





# Trabajo practico N°1

FUNDAMENTOS DE LA PROGRAMACIÓN ORIENTADA A OBJETOS

VALENTINA ROCIO ANAHÍ BEJARANO CÁCERES

[VALEXY@GMAIL.COM](mailto:VALEXY@GMAIL.COM) TUV000736

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

# FUNDAMENTOS DE LA PROGRAMACION ORIENTADO A OBJETOS

## TRABAJO PRÁCTICO/ACTIVIDAD

Nº1

APELLIDO Y NOMBRE – LU/



BEJARANO CACERES VALENTINA ROCIO ANAHI

TUV000736

PROFESOR:



MG. ING. ARIEL ALEJANDRO VEGA

Año: 2024

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS  TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS  <b>FACULTAD DE INGENIERÍA</b>  Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

## INDICE:

RESOLUCION DE EJERCICIOS.....	3
FIN DE LA RESOLUCIÓN.....	27
REFERENCIAS.....	28

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

## SECCIÓN EXPRESIONES ARITMÉTICAS Y LÓGICAS

RESOLVER CADA EJERCICIO EN UN ARCHIVO WORD Y LUEGO PROGRAMARLO EN PROCESSING. EN EL CASO DE LA PROGRAMACIÓN CREAR UN ARCHIVO POR EJERCICIO.

**EJERCICIO 1:** EVALUAR (OBTENER RESULTADO) LA SIGUIENTE EXPRESIÓN PARA  $A = 2$  Y  $B = 5$

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

RESOLUCIÓN NECESARIA EN WORD:

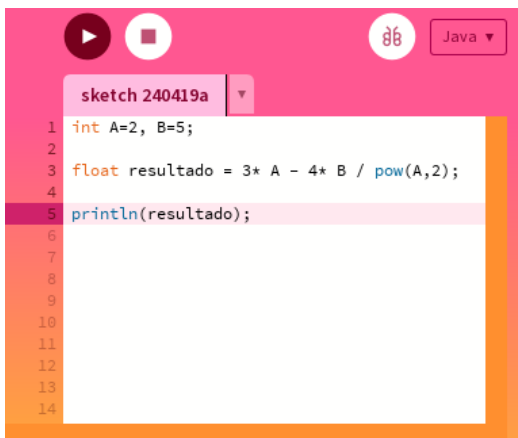
$$(3 * A) - (4 * B / (A^2))$$

$$6 - (4 * 5 / 4)$$

$$6 - 5$$

$$1$$

CAPTURA DE PROCESSING / RESULTADO:



```

1 int A=2, B=5;
2
3 float resultado = 3* A - 4* B / pow(A,2);
4
5 println(resultado);
6
7
8
9
10
11
12
13
14

```



**EJERCICIO 2:** EVALUAR LA SIGUIENTE EXPRESIÓN:

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

$$1 + 3 / 1 / 25 / 4 * 2$$

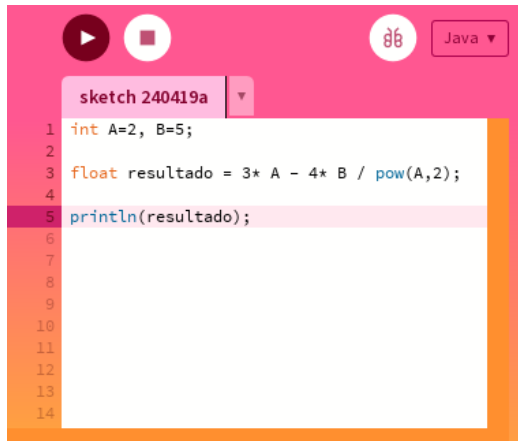
$$1 + 0,12 / 4 * 2$$

$$1 + 0,03 * 2$$

$$1,06$$

### PROCESSING:

### RESULTADO:



```

1 int A=2, B=5;
2
3 float resultado = 3* A - 4* B / pow(A,2);
4
5 println(resultado);
6
7
8
9
10
11
12
13
14

```



### EJERCICIO 3: (INCOMPLETO)

**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)  $B^2 - 4 * A * C$

$$b^2 - 4ac$$

A=2, B=4, C=1

$$4^2 - 4 \cdot 2 \cdot 1$$

$(4^2 - 4 * 2 * 1)$

$$4^2 - 4 \cdot 2 \cdot 1$$

$16 - 8$

$$16 - 8 = 8$$

8

### PROCESSING:

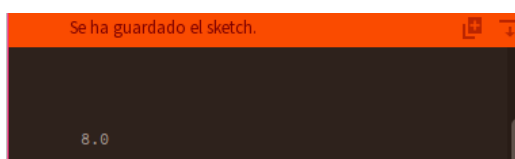
### RESULTADO:



```

1 int a = 2;
2 int b = 4;
3 int c = 1;
4
5 float resultado = pow(b, 2)-4*a*c;
6
7 println(resultado);
8
9
10
11
12
13
14

```



B)  $3 * X^4 - 5 * X^3 + X^2 - 17$

$$3X^4 - 5X^3 + 12x - 17$$

$3 * 2^4 - 5 * 2^3 + 12 * 2 - 17$

$$3 \cdot 2^4 - 5 \cdot 2^3 + 12 \cdot 2 - 17$$

$48 - 40 + 24 - 17$

$$48 - 40 + 24 - 17$$

15

$$15$$

### PROCESSING:

```

1 int x = 2;
2
3 float resultado= 3 * pow(x, 4) - 5 * pow(x, 3) + 12 * x - 17;
4
5 println(resultado);
6
7
8
9
10
11
12
13
14

```

### RESULTADO:

15.0

$$C) (B + D) / (C + 4)$$

$$B=1 \ C=2 \ D=3$$

$$(1 + 3) / (2 + 4)$$

$$4 / 6$$

$$0,66666666$$

$$(B + D) : (C + 4)$$

$$(1 + 3) : (2 + 4)$$

$$4 : 6$$

$$0,6666$$

### PROCESSING:

```

1 int B=1;
2 int C=2;
3 int D=3;
4
5 float resultado= (B+D)/ float(C+4);
6
7 println(resultado);
8
9
10
11
12
13
14

```

### RESULTADO:

Se ha guardado el sketch.

0.66666667

$$D) (X^2 + Y^2)^{(1/2)}$$

$$(2^2 + 3^2)^{(1/2)}$$

$$13^{(1/2)}$$



$$3,6055512$$

$$(x^2 + y^2)^{\left(\frac{1}{2}\right)}$$

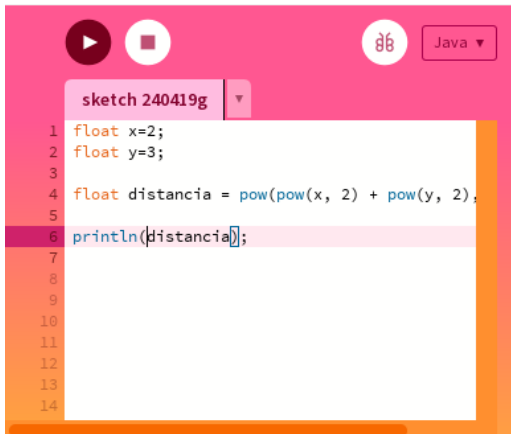
$$(2^2 + 3^2)^{\left(\frac{1}{2}\right)}$$

$$13^{\left(\frac{1}{2}\right)} = 3.605512$$



	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

#### PROCESSING:



```

1 float x=2;
2 float y=3;
3
4 float distancia = pow(pow(x, 2) + pow(y, 2), 0.5);
5
6 println(distancia);
7
8
9
10
11
12
13
14

