



**TECNICATURA UNIVERSITARIA EN
DISEÑO INTEGRAL DE
VIDEOJUEGOS**



**FACULTAD DE INGENIERÍA
Universidad Nacional de Jujuy**

**FUNDAMENTOS DE
PROGRAMACIÓN
ORIENTADA A OBJETOS**

TRABAJO PRÁCTICO/ACTIVIDAD

N° 1

APELLIDO Y NOMBRE – LU /

Aban Selene Marisol

TUV000557

Profesores:

Mg. Ing. Ariel Alejandro Vega

Año: 2024

REGLAMENTO

Crear una carpeta denominada TP01_XXXX donde XXXX es el apellido nombre del estudiante. Al producto final, subirlo en su repositorio y compartir el enlace en formulario.

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:

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

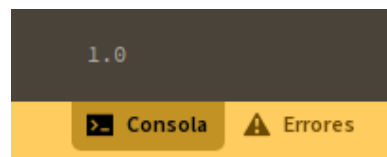
$$6-(4*B/4)$$

$$6-5$$

$$1$$

Captura de Processing

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



Ojo: Colocar la captura, no reemplaza que deban agregar a la carpeta el archivo .pde que contiene el código programado.

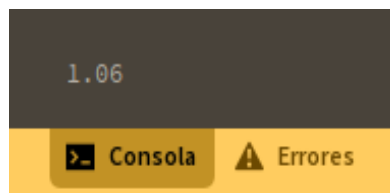
Ejercicio 2: Evaluar la siguiente expresión $4 / 2 * 3 / 6 + 6 / 2 / 1 / 5 ^ 2 / 4 * 2$

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

$$1.0 + 0.06$$

$$1.06$$

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



Ejercicio 3: Escribir las siguientes expresiones algebraicas como expresiones algorítmicas (en su forma aritmética dentro del algoritmo). En este caso no se pide evaluarlas ni programarlas.

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) $3 * X^4 - 5 * X^3 + X^{12} - 17$
- c) $(b + d) / (c + 4)$
- d) $(x^2 + y^2)^{(1/2)}$

Para aclarar que indicamos con "Luego escribirlas como expresiones algebraicas" lo aplicamos con el punto a)

$$b^2 - 4 * a * c$$

➤ Forma aritmética:

- a) $b^2 - 4 * a * c$ $a=4, b=5, c=1$
- $(5^2) - (4*4*1)$
- $25 - 16$
- 9

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

```
9.0
> Consola
! Errores
```

- b) $3*x^4-5*x^3+12-17$ $x=7$
- $(3*(7^4)) - (5*(7^3)) + (7*12) - 17$
- $7.2 - 1.7 + 84 - 17$
- 5.555



Trabajo Práctico N° 1: Operadores -
Metodología de Programación

ejercicio 4 b

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

5555.0

Consola Errores

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

$b=3, c=5, d=1$

$(3 + 1) / (5 + 4)$

$4 / 9$

0

ejercicio 4 c

```
1 int b=3, c=5, d=1;  
2  
3 float resultado= (b + d) / (c +4);  
4  
5 println(resultado);
```

0.0

Consola Errores

d) $(x^2 + y^2)^{(1/2)}$

$x=4 \ y=3$

$(4^2 + 3^2)^{(1/2)}$

5

sketch 240413f

```
1 int x=4, y=3;  
2  
3 float resultado= pow(pow(x,2)+ pow(y,3),0.5);  
4  
5 println(resultado);
```

6.5574384

Consola Errores

Resolví el ejercicio y el resultado seguía saliendo 5 aunque en processing me sale 6.55

➤ Forma algebraica:

a) $b^2 - 4.a.c$

$5^2 - 4.4.1$

$25 - 16$

9

b) $3.x^4 - 5.x^3 + x.12 - 17$

$3.7^4 - 5.7^3 + 7.12 - 17$

$$7203 - 1715 + 84 - 17$$

$$5.555$$

c) $\frac{(b+d)}{(c+4)}$

$$\frac{(3+1)}{(5+4)} = 0$$

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

$$(4^2 + 3^2)^{\left(\frac{1}{2}\right)}$$

$$(25)^{\left(\frac{1}{2}\right)} = 5$$

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

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

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

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

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

$$20 - 25 / 4$$

$$20 - 6.25$$

$$13.75$$

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

```
13.75
> Consola  Errores
```

