES a, b, c : Real // son los coeficientes de la ecuación y a debe ser diferente de 0 para que la ecuación exista discriminante: Real// sirve para analizar el comportamiento de las raíces opcion: Int // almacenara una variable auxiliar x1, x2: Real // almacena las raíces x: Real // raíz doble** |

|  |
| --- |
| **NOMBRE ALGORITMO: calcular\_raices\_de\_una\_cuadratica PROCESO DEL ALGORITMO *1\_Mostrar* “Ingrese los coeficientes de la ecuación” *2\_Mostrar* “Valor a” *3\_Leer* a *4\_Mostrar* “Valor b” *5\_Leer* b *6\_Mostrar* “Valor c” *7\_Leer* c *8\_*opcion <- 1 *9\_*discriminante <- b^2-4\*a\*c *10\_*si (discriminante < 0) entonces 11\_ opcion <- 2 12\_fin\_si 13\_si (discriminante = 0) entonces 14\_ opcion <- 3 15\_fin\_si 16\_segun\_sea (opcion) hacer 17\_ caso 1: 18\_ x1=(-b+(b^2-4\*a\*c)^(1/2))/(2\*a) 19\_ x2=(-b-(b^2-4\*a\*c)^(1/2))/(2\*a) 20\_ *Mostrar “*La ecuación tiene dos raíces y son x1= *”*+ x1 + “ y x2= ” + x2 21\_ *sentencia de ruptura* 22\_ caso 2: 23\_ *Mostrar “*No es posible calcular debido a que las raíces se encuentran en  el campo de los números complejos*” 24\_ sentencia de ruptura* 25\_ caso 3: 26\_ x=-b/(2\*a) 27\_ *Mostrar “*La ecuación tiene raíz doble y es x= *” +* x 28\_ *sentencia de ruptura* 29\_ otros: 30\_ *Mostrar “*Error*” 31\_ sentencia de ruptura 32\_*fin\_segun** |

