



TECNICATURA UNIVERSITARIA EN DISEÑO
INTEGRAL DE VIDEOJUEGOS

FACULTAD DE INGENIERÍA
Universidad Nacional de Jujuy



FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS

Trabajo Práctico/Actividad

N°

Apellido y Nombre – LU /

Tarifa, Juan Mateo-



TUV000732

Profesores:

Mg. Ing. Ariel Alejandro Vega

Ing. Carolina Cecilia Apaza

Año

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy PENSAMIENTO COMPUTACIONAL y PROGRAMACIÓN: Problema y Solución – PC y P – Algoritmos – Principio de la P</p>	
---	--	---

Indice

Ejercicio 1: Evaluar (obtener resultado) la siguiente expresión para A = 2 y B = 5

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

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

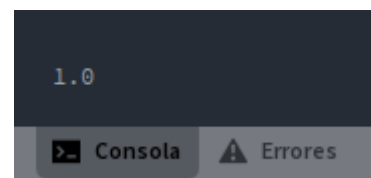
$$6 - (20/4)$$

$$6 - 5$$

$$1$$

Captura de Processing / Consola

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

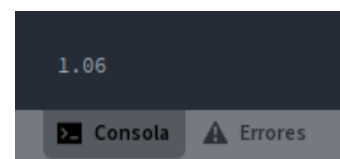


Ejercicio 2: Evaluar la siguiente expresión $4 / 2 * 3 / 6 + 6 / 2 / 1 / 5^2 / 4 * 2$

ALGEBRAICA	ARITMÉTICA
$\frac{(\frac{4}{2} \cdot 3)}{6} + (\frac{\frac{6}{2}}{\frac{1}{5^2}}) * 2$	$(((4/2)*3)/6) + (((6/2)/1)/5^2)/4 * 2$

Captura de Processing / Consola

```
1 float resultado= (((4/2)*3)/6)+(((6/2)/1)/pow(5,2)/4)*2;
2 println(resultado);
3
```



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) $b^2 - 4 * a * c$ $a=5, b=3, c=1$

ALGEBRAICA	ARITMÉTICA
$b^2 - 4ac$ $3^2 - 4 \cdot 5 \cdot 1$ $9 - 20$ -11	$b^2 - 4 * a * c$ $(3^2) - ((4*5)*1)$ $9 - 20$ -11

Captura de Processing / Consola

```
1 int a=5,b=3,c=1;
2 float resultado= (pow(b,2))-((4*a)*c);
3 println(resultado);
```

```
-11.0
Consola Errores
```

B) $3 \cdot X^4 - 5 \cdot X^3 + X^{12} - 17$ $X=2$

ALGEBRAICA	ARITMÉTICA
$3 \cdot 2^4 - 5 \cdot 2^3 + 2 \cdot 12 - 17$	$(3 \cdot (2^4)) - (5 \cdot (2^3)) + (2 \cdot 12) - 17$
$3 \cdot 16 - 5 \cdot 8 + 24 - 17$	$48 - 40 + 24 - 17$
$48 - 40 + 24 - 17$	15
15	

Captura de Processing / Consola

```
5 int x=2;
6 float resultado2= (3*(pow(x,4)))-(5*(pow(x,3)))+(x*12)-17;
7 println(resultado2);
```

```
15.0
Consola Errores
```

C) $(b + d) / (c + 4)$ $b=4, d=5, c=2$

ALGEBRAICA	ARITMÉTICA
$\frac{(b + d)}{(c + 4)}$	$(b + d) / (c + 4)$
	$(4 + 5) / (2 + 4)$
	$9 / 6$
$\frac{(4 + 5)}{(2 + 4)} = 1.5$	1.5

Captura de Processing / Consola

```
9 int b3=4,d=5,c3=2;
10 float resultado3= (b3+d)/ float(c3+4);
11 println(resultado3);
```

```
1.5
Consola Errores
```

D) $(x^2 + y^2)^{1/2}$ $x=4, y=2$

ALGEBRAICA	ARITMÉTICA
$(x^2 + y^2)^{\frac{1}{2}}$	$(x^2 + y^2)^{(1/2)}$
$(4^2 + 2^2)^{\frac{1}{2}}$	$(4^2 + 2^2)^{(1/2)}$
$(20)^{\frac{1}{2}}$	$20^{(1/2)}$
$(20)^{\frac{1}{2}} = 4.47$	4.47

Captura de Processing / Consola

```
13 int x4=4,y=2;
14 float resultado4= pow(pow(x4,2)+pow(y,2), 0.5);
15 println(resultado4);
```

```
4.472136
> Consola  Errores
```

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

ALGEBRAICA	ARITMÉTICA
$5 \cdot 4 - \frac{5^2}{4} \cdot 1$ $20 - 6.25 = 13.75$	$B * A - B^2 / 4 * C$ $5*4 - ((5^2)/4) * C$ $20 - 6.25$ 13.75