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

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

$$20/9$$

$$2.22$$

```
sketch 240413h
1 int a=4, b=5, c=1;
2
3 float resultado= (a*b)/ pow(3,2);
4
5 println(resultado);
```

```
2.2222223
> Consola  Errores
```

c) $((B+C)/2*A+10)*3*B - 6$
 $((5+1)/2*4 + 10) * 3 * 5 - 6$
 $((6/2 * 4 + 10) * 3 * 5) - 6$
 $((3 * 4 + 10) * 3 * 5) - 6$
 $(22 * 3 * 5) - 6$
 $330 - 6$
 324

```

1 int a=4, b=5, c=1;
2
3 float resultado= ((b+c)/2*4+10)*3*b)-6;
4
5 println(resultado);
6

```

324.0

> Consola
⚠ Errores

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

$R1 = y+z$

$R2 = x \geq R1$

- $R1 = 4 + 1$
- $R2 = 4 \geq R1$

Falso

```

1 int x=3, y=4, z=1;
2
3 int R1= y+z;
4 boolean R2= 3 >= R1;
5 println(R2);
6

```

false

> Consola
⚠ Errores

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

$R1 = ++contador1$

$R2 = contador1 < contador2$

- $R2 = 4 < 4$
- $R2 = \text{Falso}$

```

ejercicio 7
1 int contador1=3, contador2=4;
2
3 int R1= ++contador1;
4
5 boolean R2= contador1 < contador2;
6
7 println(R2);

```

false

> Consola
⚠ Errores

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

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

➤ $A + B - 1 < X * Y$

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

$$31 - 2 < 6$$

$$29 < 6$$

Falso

```

sketch 240413l
1 int a=31, b=-1, x=3, y=2;
2
3 boolean desigualdad= a+b-1 < x*y;
4
5 println(desigualdad);
6

```

```

false
>_ Consola  ⚠ Errores

```

Ejercicio 9: Para $x=6$, $y=8$, evaluar el resultado de

$$!(x<5) \text{CC} !(y \geq 7)$$

➤ $!(X<5) \text{CC} !(y \geq 7)$

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

Falso && Falso

Falso

```

sketch 240413m
1 int x=6, y=8;
2
3 boolean resultado= !(x<5) && ! (y>=7);
4
5 println(resultado);
6

```

```

false
>_ Consola  ⚠ Errores

```

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

$$!(i>4) \parallel !(j \leq 6)$$

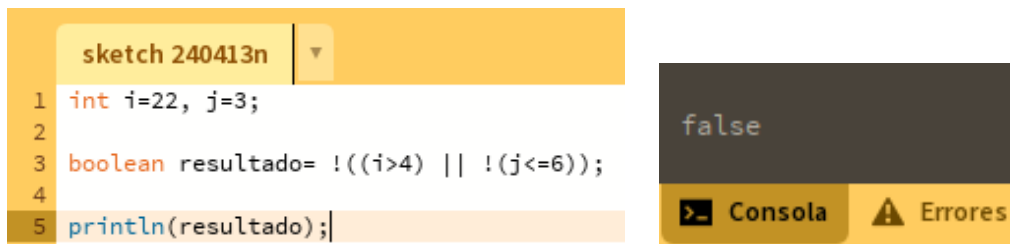
➤ $!(i>4) \parallel !(j \leq 6)$

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

$$!(\text{verdadero}) \parallel !(\text{falso})$$

!(verdadero)

Falso



```

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

```

Console: false

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

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

➤ !(a+b==c) || (c!=0)CC(b-c>=19)