## 

## 

## Ejercicio 19

****

**Definición del Problema: dibujar un círculo y una línea que suban y bajen permanentemente.**

**Análisis:**

**• Datos de Entrada:**

**posY:int**

**incremento: int**

**radio: int**

**• Datos de Salida:**

**Ellipse: figura**

**Line: figura**

**• Proceso:**

**¿Quién debe realizar el proceso?: Usuario**

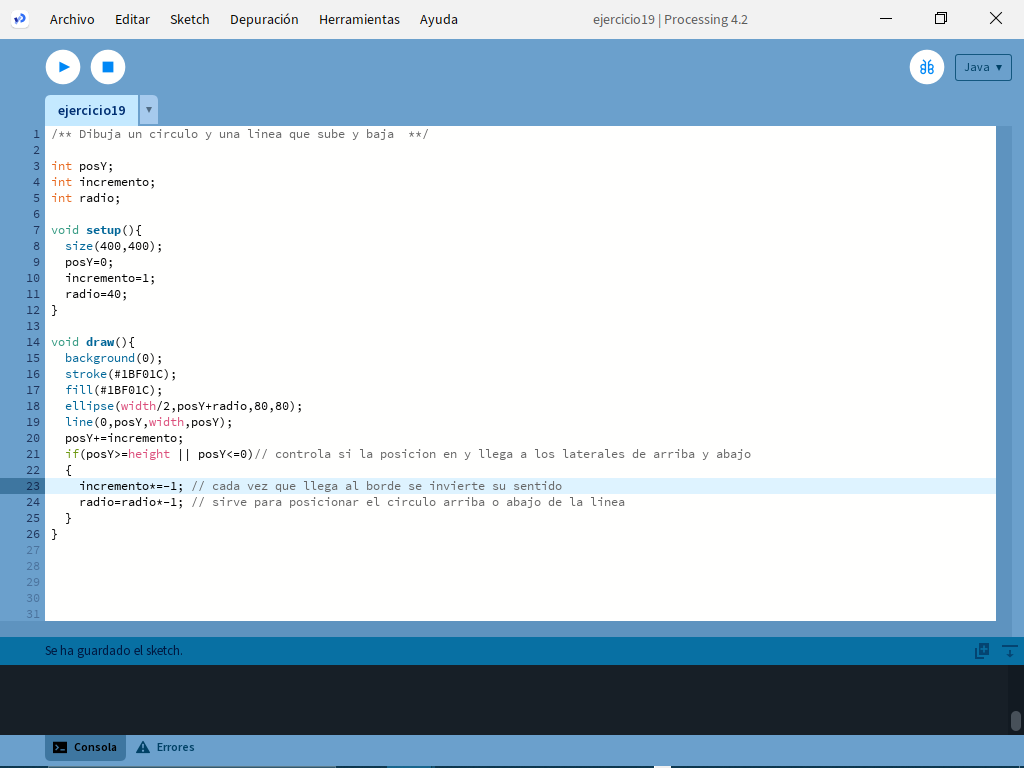
**¿Cuál es el proceso que realiza?**

**Dibuja un circulo y una línea que suben y bajan**

**Diseño**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:Usuario** |
| **VARIABLES posY : Entero // controla la posición en y de las figuras incremento: Entero// incrementa en una unidad constantemente**  **radio: Entero// guarda el radio del circulo**  **ellipse: figura // figura circular line: figura // linea** |

|  |
| --- |
| **Nombre del Algoritmo: dibujar\_circulo\_y\_linea** |
| **Proceso del algoritmo:**   1. **posY<- 0** 2. **incremento<- 1** 3. **radio<- 40** 4. **void draw(){** 5. **background(0);** 6. **ellipse(width/2,posY+radio,80,80)** 7. **line(0,posY,width,posY)** 8. **posY<-posY+incremento** 9. **SI(posY>=height o posY<=0) ENTONCES** 10. **Incremento<- incremento \* (-1)** 11. **Radio <- radio \* (-1)** 12. **FIN\_SI** 13. **}** |