Captura de Processing / Consola

```
1 int A=4,B=5,C=1;
2 float resultado= B*A-((pow(B,2))/4);
3 println(resultado);
```

```
13.75
> Consola  Errores
```

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

ALGEBRAICA	ARITMÉTICA
$\frac{(A \cdot B)}{3^2} = \frac{4 \cdot 5}{3^2} = 2.22 \dots$	$(A * B) / 3^2$ $(4*5)/3^2$ $2.22\dots$

Captura de Processing / Consola

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

```
2.2222223
> Consola  Errores
```

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

ALGEBRAICA	ARITMÉTICA
$\left(\left(\frac{B+C}{2} \cdot A + 10 \right) \cdot 3 \cdot B \right) - 6$ $\left(\left(\frac{5+1}{2} \cdot 4 + 10 \right) \cdot 3 \cdot 5 \right) - 6$	$(((B + C) / 2 * A + 10) * 3 * B) - 6$ $(((5+1)/2*4+10)*3*5) - 6$ $((6/2*4+10)*3*5) - 6$ $((3*4+10)*3*5) - 6$ $((22*3*5) - 6$ $330 - 6$

$$(22 \cdot 3 \cdot 5) - 6 = 324$$

324

Captura de Processing / Consola

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

324.0

Consola Errores

Ejercicio 6: Para $x=3$, $y=4$; $z=1$, evaluar el resultado de

$R1 = y+z$

$R2 = x \geq R1$

$$R1 = 4 + 1$$

$$R2 = 3 \geq R1$$

Falso

Captura de Processing / Consola

```
1 int x=3, y=4, z=1;
2 int R1= y+z;
3 boolean R2=x>=R1;
4 println(R2);
```

false

Consola Errores

Ejercicio 7: Para contador1=3, contador3=4, evaluar el resultado de

$R1 = ++\text{contador1}$

$R2 = \text{contador1} < \text{contador2}$

$$R1 = ++3$$

$$R2 = 4 < 4$$

Falso

Captura de Processing / Consola

```
1 int contador1=3, contador3=4;
2 int R1= ++contador1;
3 boolean R2=contador1<contador3;
4 println(R2);
```

false



Consola Errores

Ejercicio 8: Para $a=31$, $b=-1$; $x=3$, $y=2$, evaluar el resultado de $a+b-1 < x*y$

$$31 + (-1) - 1 < 3 \cdot 2$$

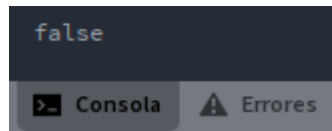
$$29 < 6$$

Falso

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy Trabajo Practico N° 1: Operadores - Metodología de programación</p>	
---	---	---

Captura de Processing / Consola

```
1 int a=31, x=3, y=2, b=-1;
2 boolean resultado= a+b-1 < 3*2;
3 print(resultado);
```



Ejercicio 9: Para x=6, y=8, evaluar el resultado de
!(x<5)CC !(y>=7)

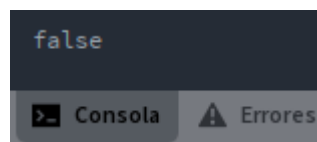
$!(6 < 5) \&\& !(8 \geq 7)$

Falso && Verdadero

Falso

Captura de Processing / Consola

```
1 int x=6, y=8;
2 boolean resultado= !(x<5) && !(y>=7);
3 println(resultado);
```



Ejercicio 10: Para i=22, j=3, evaluar el resultado de

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

$!((22 > 4) || !(3 \leq 6))$

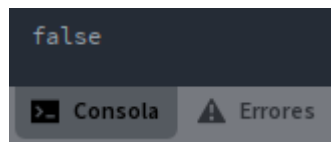
!(Verdadero || Falso)

!(Verdadero)

Falso

Captura de Processing / Consola

```
1 int i=22, j=3;
2 boolean resultado= !((i>4) || (j<=6));
3 println(resultado);
```



Ejercicio 11: Para a=34, b=12,c=8, evaluar el resultado de

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

!(34 + 12 == 8)|| (8! = 0)&&(12 – 8 ≥ 19)

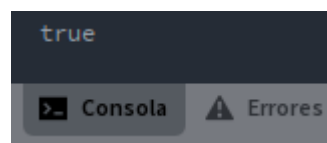
Verdadero||Verdadero&&Falso

Verdadero||Falso

Verdadero

Captura de Processing / Consola

```
1 int a=34, b=12, c=8;
2 boolean resultado= !(a+b==c) || (c!=0)&&(b-c>=19);
3 println(resultado);
```



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:

Datos de Entrada: nombre

Datos de Salida: saludo en pantalla

Proceso: ¿Quién lo realiza? Algoritmo

¿Qué proceso realiza? Se ingresa el nombre, luego el programa lee ese nombre y devuelve el saludo con el nombre ingresado y se muestra en el lienzo.