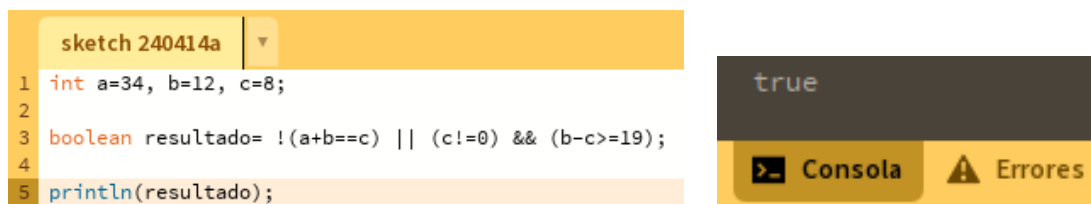
!(34 + 12==8) || (8!=0) && (12 - 8>=19)

!(46==8) || (8!=0) && (4>=19)

Verdadero || verdadero && falso

Verdadero || falso

Verdadero



```

1 int a=34, b=12, c=8;
2
3 boolean resultado= !(a+b==c) || (c!=0) && (b-c>=19);
4
5 println(resultado);

```

Console: 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.

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_ingresado // cadena

♥ Datos de salida: mensaje_saludo // cadena de texto

♥ **Proceso:**

- ◆ ¿Quién realiza el proceso? La computadora o el algoritmo
- ◆ ¿Cuál es el proceso que resuelve? Se ingresa un nombre que devolverá la creación de un saludo personalizado con el nombre proporcionado y su presentación en pantalla

Diseño:

+ Entidad que resuelve el problema: Algoritmo

+ Variables:

- ❖ nombre_ingresado: string // almacena el nombre
- ❖ mensaje_saludo: string // almacena una cadena de caracteres

+ Nombre del algoritmo: saludar_nombre

Proceso:

- Inicio
- Leer nombre_ingresado
- Mensaje_saludo ← "Hola," + nombre_ingresado + " ¡Bienvenido!"
- Mostrar saludo
- Fin

Intente plantearlo en processing de algunas variables distintas, pero el nombre es lo único que no me llega a reflejar el programa.

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 // decimales

♥ **Datos de salida:** perímetro, área// almacena valores decimales

♥ **Proceso:**

- ◆ ¿Quién realiza el proceso? Una calculadora o el usuario mismo
- ◆ ¿Cuál es el proceso que resuelve? Se calcula el área y el perímetro de un rectángulo usando las fórmulas correctas / adecuadas

$$P= 2(base + altura) \text{ y } A= base \cdot altura$$

Diseño:

+ Entidad que resuelve el problema: Persona/Usuario

+ Variables:

- ❖ base: int // almacena un valor decimal
- ❖ altura: int // almacena un valor decimal
- ❖ perimetro: float //
- ❖ area: float //
- ❖ perimetroArea: float // almacena un valor de calculos

 **Nombre del algoritmo:** `perímetro_area_rectangulo`

Proceso:

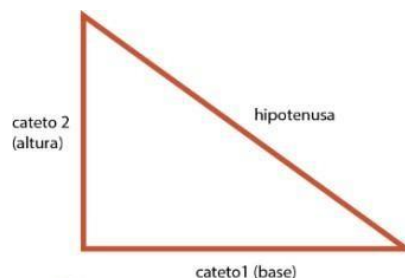
- Inicio
- Leer base
- Leer área
- $Perímetro \leftarrow 2 * (base * altura)$
- $Área \leftarrow base * altura$
- `perimetroArea` \leftarrow "el perímetro de un rectángulo es:" + `perímetro` + "y el área de un rectángulo es:" + `área`
- mostrar `perimetroArea`

```
ejercicio 13
1 int base=10, altura=7;
2
3 float perimetro= 2*( base * altura);
4 float area= base*altura;
5 String perimetroArea= "el perimetro de un rectangulo es:" + perimetro + "y el area de un rectangulo es:" + area;
6 println(perimetroArea);
7
```

el perimetro de un rectangulo es:140.0y el area de un rectangulo es:70.0

 Consola  Errores

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:** `catetoA`, `catetoB`

♥ **Datos de salida:** `hipotenusa`

♥ Proceso:

- ♦ ¿Quién realiza el proceso? Una calculadora o el usuario mismo
- ♦ ¿Cuál es el proceso que resuelve? Se calcular la longitud hipotenusa de un triángulo rectángulo las longitudes de los catetos como entrada, se aplica la fórmula: $h^2 = a^2 + b^2$

$$h = \sqrt{a^2 + b^2}$$

Diseño:

✚ Entidad que resuelve el problema: Persona/Usuario

✚ Variables:

- ❖ catetoA: int// almacena un valor decimal
- ❖ catetoB: int // almacena un valor decimal
- ❖ hipotenusa: int // almacena el valor de los cálculos

✚ Nombre del algoritmo: perímetro_area_rectangulo

Proceso:

- Inicio
- Leer catetoA
- Leer catetoB
- $Hipotenusa \leftarrow (a^2 + b^2)^{0.5}$
- Mostrar hipotenusa

```

1 int catetoA=10, catetoB=12;
2
3 float hipotenusa= (int) pow(pow(catetoA,2) + pow(catetoB,2),0.5);
4
5 println(hipotenusa);

```

15.0

➤ Consola ⚠ Errores

Ejercicio 15: Si viste algo de los apuntes y videos, 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: num1, num2

♥ Datos de salida: suma, resta, multiplicación, division

♥ Proceso:

- ♦ ¿Quién realiza el proceso? Una calculadora o el usuario mismo
- ♦ ¿Cuál es el proceso que resuelve?

Diseño:

✚ Entidad que resuelve el problema: Persona/Usuario

✚ Variables:

- ❖ Num1: int // almacena un valor entero
- ❖ Num2: int // almacena un valor entero
- ❖ Suma: int // almacena el valor de una suma
- ❖ Resta: int // almacena el valor de una resta
- ❖ Multiplicación: int // almacena el valor de una multiplicación
- ❖ División: int // almacena el valor de una división

✚ Nombre del algoritmo: calculadora básica

Proceso:

- Inicio
- Leer num1
- Leer num2
- $\text{suma} \leftarrow \text{num1} + \text{num2}$
- $\text{mostrar} \leftarrow \text{"el resultado de la suma es:"} + \text{suma}$
- $\text{resta} \leftarrow \text{num1} - \text{num2}$
- $\text{mostrar} \leftarrow \text{"el resultado de la res es:"} + \text{resta}$
- $\text{multiplicación} \leftarrow \text{num1} * \text{num2}$
- $\text{mostrar} \leftarrow \text{"el resultado de la multiplicación es:"} + \text{multiplicación}$
- $\text{división} \leftarrow \text{num1} / \text{num2}$
- $\text{mostrar} \leftarrow \text{"el resultado de la división es:"} + \text{división}$

```

1  int num1=14, num2=7;
2
3  int suma= num1 + num2;
4  println("el resultado de la suma es:" + suma);
5  int resta= num1 - num2;
6  println("el resultado de la resta es:" + resta);
7  int multiplicacion= num1 * num2;
8  println("el resultado de la multiplicacion es:" + multiplicacion);
9  int division= num1 / num2;
10 if (num2!=0){
11     println("el resultado de la division es:" + division);
12 } else{
13     println("la division por cero no esta definida.");
14 }
    
```

```

el resultado de la suma es:21
el resultado de la resta es:7
el resultado de la multiplicacion es:98
el resultado de la division es:2
    
```

➤ 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

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

Análisis:

- ♥ Datos de entrada: temperatura de grados Fahrenheit
- ♥ Datos de salida: temperatura de grados Celsius
- ♥ Proceso:

- ◆ ¿Quién realiza el proceso? Una calculadora o un programa informático
- ◆ ¿Cuál es el proceso que resuelve? Consiste convertir los grados otorgados en Fahrenheit a grados Celsius utilizando la formula correspondiente
$$c = \frac{9}{5} \cdot (F - 32)$$
- ◆ Esto seria restas los 32 grados por los grados Fahrenheit, donde multiplicamos ese resultado por $\frac{9}{5}$ y así obtenemos la temperatura de Celsius

Diseño:

- + Entidad que resuelve el problema: Persona/Usuario
- + Variables:
 - ❖ temperaturaFahrenheit: int // almacena un valor decimal
 - ❖ temperaturaCelsius: float // almacena un valor decimal
- + Nombre del algoritmo: conversor_de_temperatura_celsius

Proceso:

- Inicio
- Leer temperaturaFahrenheit
- temperaturaCelsius $\leftarrow (5.9 / 9.0) * (temperaturaFahrenheit - 32)$
- mostrar temperaturaCelsius

```

1 int temperaturaFahrenheit= 75;
2 float temperaturaCelsius= (5.0/9.0) * (temperaturaFahrenheit - 32);
3
4 println("temperatura en celsius:" + temperaturaCelsius);

```

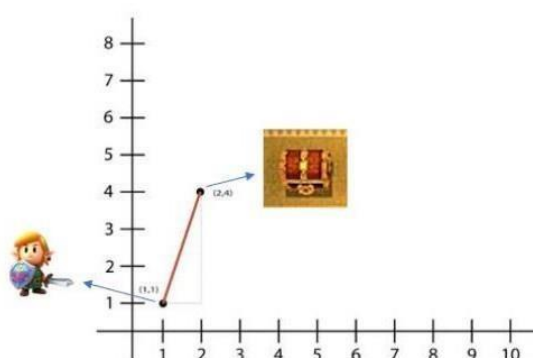
```

temperatura en celsius:23.88889

```

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



Punto 1

x1 = 1
y1 = 1

Punto 2

x2 = 2
y2 = 4

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 de Link, coordenadas del tesoro

♥ **Datos de salida:** distancia entre Link y el tesoro

♥ **Proceso:**

- ◆ ¿Quién realiza el proceso? Una calculadora que realice cálculos matemáticos o un programa informático
- ◆ ¿Cuál es el proceso que resuelve? Se calcula la distancia entre x;y entre los puntos que nos darán los catetos

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Diseño:

✚ **Entidad que resuelve el problema:** Persona/Usuario

✚ **Variables:**

- ❖ x1: int // almacena un valor decimal
- ❖ y1: int // almacena un valor decimal
- ❖ x2: int // almacena un valor decimal
- ❖ y2: int // almacena un valor decimal
- ❖ coordenadasX: float // almacena el resultado de un calculo
- ❖ coordenadasY: float // almacena el resultado de un calculo
- ❖ distancia: float // almacena el resultado de un calculo
- ❖ distanciaTesoro: float // almacena un valor

✚ **Nombre del algoritmo:** distancia_puntos

Proceso:

- Inicio
- Leer x1
- Leer y1
- Leer x2
- Leer y2
- distanciaTesoro ← 50
- coordenadasX ← x2 - x1
- coordenadasY ← y1 - y2
- distancia ← ((coordenadaX)^2 + (coordenadaY)^2)^2
- mostrar "la distancia es de:" + distancia
- mostrar mensaje "¡PowerUp activado!"



```

ejercicio 17
1 int x1=100, y1=100, x2=200, y2=400;
2 PImage linkImage; // Imagen de Link
3 PImage tesoroImage; // Imagen del tesoro
4 float distanciaTesoro=50;
5
6 void setup() {
7   size(800, 600);
8   linkImage = loadImage("link.gif");
9   tesoroImage = loadImage("c.gif");
10  linkImage.resize(80, 80);
11  tesoroImage.resize(80, 80);
12 }
13
14 void draw() {
15   background(135, 206, 250);
16   float coordenadaX = x2 - x1; // Calcular tamaño de los catetos de Link
17   float coordenadaY = y2 - y1; // Calcular tamaño de los catetos del tesoro
18
19   float distancia = sqrt(pow(coordenadaX, 2) + pow(coordenadaY, 2)); //calcular la distancia
20   String textoDistancia = "la distancia es de: " + distancia; // mostrará una cadena de texto
21   println(textoDistancia);
22
23   //mensaje ¡Power-Up activado! si Link está en la posición del cofre
24   if (distancia <= distanciaTesoro) {
25     println("¡Power-Up activado!");
26   }
27
28   //imagenes
29   image(tesoroImage, x2, y2);
30   image(linkImage, x1, y1);
31
32   //texto de coordenadas mostradas en pantalla
33   String coordenadas = "X1: " + mouseX + ", Y1: " + mouseY;
34   fill(0);
35   textSize(20);
36   textAlign(RIGHT, TOP);
37   text(coordenadas, width, 0);
38 }
39
40 void mouseMoved(){
41   x1=mouseX;
42   y1=mouseY;
43 }