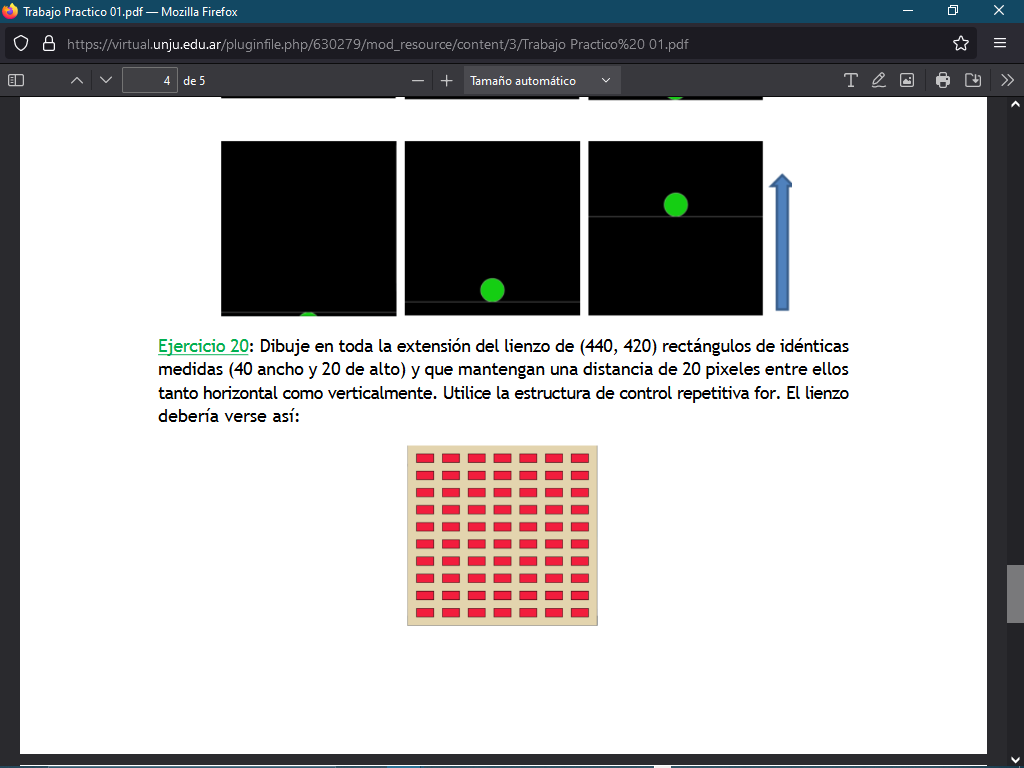
****

## Ejercicio 20

**Dibuje en toda la extensión del lienzo de (440, 420) rectángulos de idénticas**

**medidas (40 ancho y 20 de alto) y que mantengan una distancia de 20 pixeles entre ellos**

**tanto horizontal como verticalmente. Utilice la estructura de control repetitiva for. El lienzo**

**debería verse así:**

**ANALISIS**

***DEFINICION:* Dibujar rectangulos en todo el lienzo de manera ordenada usando el bucle for.**

**Datos de entrada: ancho del rectangulo, altura del rectangulo, distancia entre rectangulos**

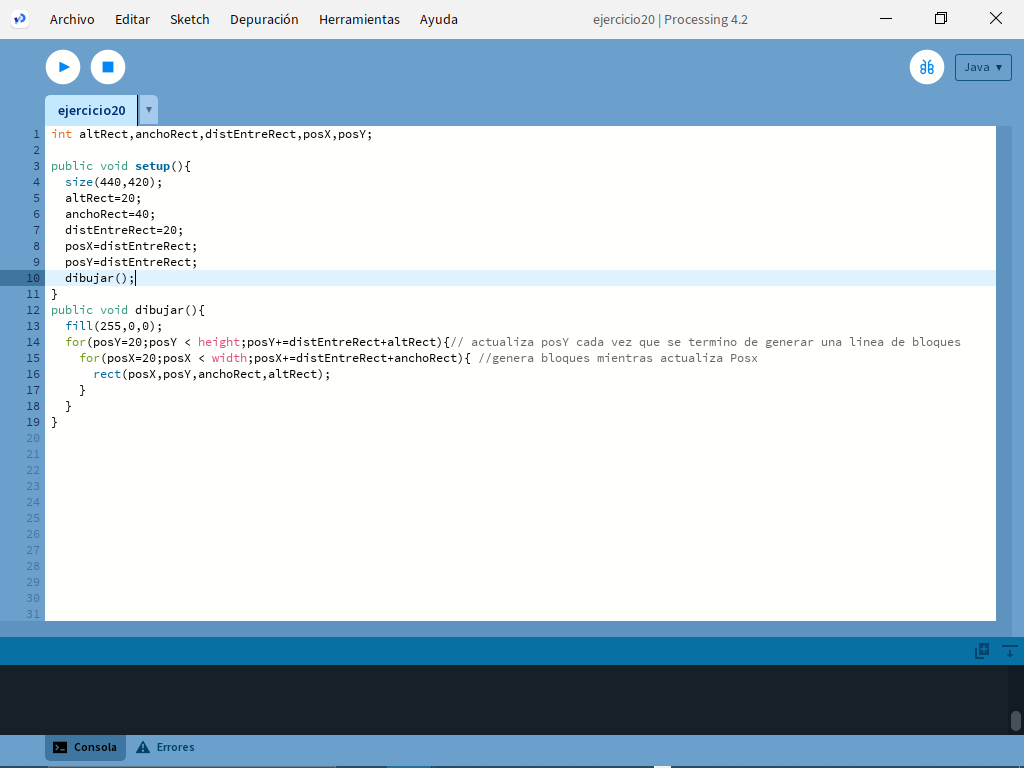
**Datos de salida: Rectangulo**

**Proceso: dibuja rectangulos por todo el lienzo**

**DISEÑO**

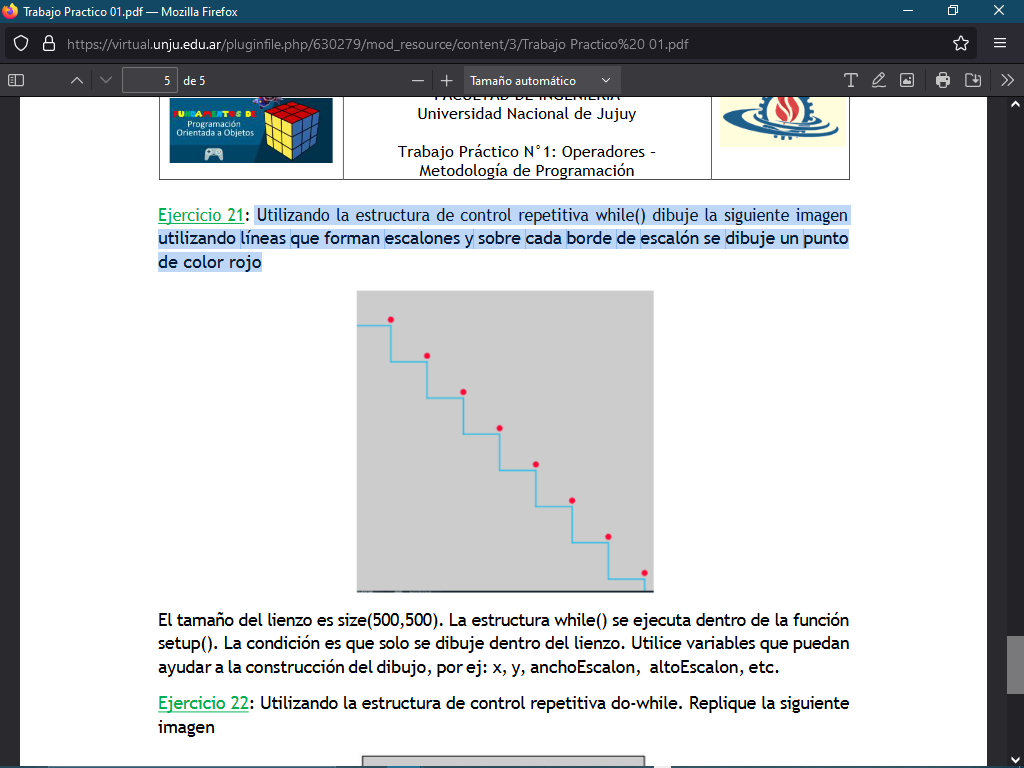
|  |
| --- |
| **Entidad que resuelve el problema: USUARIO** |
| **Variables:**   * **posX,posY: int//almacena un valor de coordenadas** * **anchoRect, altRect, distEntreRect: int// almacena valores enteros para construir el rectangulo** |

|  |
| --- |
| **Nombre del Algoritmo: dibujar\_rectangulos** |
| **Proceso del algoritmo:**   1. **anchoRect🡨40** 2. **altRect🡨20** 3. **distEntreRect🡨20** 4. **para (posY <-20) hasta alturaLienzo con paso (altRect+distanciaEntreRect) hacer** 5. **para posX <-20 hasta anchoLienzo con paso (anchoRect+distanciaEntreRect) hacer** 6. **dibujar rectángulo en (posX,posY,anchoRect,altRect)** 7. **Fin\_para** 8. **Fin\_para** |

****

## Ejercicio 21

**Utilizando la estructura de control repetitiva while() dibuje la siguiente imagen utilizando líneas que forman escalones y sobre cada borde de escalón se dibuje un punto de color rojo.**