Diseño:

Entidad que resuelve el problema: Algoritmo
Variables usuario: string //nombre proporcionado saludo: string //saludo que devuelve la consola
Nombre de algoritmo: saludo_usuario
<ul style="list-style-type: none"> Inicio Leer usuario saludo←"Hola"+usuario+"bienvenido" Mostrar saludo fin

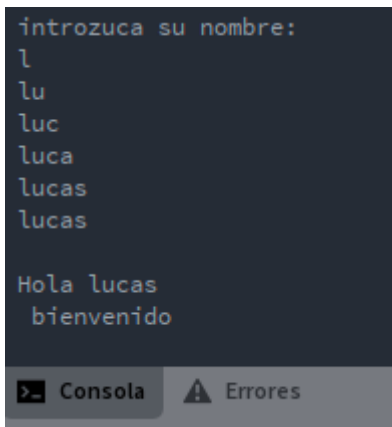
Captura de Processing


```

1 String ingreso="introduzca su nombre:",usuario="",saludo="";
2
3 void setup(){
4   size(500,500);
5   println(ingreso);
6 }
7
8 void draw(){
9   background(0);
10  text(saludo,125,height/2);
11  textSize(30);
12 }
13
14 void keyPressed(){
15   usuario += key;
16   println(usuario);
17
18   if(key == '\n'){
19     saludo="Hola "+usuario+" bienvenido";
20     println(saludo);
21   }
22 }

```

Consola / Lienzo

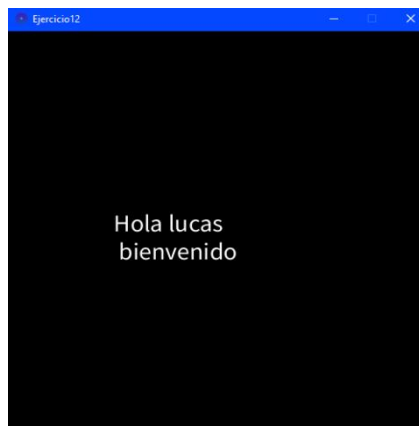


```

introduzca su nombre:
l
lu
luc
luca
lucas
lucas

Hola lucas
bienvenido

```



Ejercicio 13: Será común resolver problemas utilizando variables. Calcule el perímetro y área de un rectángulo dada su base y su altura.

Análisis:

Datos de Entrada: Base, altura

Datos de Salida: perímetro, área



Proceso: ¿Quién realiza el proceso? El usuario

¿Qué proceso realiza? Dados la base y la altura, realiza el cálculo del perímetro y luego del área del rectángulo usando la formulas

$$P: 2(Base \cdot Altura) \mid A: Base \cdot Altura$$

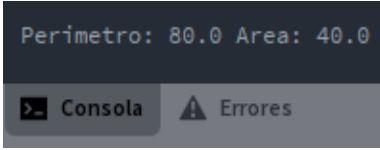
Diseño:

Entidad que resuelve el problema: Usuario
Variables base: float altura: float perimetro: float

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy Trabajo Practico N° 1: Operadores - Metodología de programación</p>	
---	---	---

<p>area: float perimetroArea: string</p>
<p>Nombre de algoritmo: resultado_perimetroArea</p>
<ul style="list-style-type: none"> • inicio • leer base • leer altura • $perimetro \leftarrow 2 * (b * a)$ • $area \leftarrow b * a$ • $perimetroArea \leftarrow \text{"Perimetro: " + perimetro + " Area: " + area}$ • mostrar perimetroArea • fin

Captura de Processing / Consola

<pre> 1 float base=8, altura=5; 2 3 float perimetro= 2*(base*altura); 4 float area= base*altura; 5 String perimetroArea= "Perimetro: "+perimetro+" Area: "+area; 6 println(perimetroArea); </pre>	
---	--

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.

Análisis:

Datos de entrada: Cateto adyacente, cateto opuesto



Datos de salida: Hipotenusa

Proceso: ¿Quién realiza el proceso? El usuario

¿Qué proceso realiza? Teniendo los datos de los catetos, se procede a encontrar el resultado de la hipotenusa utilizando la formula. $h = \sqrt{(a^2 + b^2)}$

Diseño:

<p>Entidad que resuelve el problema: Usuario</p>
<p>Variables: catetoOp: int catetoAdy: int hipotenusa: float raiz: float</p>
<p>Nombre del algoritmo: calcular_hipotenusa</p>
<ul style="list-style-type: none"> • inicio • Leer catetoOp • Leer catetoAdy • $hipotenusa \leftarrow (\text{catetoOp}^2 + \text{catetoAdy}^2)$ • $raiz \leftarrow \sqrt{hipotenusa}$

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy Trabajo Practico N° 1: Operadores - Metodología de programación</p>	
---	---	---

- *mostrar raiz*
- *fin*

Captura de Processing / Consola

```

1 int catetoOp=15, catetoAdy=10;
2
3 float hipotenusa= pow(15,2)+pow(10,2);
4 float raiz= sqrt(hipotenusa);
5 println(raiz);

```

18.027756

Consola Errores

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.