```

#### RESULTADO



PARA ACLARAR QUE INDICAMOS CON” LUEGO ESCRIBIRLAS COMO EXPRESIONES ALGEBRAICAS” LO APLICAMOS CON EL PUNTO A)

$$b^2 - 4 \cdot a \cdot c$$

**EXERCICIO 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:

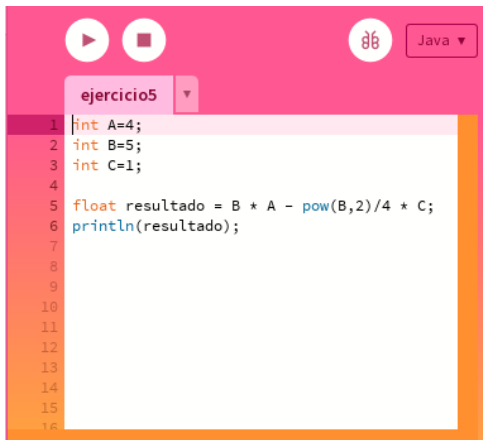
A)  $B * A - B^2 / 4 * C$

$5 * 4 - 5^2 / 4 * 1$

$20 - 25 / 4$

$13,75$

#### PROCESSING:

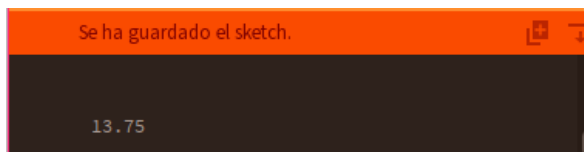


```

1 int A=4;
2 int B=5;
3 int C=1;
4
5 float resultado = B * A - pow(B,2)/4 * C;
6 println(resultado);
7
8
9
10
11
12
13
14
15
16

```



#### RESULTADO:



B)  $(A * B) / 3^2$

$(4 * 5) / 3^2$

$(20) / 9$

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

2.222223

PROCESSING:

```

1 int A=4;
2 int B=5;
3
4 float resultado = (A * B) / pow(3,2);
5
6 println(resultado);|

```

RESULTADO:

2.222223

C)  $((B + C) / 2 * A + 10) * 3 * B) - 6$

$((5 + 1) / 2 * 4 + 10) * 3 * 5) - 6$

$((6 / 2 * 4 + 10) * 3 * 5) - 6$

$(22 * 3 * 5) - 6$

$330 - 6$

$324$

PROCESSING:

```

1 int A=4;
2 int B=5;
3 int C=1;
4
5 float resultado = (((B+C)/2*A+10)*3*B)-6;
6
7 println(resultado);|

```

RESULTADO:

Se ha guardado el sketch.

324.0

**EJERCICIO 6:** PARA X=3, Y=4; Z=1, EVALUAR EL RESULTADO DE

R1 = Y+Z



R2 = X >= R1

R1= 4+1

R1= 5

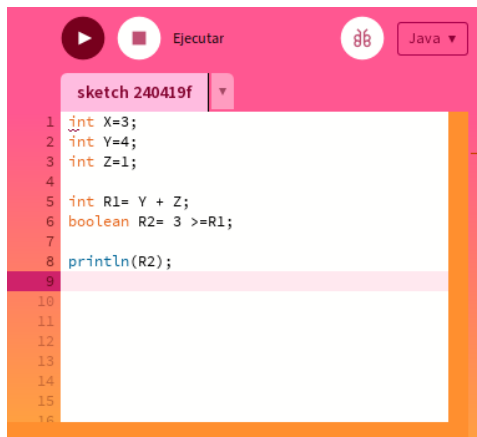
R2= 3 > 5



	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

R2= FALSO

PROCESSING:



```

1 int X=3;
2 int Y=4;
3 int Z=1;
4
5 int R1= Y + Z;
6 boolean R2= 3 >=R1;
7
8 println(R2);
9
10
11
12
13
14
15
16

```

RESULTADO:



**EXERCICIO 7:** PARA CONTADOR1=3, CONTADOR2=4, EVALUAR EL RESULTADO DE

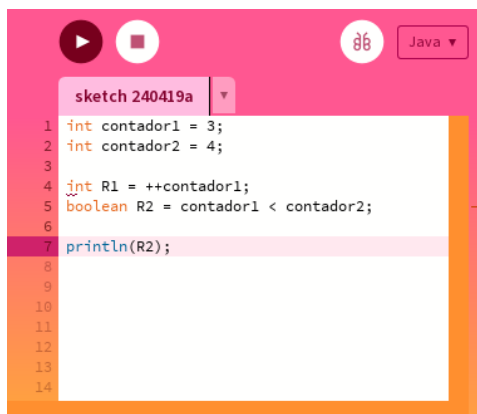
R1 = ++CONTADOR1

R2 = CONTADOR1 < CONTADOR2

R1= ++3= 4

R2= 4 < 4...FALSO

PROCESSING:



```

1 int contador1 = 3;
2 int contador2 = 4;
3
4 int R1 = ++contador1;
5 boolean R2 = contador1 < contador2;
6
7 println(R2);
8
9
10
11
12
13
14

```

RESULTADO:





**EXERCICIO 8:** PARA A=31, B=-1, X=3, Y=2, EVALUAR EL RESULTADO DE A+B-1 < X\*Y

31 + (-1) -1 < 3 \* 2

31 -1 -1 < 6

31 -2 < 6

29 < 6...FALSO

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

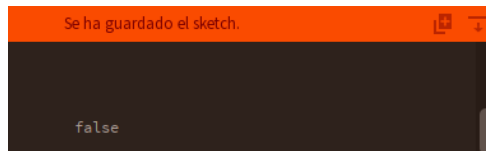
#### PROCESSING:

```

ejercicio8
1 int a = 31;
2 int b = 1;
3 int x = 3;
4 int y = 2;
5
6 boolean resultado= a + b -1 < x * y;
7
8 println(resultado);
9
10
11
12
13
14

```

#### RESULTADO:



#### EJERCICIO 9: PARA X=6, Y=8, EVALUAR EL RESULTADO DE !(X<5) &&!(Y>=7)

!(6<5) && !(8>=7)

(5<6) && (7>=8)

VERDADERO && FALSO

FALSO

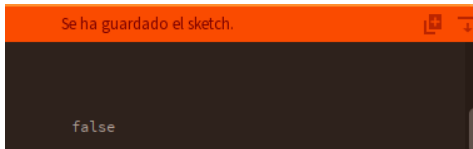
#### PROCESSING:

```

ejercicio9
1 int x = 6;
2 int y = 8;
3
4 boolean resultado = !(x < 5) && !(y >= 7);
5
6 println(resultado);
7
8
9
10
11
12
13
14

```

#### RESULTADO:



#### EJERCICIO 10: PARA I=22, J=3, EVALUAR EL RESULTADO DE



!(I>4) || !(J<=6)

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

!(VERDADERO) || !(VERDADERO)

FALSO || FALSO

FALSO

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

PROCESSING:

RESULTADO:

```

ejercicio10
1 int i=22;
2 int j=3;
3
4 boolean resultado= !((i>4) || !(j<=6));
5
6 println(resultado);
7
8
9
10
11
12
13
14

```

```

Se ha guardado el sketch.

false

```

**EJERCICIO 11:** PARA A=34, B=12, C=8, EVALUAR EL RESULTADO DE  $!(A+B==C) \parallel (C!=0) \&\& (B-C>=19)$

$!(A+B==C) \parallel (C!=0) \&\& (B-C>=19)$

$!(34 + 12 == 8) \parallel (8 != 0) \&\& (12 - 8 >= 19)$

$!(46 == 8) \parallel (8 != 0) \&\& (4 >= 19)$

VERDADERO  $\parallel$  VERDADERO  $\&\&$  FALSO

VERDADERO  $\parallel$  FALSO

VERDADERO

PROCESSING:

RESULTADO:

```

ejercicio11
1 int a=34;
2 int b=12;
3 int c=8;
4
5 boolean resultado= !(a+b==c) || (c!=0) && (b-c>=19);
6
7 println(resultado);
8
9
10
11
12
13
14
15

```

```



Se ha guardado el sketch.

true

```

**SECCIÓN ANÁLISIS – DISEÑO Y CODIFICACIÓN DE ALGORITMOS – APLICACIÓN DE ESTRUCTURAS DE CONTROL**

PARA CADA EJERCICIO, EN EL ARCHIVO WORD AGREGAR LAS SECCIONES DE ANÁLISIS Y DISEÑO, MIENTRAS QUE, PARA LA CODIFICACIÓN, CREAR EL ARCHIVO DE PROCESSING.