****

**Definición del Problema: dibujar un circulo y una línea que suban y bajen permanentemente.**

**Análisis:**

**• Datos de Entrada:**

**x:int**

**y:int**

**altoEsc:int**

**anchoEsc:int**

**• Datos de Salida:**

**Line: figura**

**Point: figura**

**• Proceso:**

**¿Quién debe realizar el proceso?: Lienzo**

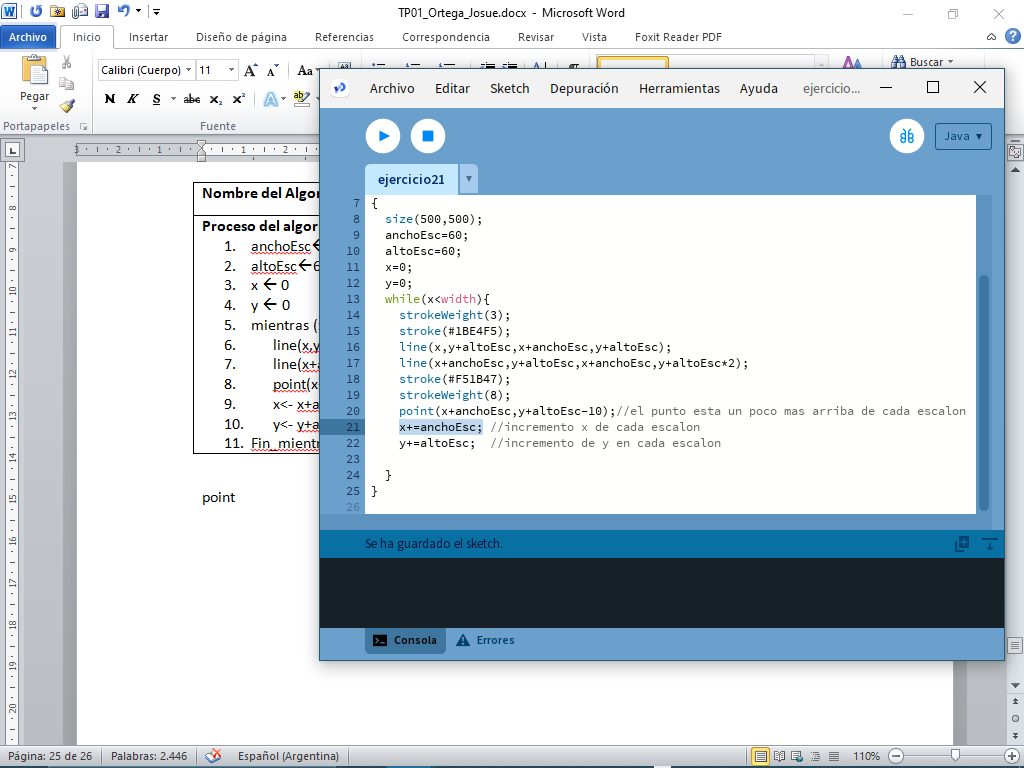
**¿Cuál es el proceso que realiza ?**

**Dibuja una escalera conociendo el ancho y la altura de cada escalon, y un punto rojo encima**

**Diseño**

|  |
| --- |
| **Entidad que resuelve el problema: Lienzo** |
| **Variables:**   * **x, y: int//almacena un valor de coordenadas** * **anchoEsc, altoEsc: int// almacena valores enteros para construir cada escalon** |

|  |
| --- |
| **Nombre del Algoritmo: dibujar\_escalera\_con\_puntos** |
| **Proceso del algoritmo:**   1. **anchoEsc🡨60** 2. **altoEsc🡨60** 3. **x 🡨 0** 4. **y 🡨 0** 5. **mientras (x<width) hacer** 6. **line(x,y+altoEsc,x+anchoEsc,y+altoEsc)** 7. **line(x+anchoEsc,y+altoEsc,x+anchoEsc,y+altoEsc\*2)** 8. **point(x+anchoEsc,y+altoEsc-10)** 9. **x<- x+anchoEsc** 10. **y<- y+altoEsc** 11. **Fin\_mientras** |