Análisis:

Datos de entrada: numC, numD

Datos de salida: suma, resta, multiplicación, división

Proceso: ¿Quién realiza el proceso? Calculadora o usuario

¿Qué proceso realiza? Identifica los 2 números y luego comienza con las operaciones siguiendo un orden.

Diseño:

Entidad que resuelve el problema: Calculadora

Variables:

numC= int
numD= int
suma= int
resta= int
multip= int
div= int

Nombre de algoritmo: calcular_operaciones

- inicio
- Leer numC
- Leer numD
- $\text{suma} \leftarrow \text{numC} + \text{numD}$
- $\text{mostrar} \leftarrow \text{"Suma: " + suma}$
- $\text{resta} \leftarrow \text{numC} - \text{numD}$
- $\text{mostar} \leftarrow \text{"Resta: " + resta}$
- $\text{multip} \leftarrow \text{numC} * \text{numD}$
- $\text{mostrar} \leftarrow \text{"Multiplicación: " + multip}$
- $\text{div} \leftarrow \text{numC} / \text{numD}$
- $\text{mostrar} \leftarrow \text{"División: " + div}$
- fin

Captura de Processing / Consola

```

1 int numC=80, numD=6;
2
3 int suma= numC + numD;
4 println("Suma: " + suma);
5 int resta= numC - numD;
6 println("Resta: " + resta);
7 int multip= numC * numD;
8 println("Multiplicación: " + multip);
9 int div= numC / numD;
10 println("División: " + div);

```

```

Suma: 86
Resta: 74
Multiplicación: 480
División: 13

```

Consola Errores

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.

Análisis:

Datos de entrada: Temperatura en Fahrenheit

Datos de salida: Temperatura en Celsius

Proceso: ¿Quién realiza el proceso? La calculadora

¿Qué proceso realiza? Se toma la temperatura en grados Fahrenheit, luego se la convierte a grados Celsius a través de su fórmula correspondiente $C = (F - 32) \cdot \frac{5}{9}$

Diseño:

Entidad que resuelve el problema: Calculadora
Variables: tempFahr: float tempCel: float
Nombre de algoritmo: convertir_temperatura
<ul style="list-style-type: none"> inicio Leer tempFahr tempCel ← (tempFahr - 32) * (5.0/9.0) mostrar tempCel fin

Captura de Processing / Consola

```

1 float tempFahr= 212;
2 float tempCel= (tempFahr -32) * 5.0/9.0;
3 println("La temperatura en celsius es: " + tempCel);

```

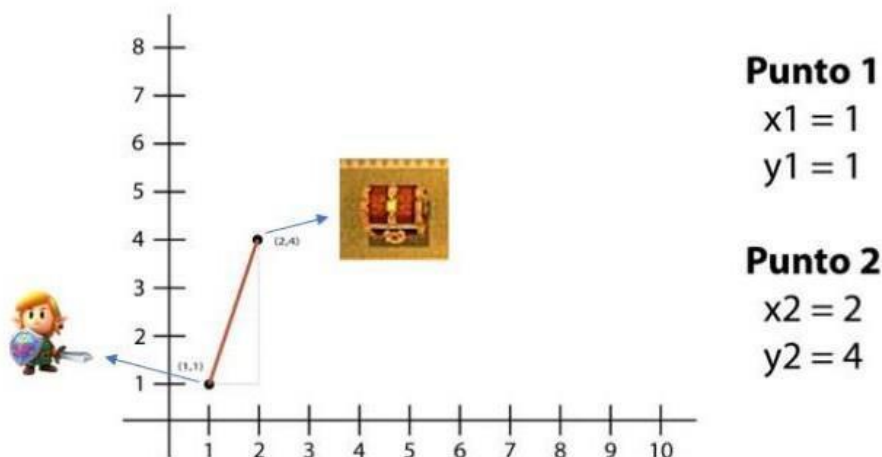
```

La temperatura en celsius es: 100.0

```

Consola Errores

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 Circulo, y al tesoro con un cuadrado. Además, mueva a Link mediante el mouse.