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

**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.

ANÁLISIS DEL PROBLEMA
<p><b>DATOS DE ENTRADA:</b> NOMBRE_INGRESADO //CADENA</p> <p><b>DATOS DE SALIDA:</b> MENSAJE_SALUDO //CADENA DE TEXTO</p> <p><b>PROCESO:</b></p> <p>¿QUIEN DEBE REALIZAR EL PROCESO?: ALGORITMO</p> <p>¿CUAL ES EL PROCESO QUE RESUELVE?: PROPORCIONAR UN NOMBRE QUE DEVOLVERÁ LA CREACIÓN DE UN SALUDO CON UN NOMBRE Y SU PRESENTACIÓN EN PANTALLA.</p>
DISEÑO DEL PROBLEMA
ENTIDAD QUE RESUELVE EL PROBLEMA: ALGORITMO
<p><b>VARIABLES:</b></p> <ul style="list-style-type: none"> <li>♥ <b>TEXTO_CONSOLE:</b> ESTA VARIABLE PARECE SER UTILIZADA PARA IMPRIMIR UN TEXTO EN LA CONSOLA, PERO NO ESTÁ DEFINIDA EN EL CÓDIGO PROPORCIONADO.</li> <li>♥ <b>MENSAJE_SALUDO:</b> ESTA VARIABLE ALMACENA EL MENSAJE DE SALUDO QUE SE MOSTRará EN LA VENTANA GRÁFICA. SE ACTUALIZA DINÁMICAMENTE SEGÚN EL NOMBRE INGRESADO POR EL USUARIO.</li> <li>♥ <b>NOMBRE_INGRESADO:</b> ESTA VARIABLE ALMACENA EL NOMBRE INGRESADO POR EL USUARIO A MEDIDA QUE SE PRESIONAN LAS TECLAS. SE UTILIZA PARA CONSTRUIR EL MENSAJE DE SALUDO PERSONALIZADO.</li> </ul>
NOMBRE DEL ALGORITMO: NOMBRE_SALUDAR
<p><b>PROCESO DEL ALGORITMO:</b></p> <ul style="list-style-type: none"> <li>♥ <b>INICIO</b></li> <li>♥ <b>LEER NOMBRE_INGRESADO</b></li> <li>♥ <b>MENSAJE_SALUDO</b> ← “HOLA, ” + NOMBRE_INGRESADO + “ ¡BIENVENIDO!”</li> <li>♥ <b>MOSTRAR SALUDO</b></li> <li>♥ <b>FIN</b></li> </ul>

## PROCESSING Y RESULTADO

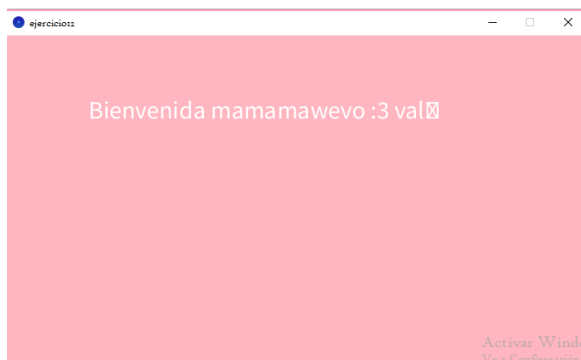


```

ejercicio12
5 void setup(){
6   size(500,500);
7   println(texto_consola);
8
9 }
10
11 void draw(){
12   background(#FFB6C1);
13   text(mensaje_saludo,100,100);
14   textSize(30);
15 }
16
17 void keyPressed(){
18   nombre_ingresado+= key;
19   println(nombre_ingresado);
20   mensaje_saludo = "Bienvenida mamamawevo :3 " + nombre_ingresado;
21
22   if (key == '\n'){
23     mensaje_saludo = "Bienvenida lokita, " + nombre_ingresado;
24   }
25 }
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100



```

## VENTANA DE VISUALIZACION



**EJERCICIO 12:** SERÁ COMÚN RESOLVER PROBLEMAS UTILIZANDO VARIABLES. CALCULE EL PERÍMETRO Y ÁREA DE UN RECTÁNGULO DADA SU BASE Y SU ALTURA.

<b>ANÁLISIS DEL PROBLEMA</b>
<p><b>DATOS DE ENTRADA:</b> BASE, ALTURA</p> <p><b>DATOS DE SALIDA:</b> PERÍMETRO Y ÁREA</p> <p><b>PROCESO:</b></p> <p>¿QUIEN DEBE REALIZAR EL PROCESO?: ALGORITMO</p> <p>¿CUAL ES EL PROCESO QUE RESUELVE?: PROPORCIONA VALORES DE BASE Y ALTURA PARA DEVOLVER LOS VALORES DE PERÍMETRO Y ÁREA</p>
<b>DISEÑO DEL PROBLEMA</b>
ENTIDAD QUE RESUELVE EL PROBLEMA: ALGORITMO
<p><b>VARIABLES:</b></p> <p>♥ <b>BASE:</b> ALMACENA LA LONGITUD DE LA BASE DEL RECTÁNGULO.</p> <p>♥ <b>ALTURA:</b> ALMACENA LA LONGITUD DE LA ALTURA DEL RECTÁNGULO.</p>

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

<p>♥ <b>PERÍMETRO:</b> ALMACENA EL VALOR DEL PERÍMETRO DEL RECTÁNGULO, CALCULADO COMO <math>2 \times \text{BASE} + 2 \times \text{ALTURA}</math>.</p> <p>♥ <b>ÁREA:</b> ALMACENA EL VALOR DEL ÁREA DEL RECTÁNGULO, CALCULADO COMO <math>\text{BASE} \times \text{ALTURA}</math>.</p>
<p><b>NOMBRE DEL ALGORITMO:</b> RECTANGULO_AREA_PERIMETRO</p>
<p><b>PROCESO DEL ALGORITMO:</b></p> <ul style="list-style-type: none"> <li>♥ <i>INICIO</i></li> <li>♥ <i>DEFINIR LAS VARIABLES</i></li> <li>♥ <i>CALCULA EL PERIMETRO</i></li> <li>♥ <i>CALCULA EL AREA</i></li> <li>♥ <i>MUESTRA LOS RESULTADOS</i></li> <li>♥ <i>FIN</i></li> </ul>

#### PROCESSING:

```

ejercicio13
1 int base = 4;
2 int altura = 2;
3
4 int resultado_perimetro = 2 * base + 2 * altura;
5 int resultado_area = base * altura;
6
7 println(resultado_perimetro);
8 println(resultado_area);
9
10
11
12
13
14
15
16
17
18
19

```

#### RESULTADO:



```

12
8

```

**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

<p><b>ANÁLISIS DEL PROBLEMA</b></p>
<p><b>DATOS DE ENTRADA:</b> CATETO1-CATETO2</p> <p><b>DATOS DE SALIDA:</b> RESULTADO DE LA HIPOTENUSA</p> <p><b>PROCESO:</b></p> <p>¿QUIEN DEBE REALIZAR EL PROCESO?: ALGORITMO</p> <p>¿CUAL ES EL PROCESO QUE RESUELVE?: PROPORCIONAR VALORES A LOS CATETOS Y DEVOLVER EL VALOR DE LA HIPOTENUSA</p>
<p><b>DISEÑO DEL PROBLEMA</b></p>