****

## Ejercicio 22

**Utilizando la estructura de control repetitiva do-while. Replique la siguiente imagen.**

****

**Definición del Problema: dibujar círculos de colores aleatorios sobre líneas de manera intercalada**

**Análisis:**

**• Datos de Entrada:**

**x:int**

**y:int**

**separacionLinea:int**

**cont:int**

**radio:int**

**• Datos de Salida:**

**Line: figura**

**Ellipse:figura**

**• Proceso:**

**¿Quién debe realizar el proceso?: Lienzo**

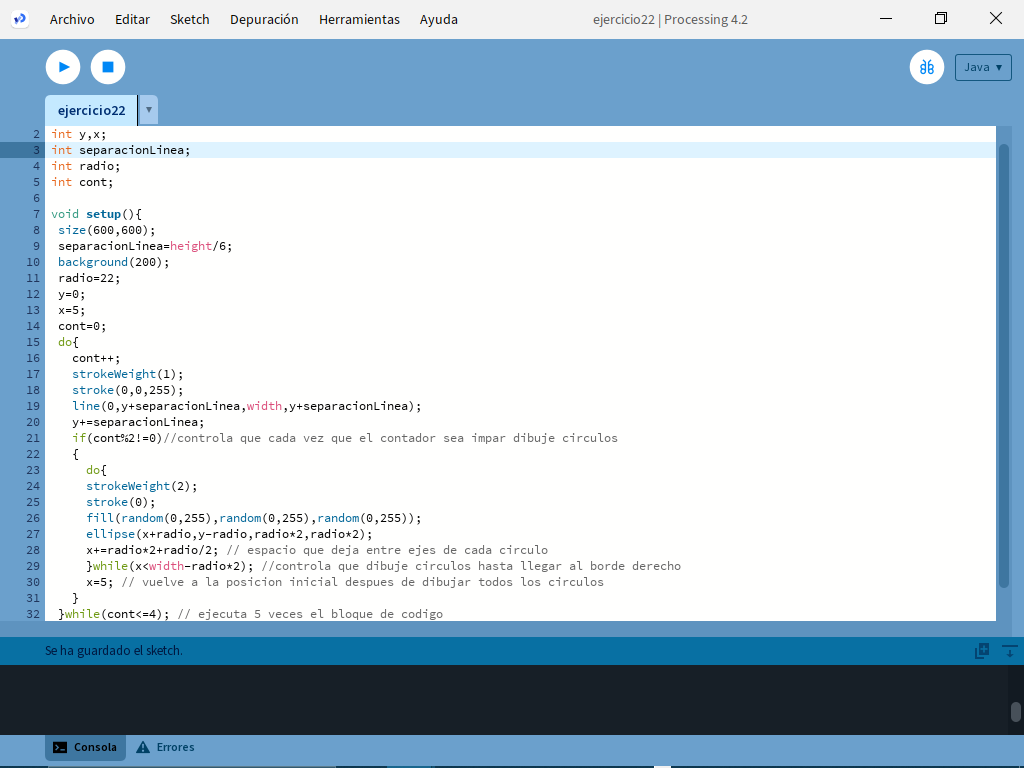
**¿Cuál es el proceso que realiza?**

**Realiza círculos sobre línea de manera intercalada**

**Diseño**

|  |
| --- |
| **Entidad que resuelve el problema: Lienzo** |
| **Variables:**   * **x, y: int//almacena un valor de coordenadas** * **separacionLinea : int// guarda la distancia entre cada línea** * **cont: int // almacena un contador que controla la cantidad de veces para ejecutar el código** * **radio: int // guarda el valor del radio de los circulos** |

|  |
| --- |
| **Nombre del Algoritmo: dibujar\_circulos\_sobre\_lineas** |
| **Proceso del algoritmo:**   1. **radio🡨20** 2. **separacionLinea🡨 height/6** 3. **x 🡨 5 // para que no queden pegados al borde los circulos** 4. **y 🡨 0** 5. **hacer** 6. **cont<- cont+1** 7. **line(0,y+separacionLinea,width,y+separacionLinea)** 8. **y<- y+separacionLinea** 9. **Si cont%2 !=0 entonces** 10. **hacer** 11. **ellipse(x+radio,y-radio,radio\*2,radio\*2)** 12. **x<- x+ radio\*2+radio/2** 13. **mientras (x<width-radio\*2)** 14. **x<- 5** 15. **Fin\_si** 16. **mientras ( cont<=4)** |

****

**Conclusión**

**Processing es una herramienta que no admite inputs, y tiene muchas funciones interesantes, como poder realizar formas geométricas, y además permite simular el funcionamiento de videojuegos.**

**La parte que más complicaciones tuve fue en la de estructuras iterativas, es complejo usarlas, para dar movimientos a las figuras.**

**Fuentes bibliográficas**

**Se deben enunciar las fuentes (apuntes de la materia, páginas web, videos de youtube, libro (nombre, autores, año), etc)**