Análisis:

Datos de entrada: Coordenadas Link, coordenadas cofre



Datos de salida: Distancia entre link y cofre

Proceso: ¿Quién lo realiza? El programa

¿Qué proceso realiza? Calculamos las diferencias entre las X e Y, obtenemos los catetos para lleva a cabo el teorema de Pitágoras $h = \sqrt{((x2 - x1) + (y2 - y1))}$

Diseño:

Entidad que resuelve el problema: Programa
Variables: $x1, y1, x2, y2 = \text{float}$ $\text{caja} = \text{float}$ $\text{coordX} = \text{float}$ $\text{coordY} = \text{float}$ $\text{dist} = \text{float}$
Nombre del algoritmo: distancia_linkCofre
<ul style="list-style-type: none"> Inicio Leer x1

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy Trabajo Practico N° 1: Operadores - Metodología de programación</p>	
---	---	---

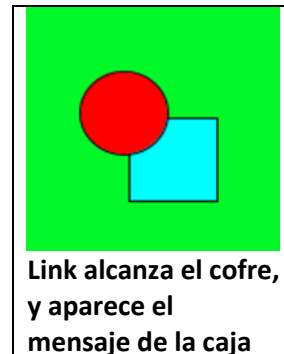
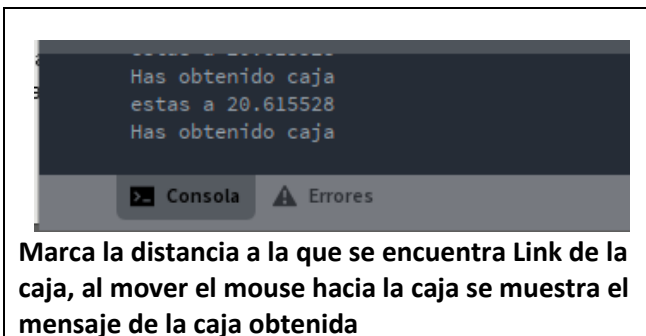
- Leer y1
- Leer x2
- Leer y2
- caja←25
- coordX← x2 – x1
- coordY← y2 – y1
- $dist \leftarrow \sqrt{(coordX^2) + (coordY^2)}$
- mostrar "estas a "+ dist
- **si** (dist <= caja) **entonces**
- mostrar "Has obtenido caja"
- *fin_si*
- *fin*

Captura de Processing

```

1  float x1=100, y1=100, x2=200,y2=400;
2  float caja= 25;
3
4  void setup(){
5      size(600,600);
6  }
7
8  void draw(){
9      background(#42F707);
10     float coordX= x2 - x1;
11     float coordY= y2 - y1;
12
13     float dist= sqrt(pow(coordX,2)+ pow(coordY,2));
14     String texto= "estas a " + dist;
15     println(texto);
16     if(dist <=caja){
17         println("Has obtenido caja");
18     }
19     fill(0,255,255);
20     rect(x2,y2,50,50);
21     fill(255,0,0);
22     ellipse(mouseX,mouseY,50,50);
23 }
24
25 void mouseMoved()
26     x1=mouseX;
27     y1=mouseY;
28 }
```

Consola / Lienzo



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.

Análisis:

Datos de entrada: a, b, c

Datos de salida: Raíces

Proceso: ¿Quién realiza el proceso? El usuario o calculadora

¿Qué proceso realiza? Asigna los valores a las letras, luego resuelve el discriminante y luego

procede a buscar las raíces $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

Diseño:

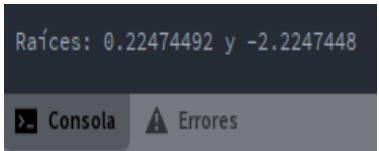
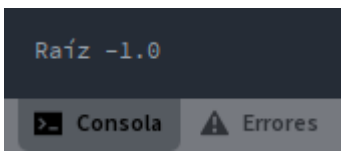
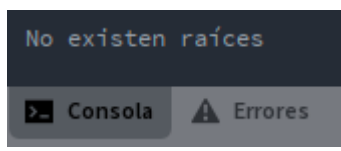
Entidad que resuelve el algoritmo: Usuario
Variables: a= float b= float c= float discriminante= float
Nombre del algoritmo: raíces_ecuacion
<ul style="list-style-type: none"> Inicio Leer a Leer b Leer c discriminante ← $b^2 - 4 * a * c$ si (discriminante > 0) entonces $x1 \leftarrow (-b + (\sqrt{\text{discriminante}})) / 2 * a$ $x2 \leftarrow (-b - (\sqrt{\text{discriminante}})) / 2 * a$ mostrar ← "Raíces: " + x1 + " y " + x2 si_no si (discriminante == 0) entonces $x \leftarrow -b / (2 * a)$ mostrar ← "Raíz: " + x si_no mostrar ← "No existen raíces" fin

Captura de Processing / Consola

```



1 float a=2,b=4,c=-1;
2
3 void setup(){
4     float discriminante= pow(b,2)-4*a*c;
5
6     if(discriminante>0){
7         //raíces distintas
8         float x1= (-b+sqrt(discriminante))/(2*a);
9         float x2= (-b-sqrt(discriminante))/(2*a);
10        println("Raíces: "+ x1+" y "+ x2);
11    }
12    else if (discriminante==0){
13        //raíces iguales
14        float x=-b / (2*a);
15        println("Raíz"+x);
16    }
17    else{
18        //Sin raíces
19        println("No existen raíces");
20    }
21 }

```

2 RAICES (2,4,-1)	1 RAIZ(2,4,2)	NINGUNA RAIZ(1,1,1)
		

Ejercicio 19: Declare las variables necesarias para dibujar 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.