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

#### ENTIDAD QUE RESUELVE EL PROBLEMA: ALGORITMO

##### VARIABLES:

- ♥ **TEXTO\_CONSOLE:** ESTA VARIABLE PARECE SER UTILIZADA PARA IMPRIMIR UN TEXTO EN LA CONSOLA, PERO NO ESTÁ DEFINIDA EN EL CÓDIGO PROPORCIONADO.
- ♥ **MENSAJE\_SALUDO:** ESTA VARIABLE ALMACENA EL MENSAJE DE SALUDO QUE SE MOSTRará EN LA VENTANA GRÁFICA. SE ACTUALIZA DINÁMICAMENTE SEGÚN EL NOMBRE INGRESADO POR EL USUARIO.
- ♥ **NOMBRE\_INGRESADO:** ESTA VARIABLE ALMACENA EL NOMBRE INGRESADO POR EL USUARIO A MEDIDA QUE SE PRESIONAN LAS TECLAS. SE UTILIZA PARA CONSTRUIR EL MENSAJE DE SALUDO PERSONALIZADO.

ESAS SON LAS VARIABLES QUE SE UTILIZAN EN EL CÓDIGO PROPORCIONADO.

##### NOMBRE DEL ALGORITMO: CALCULOS BASICOS

##### PROCESO DEL ALGORITMO:

- ♥ **INICIO**
- ♥ **LEER NOMBRE\_INGRESADO**
- ♥ **MENSAJE\_SALUDO** ← "HOLA " + NOMBRE\_INGRESADO + " ;BIENVENIDO!"
- ♥ **MOSTRAR SALUDO**
- ♥ **FIN**

##### PROCESSING:

```

ejercicio13
1 | int base = 4;
2 | int altura = 2;
3 |
4 | int resultado_perimetro = 2 * base + 2 * altura;
5 | int resultado_area = base * altura;
6 |
7 | println(resultado_perimetro);
8 | println(resultado_area);
9 |
10 |
11 |
12 |
13 |
14 |
15 |
16 |
17 |
18 |
19 |

```

##### RESULTADO:



```

12
8

```

**EXERCICIO 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. OBTIENGA MUESTRE



	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS</p> <p>FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	---	---

## LOS RESULTADOS.

### PROCESSING:

```

1 int num1=110;
2 int num2=12;
3
4 int suma=num1+num2;
5 println("suma_resultado"+suma);
6
7 int resta=num1-num2;
8 println("resta_resultado"+resta);
9
10 int multiplicacion=num1*num2;
11 println("multiplicacion_resultado"+multiplicacion);
12
13 int division=num1/num2;
14 if(num2!=0) {
15     println("division_resultado"+division);
16     println("la division por cero no esta definida");
17 }
18

```



### RESULTADO

```

suma_resultado122
resta_resultado98
multiplicacion_resultado1320
division_resultado9
la division por cero no esta definida

```

ANÁLISIS DEL PROBLEMA
<p><b>DATOS DE ENTRADA:</b> NUM1=110/NUM2=12</p> <p><b>DATOS DE SALIDA:</b> SUMA, RESTA, MULTIPLICACION, DIVISION</p> <p><b>PROCESO:</b></p> <p>¿QUEN DEBE REALIZAR EL PROCESO?: CALCULADORA</p> <p>¿CUAL ES EL PROCESO QUE RESUELVE?: REALIZA TODOS LOS CALCULOS CON LOS NUMEROS DADOS</p>
DISEÑO DEL PROBLEMA
ENTIDAD QUE RESUELVE EL PROBLEMA: CALCULADORA
<p><b>VARIABLES:</b></p> <ul style="list-style-type: none"> <li>♥ NUM1: ALMACENA EL PRIMER NÚMERO ENTERO, QUE ES 110.</li> <li>♥ NUM2: ALMACENA EL SEGUNDO NÚMERO ENTERO, QUE ES 12.</li> <li>♥ SUMA: ALMACENA EL RESULTADO DE LA SUMA DE NUM1 Y NUM2.</li> <li>♥ RESTA: ALMACENA EL RESULTADO DE LA RESTA DE NUM1 Y NUM2.</li> <li>♥ MULTIPLICACION: ALMACENA EL RESULTADO DE LA MULTIPLICACIÓN DE NUM1 Y NUM2.</li> <li>♥ DIVISION: ALMACENA EL RESULTADO DE LA DIVISIÓN DE NUM1 ENTRE NUM2.</li> </ul>
NOMBRE DEL ALGORITMO: CALCULOS BÁSICOS
<p><b>PROCESO DEL ALGORITMO:</b></p> <ul style="list-style-type: none"> <li>♥ INICIO</li> <li>♥ DEFINE LAS VARIABLES</li> <li>♥ REALIZA LOS CALCULOS</li> <li>♥ LANZA LOS RESULTADOS</li> <li>♥ FIN</li> </ul>

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

**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

$$\text{temperaturaCelsius} = (\text{temperaturaFahrenheit} - 32) / 1.8$$

<p><b>ANÁLISIS DEL PROBLEMA</b></p> <p><b>DATOS DE ENTRADA:</b> TEMPERATURA EN FAHRENHEIT</p> <p><b>DATOS DE SALIDA:</b> TEMPERATURA EN CELSIUS</p> <p><b>PROCESO:</b></p> <p>¿QUIEN DEBE REALIZAR EL PROCESO?: CALCULADORA</p> <p>¿CUAL ES EL PROCESO QUE RESUELVE?: TOMA LA TEMPERATURA DADA EN FAHRENHEIT Y LA CONVIERTE EN TEMPERATURA EN CELSIUS</p>
<p><b>DISEÑO DEL PROBLEMA</b></p>
<p>ENTIDAD QUE RESUELVE EL PROBLEMA: CALCULADORA</p>
<p><b>VARIABLES:</b></p> <ul style="list-style-type: none"> <li>♥ <b>TEMPERATURAFAHRENHEIT:</b> ALMACENA LA TEMPERATURA EN GRADOS FAHRENHEIT QUE SE DESEA CONVERTIR A GRADOS CELSIUS.</li> <li>♥ <b>TEMPERATURACELSIUS:</b> ALMACENA LA TEMPERATURA EQUIVALENTE EN GRADOS CELSIUS, CALCULADA MEDIANTE LA FÓRMULA DE CONVERSIÓN.</li> </ul>
<p>NOMBRE DEL ALGORITMO: CONVERSION DE TEMPERATURA</p>
<p><b>PROCESO DEL ALGORITMO:</b></p> <ul style="list-style-type: none"> <li>♥ <b>INICIO</b></li> <li>♥ <b>DETERMINAR LA TEMPERATURA EN FAHRENHEIT</b></li> <li>♥ <b>CALCULA LA EQUIVALENCIA EN CELSIUS</b></li> <li>♥ <b>ARROJA EL RESULTADO EN CELSIUS</b></li> <li>♥ <b>FIN</b></li> </ul>

**PROCESSING:**

```

ejercicio16
1 float temperatura_Fahrenheit = 34;
2
3 float temperatura_Celsius = (temperatura_Fahrenheit - 32) * 5 / 9;
4
5 println("La temperatura en grados Celsius es: " + temperatura_Celsius);
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26