	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy Trabajo Practico N° 1: Operadores - Metodología de programación</p>	
---	---	---

Análisis:

Datos de entrada: distancia del punto, ancho y alto del punto

Datos de salida: línea y círculo en bucle

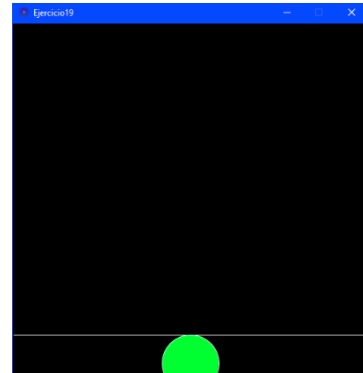
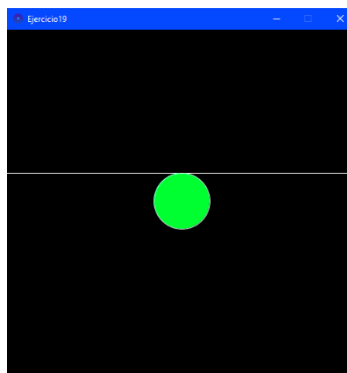
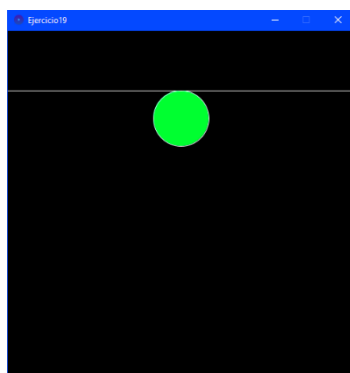
Proceso: ¿Quién realiza el proceso? El programa

¿Qué proceso realiza? Procesa los datos sobre la línea y el punto, luego grafica ambos de manera constante a través de un bucle

Diseño:

Entidad que resuelve el algoritmo: Lienzo
Variables: linea= int circ= int
Nombre del algoritmo: bucle_lineaCirculo
<ul style="list-style-type: none"> • Inicio • Leer linea • Leer circ • anchoLienzo ← 500 • altoLienzo ← 500 • para x ← 0 hasta x < 1 con paso incremento 1 hacer • linea ← linea + circ • fin_para • si linea ≥ altoLienzo O linea ≤ 0 entonces • circ ← circ * (-1) • fin_si • mostrar linea • dibujar linea en (circ, linea, anchoLienzo, lienzo) • dibujar círculo en (anchoLienzo/2, linea+40, 80, 80)

Lienzo



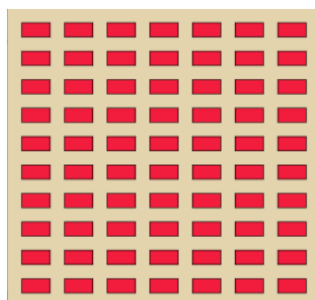
Captura de Processing

```

1  int linea;
2  int circ=1;
3
4  void setup(){
5      size(500,500);
6      linea = 250;
7  }
8
9  void draw(){
10     background(0);
11     for (int x =0; x<1; x++){
12         linea= linea+ circ;
13     }
14     if (linea>= height || linea <=0){
15         circ= circ *-1;
16     }
17     print(linea);
18     stroke(255);
19     fill(0,255,0);
20     line(circ, linea, width, linea);
21     ellipse(width/2, linea+40, 80, 80);
22 }

```

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:



Análisis:

Datos de entrada: El ancho y alto del lienzo.

Los valores del alto, ancho del rectángulo y distancia entre los rectángulos.

El dibujo de los rectángulos en el lienzo.

Datos de salida: Los rectángulos dibujados en el lienzo

Proceso: ¿Quién realiza el proceso? Un programa, en este caso, processing

¿Qué proceso realiza? Dibuja rectángulos en el lienzo con las medidas que se especifican (teniendo en cuenta su ancho, alto y distancia entre cada uno), hacemos uso de un bucle para lograr la tarea.

Diseño:

Entidad que resuelve el algoritmo: Programa
Variables: anchoRect, altoRect, distEntreRect: entero altoLienzo, anchoLienzo: entero coordenadas: float //almacenan coordenadas en x,y

Nombre del algoritmo: dibujarRec

- Inicio
- anchoLienzo \leftarrow 440
- altoLienzo \leftarrow 420
- distRec \leftarrow 20
- anchoRec \leftarrow 40
- altoRec \leftarrow 20
- **para** x=coordenadas.x **hasta** anchoLienzo **con paso** (anchoRec+distRec)
- **hacer**
- **para** y=coordenadas.y **hasta** altoLienzo **con paso** (altoRec+distRec)
- **hacer**
- dibujar rectángulo en (x,y,anchoRec,altoRec)
- fin_para
- fin_para
- fin

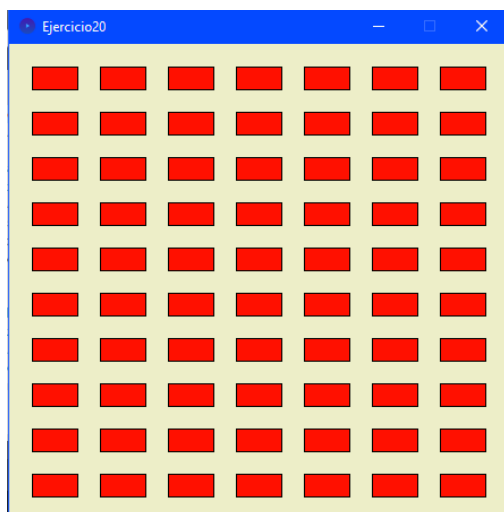
Captura de Processing

```

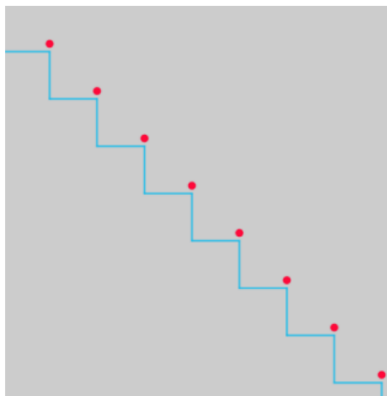
1  PVector coordenadas;
2  int altoRec, anchoRec, distRec;
3
4  void setup(){
5      size(440,420);
6      distRec = 20;
7      anchoRec= 40;
8      altoRec= 20;
9      coordenadas= new PVector(distRec,distRec);
10 }
11
12 void draw(){
13     background(#F2EECB);
14     fill(#C11010);
15     stroke(#030302);
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 }

```

Lienzo



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.



El tamaño del lienzo es size(500,500). La estructura while() se ejecuta dentro de la función setup(). La condición es que solo se dibuje dentro del lienzo. Utilice variables que puedan ayudar a la construcción del dibujo, por ej: x, y, anchoEscalon, altoEscalon, etc.

Análisis:

Datos de entrada: puntoA, puntoB, puntoC, puntoD, distancia, alto y ancho del lienzo

Datos de salida: Escalones con puntos en los bordes, mostrado en el lienzo

Proceso: ¿Quién realiza el proceso? El programa

¿Qué proceso resuelve? Mediante el bucle while() repite el proceso de dibujar escalones y puntos rojos en sus bordes

Diseño:

Entidad que resuelve el algoritmo: Programa
Variables: pA, pB, pC, pD: int //vectores dist= int
Nombre del algoritmo: escalones_puntos
<ul style="list-style-type: none"> Inicio anchoLienzo←500 altoLienzo← 500 dist← 60 mientras (pA <= anchoLienzo) hacer dibujar línea horizontal en (pA.x, pA.y, pB.x, pB.y) dibujar línea vertical en (B.x, pB.y, pC.x, pC.y) dibujar circulo en (pD.x, pD.y) pA.x ← pC.x pA.y ← pC.y fin_mientras fin

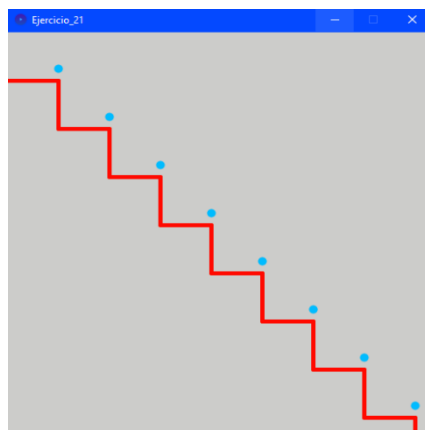
Captura de Processing

```

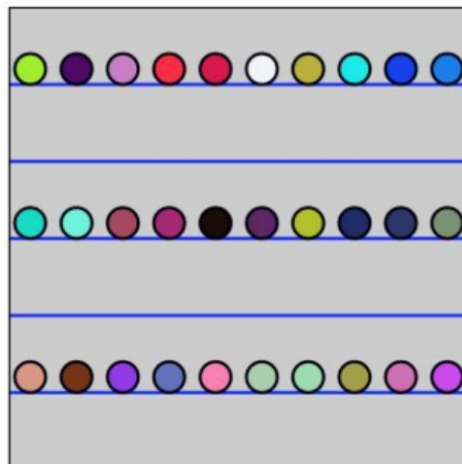
1  int dist;
2  PVector pA, pB, pC, pD;
3
4  void setup(){
5      size(500,500);
6      dist=60;
7      pA= new PVector(0,dist);
8
9      while(pA.y <= height){
10         escalon();
11         punto();
12         repe();
13     }
14 }
15 void escalon(){
16     stroke(#D80910);
17     strokeWeight(5);
18     pB= new PVector (pA.x+dist, pA.y);
19     line(pA.x,pA.y,pB.x,pB.y);
20     pC= new PVector(pB.x,pB.y+dist);
21     line(pB.x,pB.y,pC.x,pC.y);
22 }
23 void punto(){
24     stroke(#11BBED);
25     strokeWeight(10);
26     pD= new PVector(pB.x,pB.y-15);
27     point(pD.x,pD.y);
28 }
29
30 void repe(){
31     pA.x = pC.x;
32     pA.y = pC.y;
33 }