```

**RESULTADO:**

```

La temperatura en grados Celsius es: 1.1111112

```

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

**EXERCICIO 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.

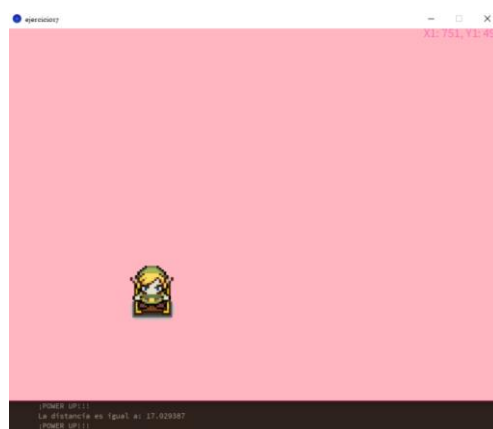
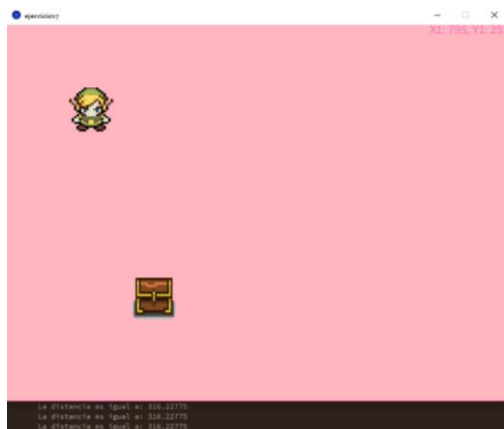
#### PROCESSING:



```

ejercicio17
1 float x1 = 100;
2 float y1 = 100;
3 float x2 = 200;
4 float y2 = 400;
5 PImage linkImage;
6 PImage tesoroImage;
7 float distanciaTesoro = 50;
8 float tamanoImagen = 70; // Tamaño de la imagen de Link y del tesoro
9
10 void setup() {
11   size(800, 600);
12   linkImage = loadImage("link.png");
13   tesoroImage = loadImage("treasurechest.png");
14   linkImage.resize((int)tamanoImagen, (int)tamanoImagen);
15   tesoroImage.resize((int)tamanoImagen, (int)tamanoImagen);
16 }
17
18 void draw() {
19   background(#FFB6C1);
20   float coordenadaX = x2 - x1;
21   float coordenadaY = y2 - y1;
22   float distancia = sqrt(pow(coordenadaX, 2) + pow(coordenadaY, 2));
23   String textoDistancia = "La distancia es igual a: " + distancia;
24   println(textoDistancia);
25
26   if (distancia <= distanciaTesoro) {
27     println("¡POWER UP!!!");
28   }
29
30   image(tesoroImage, x2, y2);
31   image(linkImage, x1, y1);
32
33   String coordenadas = "X1: " + mouseX + ", Y1: " + mouseY;
34   fill(#FF69B4);
35   textSize(20);
36   textAlign(RIGHT, TOP);
37   text(coordenadas, width, 0);
38 }
39
40 void mousePressed() {
41   // Verificar si el clic del mouse está dentro de la imagen de Link
42   if (mouseX > x1 && mouseX < x1 + tamanoImagen && mouseY > y1 && mouseY < y1 + tamanoImagen) {
43     x1 = mouseX - tamanoImagen / 2; // Centrar la imagen de Link en el cursor del mouse
44     y1 = mouseY - tamanoImagen / 2;
45   }
46 }
47
48
49
50
51
52

```

#### VENTANA DE VISUALIZACIÓN Y RESULTADO:



	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

ANÁLISIS DEL PROBLEMA
<p><b>DATOS DE ENTRADA:</b> COORDENADAS DE LA UBICACIÓN DE LINK Y EL TESORO, IMÁGENES (LINK Y ZELDA).</p> <p><b>DATOS DE SALIDA:</b> DISTANCIA ENTRE LINK Y EL TESORO Y COORDENADAS DEL MOUSE</p> <p><b>PROCESO:</b></p> <p>¿QUIEN DEBE REALIZAR EL PROCESO?: PROGRAMA DE PROCESSING</p> <p>¿CUAL ES EL PROCESO QUE RESUELVE?: RESUELVE EL PLANTEAMIENTO E INTERACCION ENTRE DOS IMÁGENES, DADO QUE UNO SE MUEVE AL INTERACTUAR CON UNA DE LAS IMAGENES</p>
DISEÑO DEL PROBLEMA
ENTIDAD QUE RESUELVE EL PROBLEMA: PROGRAMA DE PROCESSING
<p><b>VARIABLES:</b></p> <ul style="list-style-type: none"> <li>♥ X1, Y1: COORDENADAS DE LINK.</li> <li>♥ X2, Y2: COORDENADAS DEL TESORO.</li> <li>♥ LINKIMAGE, TESOROIMAGE: IMÁGENES DE LINK Y DEL TESORO.</li> <li>♥ DISTANCIATESORO: DISTANCIA MÁXIMA PARA ACTIVAR EL "POWER UP".</li> <li>♥ TAMANOIMAGEN: TAMAÑO DE LAS IMÁGENES DE LINK Y DEL TESORO.</li> </ul>
NOMBRE DEL ALGORITMO: TESORO LINK DISTANCIA
<p><b>PROCESO DEL ALGORITMO:</b></p> <ul style="list-style-type: none"> <li>♥ INICIO</li> <li>♥ CARGAR Y REAJUSTAR EL TAMAÑO DE LA IMAGEN</li> <li>♥ CALCULO DE LA DISTANCIA ENTRE LAS DOS IMÁGENES</li> <li>♥ DETERMINAR LA DISTANCIA EN LA QUE SE ACTIVARA EL POWER UP</li> <li>♥ MOSTRAR LAS COORDENADAS DEL MOUSE</li> <li>♥ PROBAR EL CODIGO EN LA VENTANA DE VISUALIZACION</li> <li>♥ FIN</li> </ul>

**EXERCICIO 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. OBTIENGA EL CODIGO EN PROCESSING.

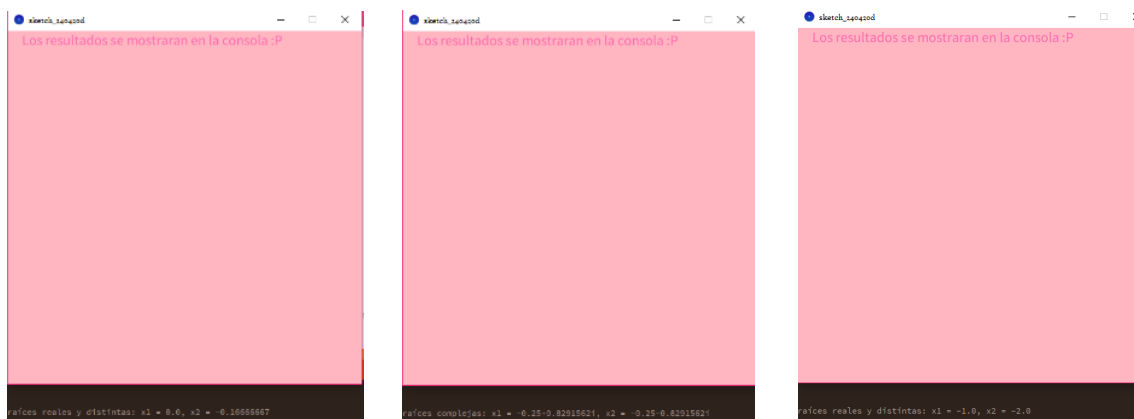
## PROCESSING:

```



1 float a=2;
2 float b=1;
3 float c=-1;
4
5 void setup() {
6   size(500, 500);
7   noLoop();
8   textSize(20);
9   fill(#FF69B4);
10
11   calcularRaices(a, b, c);
12 }
13
14 void calcularRaices(float a, float b, float c) {
15   if (a == 0) {
16     println("No es una ecuación cuadrática");
17     return;
18   }
19
20   float discriminante = b*b - 4*a*c;
21   float x1, x2, realPart, imagPart;
22
23   if (discriminante >= 0) {
24     x1 = (-b + sqrt(discriminante)) / (2*a);
25     x2 = (-b - sqrt(discriminante)) / (2*a);
26     println(discriminante == 0 ? "raíz única: x = " + x1 : "raíces reales y distintas: x1 = " + x1 + ", x2 = " + x2);
27   } else {
28     realPart = -b / (2*a);
29     imagPart = sqrt(-discriminante) / (2*a);
30     println("raíces complejas: x1 = " + realPart + "+" + imagPart + "i, x2 = " + realPart + "-" + imagPart + "i");
31   }
32 }
33
34 void draw() {
35   background(#FFB6C1);
36   text("Los resultados se mostrarán en la consola :P", 20, 20);
37 }