```

Lienzo



Ejercicio 22: 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 franjas 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 sucesivamente. Las líneas tienen un color fijo, los círculos asumen colores aleatorios.

Análisis:

Datos de entrada: líneas, círculos, ancho y alto del lienzo

Datos de salida: bolitas de diferente color sobre líneas distanciadas

Proceso: ¿Quién realiza el proceso? Programa

¿Qué proceso realiza? Divide el lienzo en líneas alternadas, con círculos de diferentes colores esparcidos sobre ellas

Diseño:

Entidad que resuelve el problema: Programa
Variables: <code>distCirc= int</code> <code>rectaX, rectaY, circX, circY, distCirc= int</code> <code>anchoLienzo, altoLienzo: int</code>
Nombre del algoritmo: <code>circulo_repeticion</code>
<ul style="list-style-type: none"> • <i>Inicio</i> • <code>anchoLienzo ← 600</code> • <code>altoLienzo ← 600</code> • <code>rectaX ← 0</code> • <code>rectaY ← 100</code> • <code>distCirc ← 30;</code> • <code>circY ← 75</code> • hacer • <code>circX ← distCirc</code> • hacer • <i>dibujar linea en (rectaX, rectaY, anchoLienzo, rectaY)</i> • <i>dibujar circulo en (circX, circY, 50)</i>

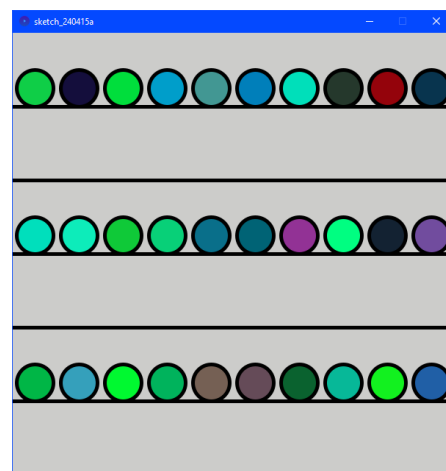
- $\text{circX} \leftarrow \text{circX} + \text{distCirc} * 2$
- **fin_hacer**
- **mientras** ($\text{circuloX} < \text{ancholienzo}$)
- $\text{rectaY} \leftarrow \text{rectaY} + 100;$
- $\text{circY} \leftarrow \text{circY} + 200;$
- **fin_hacer**
- **mientras** (lineaY sea menor que altoLienzo)
- *fin*



Captura de Processing / Lienzo

```

1 void setup(){
2   size(600,600);
3   int rectaX=0, rectaY=100, circY=75;
4   int distCirc=30;
5
6   do{
7     int circX= distCirc;
8     do{
9       stroke(10);
10      line(rectaX,rectaY,width,rectaY);
11      fill(random(120),random(255),random(180));
12      strokeWeight(5);
13      circle(circX,circY,50);
14      circX += distCirc*2;
15    }while(circX < width);
16    rectaY+= 100;
17    circY+=200;
18  }while(rectaY < height);
19 }

```



	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy PENSAMIENTO COMPUTACIONAL y PROGRAMACIÓN: Problema y Solución – PC y P – Algoritmos – Principio de la P</p>	
---	--	---

Fuentes bibliográficas

Canales de ayuda

<https://www.youtube.com/@noobscourse1591/videos>

<https://www.youtube.com/@DanielMarcial22>

<https://www.youtube.com/@Airroom/videos>

<https://www.youtube.com/@arielvega3350>

Fahrenheit a Celsius (Conversor de Google)

<https://www.google.com/search?client=opera-gx&q=convertir+de+fahrenheit+a+celsius&sourceid=opera&ie=UTF-8&oe=UTF-8>

Tutorial de Processing 3.0 – Buenos Aires Ciudad

Hola Mundo con Processing- Roció Abascal, Erick López, Sergio Zepeda- 2015