```

## RESULTADOS REEMPLAZANDO A B Y C:



<b>ANÁLISIS DEL PROBLEMA</b>
<p><b>DATOS DE ENTRADA:</b> VALORES DE A, B Y C</p> <p><b>DATOS DE SALIDA:</b> RAICES DE LA ECUACION</p> <p><b>PROCESO:</b></p> <p>¿QUIEN DEBE REALIZAR EL PROCESO?: EL PROGRAMA DE PROCESSING</p> <p>¿CUAL ES EL PROCESO QUE RESUELVE?: RESUELVE LOS CALCULOS USANDO CUALQUIER VALOR QUE SE LE A C, B O A Y LANZA SUS RAICES</p>
<b>DISEÑO DEL PROBLEMA</b>
<b>ENTIDAD QUE RESUELVE EL PROBLEMA:</b> PROGRAMA DE PROCESSING
<b>VARIABLES:</b>

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS</p> <p>FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	---	---

- ♥ **A, B Y C:** QUE REPRESENTAN LOS VALORES A REEMPLAZAR DENTRO DE LA ECUACION
- ♥ **DISCRIMINANTE:** CALCULA EL VALOR DEL DISCRIMINANTE PARA VER LA NATURALEZA DE LAS RAICES
- ♥ **RAIZ1, RAIZ2:** GUARDAN LOS VALORES DE LAS RAICES
- ♥ **RAIZREAL, RAIZ IMAGINARIA**

**NOMBRE DEL ALGORITMO:** TESORO\_LINK\_DISTANCIA

**PROCESO DEL ALGORITMO:**

- ♥ **INICIO**
- ♥ **LEE A, B Y C**
- ♥ **CALCULA EL DISCRIMINANTE**
- ♥ **EVALUA EL VALOR DEL DISCRIMINANTE**
- ♥ **SI ES POSITIVO CALCULARA DOS RAICES REALES**
- ♥ **SI ES CERO CALCULA UNA RAIZ DOBLE**
- ♥ **SI ES NEGATIVO CALCULA PARTES REALES E IMAGINARIAS DE ESTE**
- ♥ **FIN**

**EXERCICIO 19:** DECLARE LAS VARIABLES NECESARIAS PARA DIBUTAR UNA LÍNEA QUE SE DIBUJA DESDE LAS COORDENADAS INICIALES DEL LIENZO Y SE EXTIENDE POR TODO EL ANCHO. SOBRE EL PUNTO MEDIO DE LA LÍNEA Y A UNA DISTANCIA DE 40PX (EN SENTIDO VERTICAL DESDE LA LÍNEA) DIBUJE UNA ELIPSE QUE TENGA COMO ANCHO 80PX Y DE ALTO 80PX. DENTRO DE LA FUNCIÓN DRAW(), ACTUALICE LAS VARIABLES NECESARIAS PARA QUE LA LÍNEA DESDE SU INICIO SE MUEVA EN DIRECCIÓN HACIA ABAJO ARRASTRANDO LA ELIPSE. MANTENGA EN CERO EL VALOR PARA BACKGROUND(). CUANDO LA LÍNEA SUPERE LA POSICIÓN DE LA ALTURA DEL LIENZO, DEBE INVERTIR SU SENTIDO, ES DECIR DIRIGIRSE HACIA ARRIBA ARRASTRANDO LA ELIPSE. CUANDO LA LÍNEA ALCANCE NUEVAMENTE EL VALOR 0 PARA SU POSICIÓN EN Y, EL DESPLAZAMIENTO DEBE SER HACIA ABAJO Y ASÍ SUCESIVAMENTE. EL LIENZO DEBERÍA VERSE COMO EN LAS SIGUIENTES FIGURAS

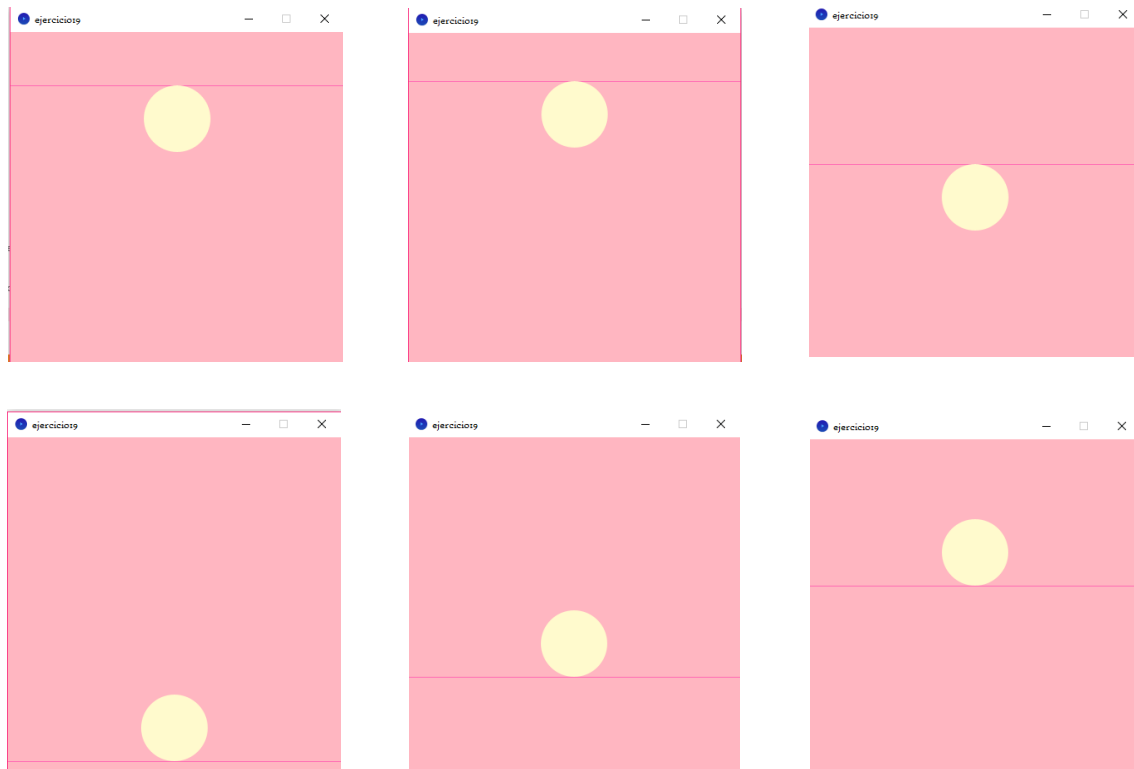
**PROCESSING:**

```

ejercicio19
1 float líneaY;
2 float ellipseOffset;
3 float velocidad;
4
5 void setup() {
6   size(400, 400);
7   líneaY = 0;
8   ellipseOffset = 40;
9   velocidad = 1;
10 }
11
12 void draw() {
13   background(FFB6C1);
14   stroke(FF69B4);
15   line(0, líneaY, width, líneaY);
16   noStroke();
17   fill(FFFACD);
18   ellipse(width/2, líneaY + ellipseOffset, 80, 80);
19   líneaY += velocidad;
20
21   // Invertir dirección y posición de la elipse cuando la línea alcanza los extremos
22   if (líneaY > height || líneaY < 0) {
23     velocidad *= -1;
24     ellipseOffset *= -1; // Cambia la posición de la elipse hacia arriba o abajo
25   }
26 }



```

### VENTANA DE VISUALIZACIÓN:



<b>ANÁLISIS DEL PROBLEMA</b>
<p><b>DATOS DE ENTRADA:</b> POSICION INICIAL DE LA LINEA Y EL ELPSE Y SU VELOCIDAD</p> <p><b>DATOS DE SALIDA:</b> LA ANIMACION DE LA LINEA Y EL ELPSE EN LA VENTANA DE VISUALIZACION</p> <p><b>PROCESO:</b></p> <p>¿QUIEN DEBE REALIZAR EL PROCESO?: PROCESSING</p> <p>¿CUAL ES EL PROCESO QUE RESUELVE?: CREAR UNA ANIMACIÓN DONDE UNA LÍNEA SE MUEVE HACIA ARRIBA Y HACIA ABAJO EN EL LIENZO, ARRASTRANDO UN CIRCULO QUE TAMBIÉN SE MUEVE VERTICALMENTE JUNTO CON ELLA.</p>
<b>DISEÑO DEL PROBLEMA</b>
<b>ENTIDAD QUE RESUELVE EL PROBLEMA:</b> PROCESSING
<p><b>VARIABLES:</b></p> <ul style="list-style-type: none"> <li>♥ <b>LINEA:</b> ALMACENA LA POSICIÓN VERTICAL ACTUAL DE LA LÍNEA EN EL LIENZO.</li> <li>♥ <b>ELIPSEOFFSET:</b> REPRESENTA EL DESPLAZAMIENTO VERTICAL DE LA ELPSE DESDE LA LÍNEA.</li> <li>♥ <b>VELOCIDAD:</b> INDICA LA DIRECCIÓN Y LA VELOCIDAD DE MOVIMIENTO DE LA LÍNEA EN EL LIENZO.</li> </ul>



	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

NOMBRE DEL ALGORITMO: PELOTTA\_SUBE\_BAJA

PROCESO DEL ALGORITMO:

- ♥ INICIO
- ♥ LEER LINEA Y DIRECCIÓN DE MOVIMIENTO
- ♥ DEFINIR DIMENSIONES DEL LIENZO
- ♥ ITERAR SOBRE EL LIENZO MOVIENDO LA LÍNEA
- ♥ SI LA LÍNEA ALCANZA LOS EXTREMOS, INVERTIR LA DIRECCIÓN
- ♥ MOSTRAR LA POSICIÓN DE LA LÍNEA
- ♥ DIBUJAR LA LÍNEA Y LA ELIPSE EN EL LIENZO
- ♥ FIN

**EXERCICIO 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 PÍXELES ENTRE ELLOS TANTO HORIZONTAL COMO VERTICALMENTE. UTILICE LA ESTRUCTURA DE CONTROL REPETITIVA FOR. EL LIENZO DEBERÍA VERSE ASÍ:

DATOS DE ENTRADA:

- ♥ TAMAÑO DEL LIENZO: 440 P. DE ANCHO Y 420 P. DE ALTO.
- ♥ MEDIDAS DEL RECTÁNGULO: 40 P. DE ANCHO Y 20 P. DE ALTO.
- ♥ DISTANCIA ENTRE RECTÁNGULOS: 20 P. (HORIZONTAL Y VERTICAL)

DATOS DE SALIDA

VARIOS RECTÁNGULOS A LO LARGO DEL LIENZO

¿QUIÉN DEBE REALIZAR EL PROCESO?:

ESTE PROCESO PUEDE SER REALIZADO POR PROCESSING

¿CUÁL ES EL PROCESO QUE RESUELVE?:

EL PROCESO RESUELVE LA TAREA DE DIBUJAR UNA CUADRÍCULA DE RECTÁNGULOS EN TODO EL LIENZO, HACIENDO QUE CADA RECTÁNGULO TENGA LAS MEDIDAS EXACTAS Y QUE HAYA UNA DISTANCIA DE 20 PÍXELES ENTRE ELLOS

ENTIDAD QUE RESUELVE EL PROBLEMA:

EL PROBLEMA ES RESUELTO POR EL PROGRAMA PROCESSING

DISEÑO DEL PROBLEMA

ENTIDAD QUE RESUELVE EL PROBLEMA:

NOMBRE DEL ALGORITMO: RECTÁNGULOS\_REPETIDOS

### VARIABLES:

- ♥ PVECTOR COORDENADAS: ALMACENA LAS COORDENAS X E Y DE LA POSICIÓN DE LOS RECTÁNGULOS
- ♥ INTALTOREC, ANCHOREC, DISTREC: ESTAS TRES VARIABLES DETERMINAN EL ALTO, ANCHO DE LOS RECTÁNGULOS Y LA DISTANCIA ENTRE ELLOS
- ♥ SIZE: ESTABLECE LAS MEDIDAS DE LA VENTANA DE VISUALIZACIÓN
- ♥ BACKGROUND, FILL, STROKE: EL PRIMERO DETERMINA EL COLOR DE FONDO DE LA VENTANA, EL SEGUNDO EL COLOR DE LOS RECTÁNGULOS Y EL ULTIMO EL COLOR DEL CONTORNO
- ♥ FOR: SIRVE COMO BUCLE PARA REPETIR LOS RECTÁNGULOS

### PROCESO:

- ♥ **INICIO**
- ♥ ESTABLECER EL TAMAÑO DEL LIENZO EN 440X420 P.
- ♥ DEFINIR LAS MEDIDAS DEL RECTÁNGULO: 40 P. DE ANCHO Y 20 P. DE ALTO.
- ♥ ESTABLECER LA DISTANCIA ENTRE LOS RECTÁNGULOS EN 20 P. (HORIZONTAL Y VERTICAL)
- ♥ UTILIZAR UNA EL CÓDIGO REPETITIVO FOR PARA DIBUJAR UNA CUADRÍCULA DE RECTÁNGULOS EN TODO EL LIENZO
- ♥ **FIN**

### PROCESING:

```

ejercicio20
1 PVector coordenadas;
2 int altoRec, anchoRec, distRec;
3
4 void setup(){
5   size(440,420);
6   distRec = 20;
7   anchoRec= 40;
8   altoRec= 20;
9   coordenadas= new PVector(distRec,distRec);
10 }
11
12 void draw(){
13   background(#FFB6C1);
14   fill(#FFFACD);
15   stroke(#FF69B4);
16   dibujarRec();
17 }
18
19 void dibujarRec(){
20   for(float x=coordenadas.x;x<width;x+=(anchoRec+distRec)){
21     for(float y=coordenadas.y;y<height;y+=(altoRec+distRec)){
22       rect(x,y,anchoRec,altoRec);
23     }
24   }
25 }

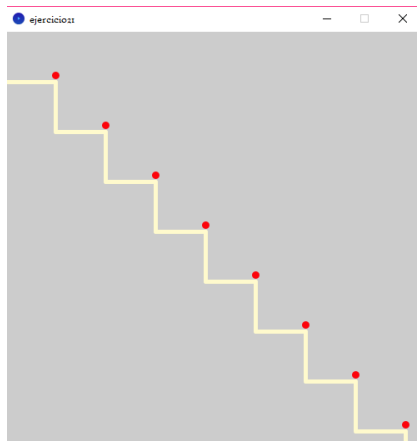
```

### VENTANA DE VISUALIZACIÓN:



**EXERCICIO 20:** 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

### VISUALIZACIÓN DE LA VENTANA



### PROCESSING:



### DATOS DE ENTRADA:

DISTANCIA ENTRE LOS PUNTOS

### DATOS DE SALIDA

ENTREGA EL GRAFICO DE ESCALERAS Y PUNTOS ROJOS



### PROCESO:

¿QUIÉN DEBE REALIZAR EL PROCESO?: POR PROCESSING

¿CUÁL ES EL PROCESO QUE RESUELVE?: CREA UNA REPRESENTACIÓN VISUAL DE LA ESCALERA Y LOS PUNTOS ROJOS

### ENTIDAD QUE RESUELVE EL PROBLEMA:

EL PROBLEMA ES RESUELTO POR EL PROGRAMA PROCESSING

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS</p> <p>FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	---	---

<p><b>DISÑO DEL PROBLEMA</b></p>
<p>ENTIDAD QUE RESUELVE EL PROBLEMA: PROCESSING</p>
<p>NOMBRE DEL ALGORITMO: ESCALERA_PUNTITOS</p>
<p><b>VARIABLES:</b></p> <ul style="list-style-type: none"> <li>♥ <b>DISTANCIA:</b> UN ENTERO QUE REPRESENTA LA DISTANCIA ENTRE LOS PUNTOS EN LA ESCALERA.</li> <li>♥ <b>PUNTOA, PUNTOB, PUNTOC, PUNTOD:</b> OBJETOS PVECTOR QUE REPRESENTAN DIFERENTES PUNTOS EN EL LIENZO.</li> </ul>
<p><b>PROCESO:</b></p> <ul style="list-style-type: none"> <li>♥ <b>INICIO</b></li> <li>♥ <b>CONFIGURACIÓN INICIAL</b></li> <li>♥ <b>BUCLE DE LA DISTANCIA</b></li> <li>♥ <b>GRAFICAR LA ESCALERA</b></li> <li>♥ <b>GRAFICAR LA ELIPSE</b></li> <li>♥ <b>SE ACTUALIZA EL PUNTO DE INICIO</b></li> <li>♥ <b>REPETICION DEL INICIO</b></li> <li>♥ <b>SE GRAFICA LA ESCALERA CON LAS MEDIDAS FINALES</b></li> <li>♥ <b>LANZA EL GRAFICO</b></li> <li>♥ <b>FIN</b></li> </ul>



**EXERCICIO 21:** UTILIZANDO LA ESTRUCTURA DE CONTROL REPETITIVA DO-WHILE. REPLIQUE LA SIGUIENTE IMAGEN. LA IMAGEN DEBE SER CONSTRUIDA DESDE LA FUNCIÓN SETUP(). DEFINA EL TAMAÑO DEL LIENZO EN SIZE(600,600). VERTICALMENTE SE DIVIDE EL LIENZO EN FRANTAS DE IGUAL MEDIDA, SE DEBEN DIBUJAR LOS CÍRCULOS SOBRE CADA LÍNEA DE POR MEDIO ES DECIR EN LA LÍNEA 1 SE DIBUJAN CÍRCULOS CON DISTANCIAMIENTO, EN LA LÍNEA 2 NO SE DIBUJA Y ASÍ SUCEATIVAMENTE. LAS LÍNEAS TIENEN UN COLOR FIJO, LOS CÍRCULOS ASUMEN COLORES ALEATORIOS.

**PROCESSING:**

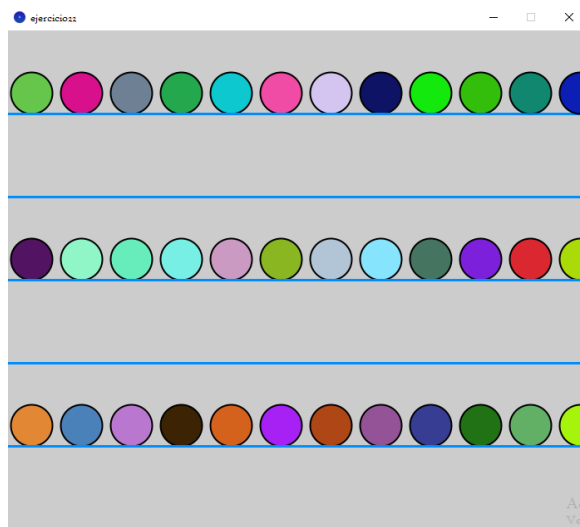
```

ejercicio22
4  int lineaY=100;
5  int circuloY=75;
6  int distanciaCirculo=30;
7  do{
8    int circuloX=distanciaCirculo;
9    do{
10     stroke(#0080FC);
11     line(lineaX,lineaY,width,lineaY);
12     fill(random(255),random(255),random(255));
13     stroke(0);
14     strokeWeight(2);
15     ellipse(circuloX,circuloY,50,50);
16     circuloX+=distanciaCirculo;
17   }while(circuloX<width);
18   lineaY+=100;
19   circuloY+=200;
20   }while(lineaY<height);
21 }

```

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

### VENTANA DE VISUALIZACIÓN:



### ANÁLISIS DEL PROBLEMA

**DATOS DE ENTRADA:** DATOS DE DISTANCIA Y ESPECIFICACIONES DEL GRAFICO

**DATOS DE SALIDA:** GRAFICO FINAL CON COLORES ALEATORIOS

**PROCESO:**

¿QUIEN DEBE REALIZAR EL PROCESO?: PROGRAMA DE PROCESSING

¿CUAL ES EL PROCESO QUE RESUELVE?: RESUELVE EL PLANTEAMIENTO DE UNA IMAGEN ESPECIFICA CON DATOS EXACTOS DE UBICACIÓN Y TAMAÑO

### DISEÑO DEL PROBLEMA

ENTIDAD QUE RESUELVE EL PROBLEMA: PROGRAMA DE PROCESSING



**VARIABLES:**

- ♥ **LINEAX:** CONTROLA LA POSICIÓN INICIAL EN X DE LAS LÍNEAS.
- ♥ **LINEAY:** CONTROLA LA POSICIÓN INICIAL EN Y DE LAS LÍNEAS.
- ♥ **CIRCULOY:** CONTROLA LA POSICIÓN EN Y DE LOS CÍRCULOS.
- ♥ **DISTANCIACIRCULO:** CONTROLA LA DISTANCIA ENTRE CADA CÍRCULO EN EL EJE X.
- ♥ **CIRCULOX:** CONTROLA LA POSICIÓN EN X DE LOS CÍRCULOS DENTRO DEL BUCLE INTERNO.

**NOMBRE DEL ALGORITMO:** REPETICION CIRCULOS

**PROCESO DEL ALGORITMO:**

- ♥ **INICIO**
- ♥ **CONFIGURAR EL LIENZO.**
- ♥ **INICIALIZAR LAS VARIABLES.**
- ♥ **DESBURJAR LÍNEAS HORIZONTALES Y CÍRCULOS EN PATRÓN.**
- ♥ **REPETIR EL PASO ANTERIOR HASTA LLENAR EL LIENZO DE PELOTTAS**
- ♥ **RECIBIR EL GRAFICO**
- ♥ **FIN**

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS  TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS  <b>FACULTAD DE INGENIERÍA</b>  Universidad Nacional de Jujuy</p> <p>Trabajo Práctico 01: Operadores - Metodología de programación</p>	
---	--	---

#### REFERENCIAS:

♥ VIDEOS DEL AULA

♥ CURSOS DE YOUTUBE:

[HTTPS://WWW.YOUTUBE.COM/WATCH?V=60R2JKNMqIE&LIST=PLTYMMY0EKYQFSLPESM27Y4EZK2FJRGUJ](https://www.youtube.com/watch?v=60R2JKNMqIE&list=PLTYMMY0EKYQFSLPESM27Y4EZK2FJRGUJ)