```

```

la distancia es de: 318.55453
la distancia es de: 318.55453
la distancia es de: 318.55453
la distancia es de: 318.55453

```

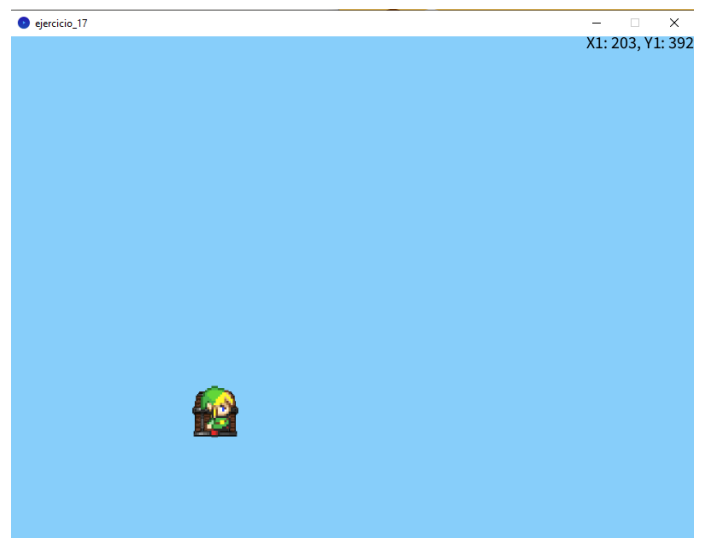
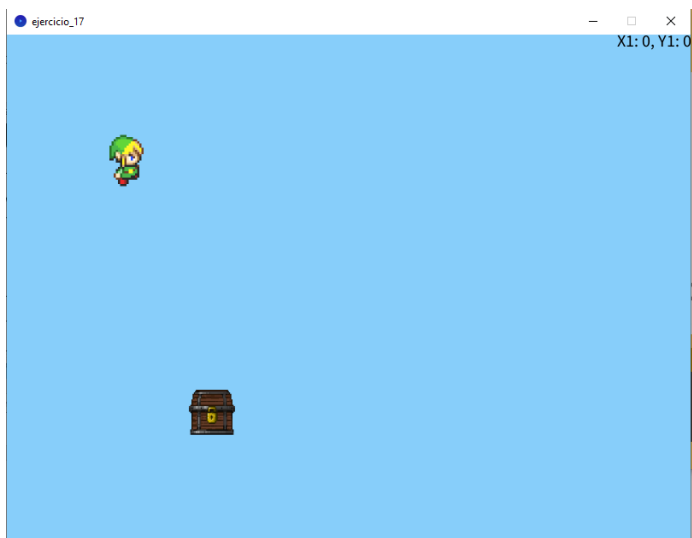
Consola Errores

```

la distancia es de: 5.0990195
¡Power-Up activado!
la distancia es de: 5.0990195
¡Power-Up activado!

```

Consola Errores



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:** coeficientes de la ecuación cuadrática

♥ **Datos de salida:** raíces de la ecuación cuadrática

♥ **Proceso:**

- ◆ ¿Quién realiza el proceso? Una calculadora que realice cálculos matemáticos o un programa informático
- ◆ ¿Cuál es el proceso que resuelve? Calcular el discriminante de la ecuación cuadrática usando la siguiente formula

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Diseño:

✚ **Entidad que resuelve el problema:** Persona/Usuario

✚ **Variables:**

- ❖ a: int // almacena un valor decimal
- ❖ b: int // almacena un valor decimal
- ❖ c: int // almacena un valor decimal
- ❖ discriminante: float // almacena el valor de los calculos

✚ **Nombre del algoritmo:** encontrar_raices

Proceso:

- Inicio
- Leer a
- Leer b
- Leer c
- Discriminante $\leftarrow b^2 - 4 * a * c$
- Si (discriminante > 0) entonces
- Raiz1 $\leftarrow (-b + (\text{discriminante})^{0.5}) / (2 * a)$
- Raiz2 $\leftarrow (-b - (\text{discriminante})^{0.5}) / (2 * a)$
- Mostrar “las raíces son:” + raiz1 “y” + raiz2
- Si_no si (discriminante == 0) entonces
- Raíz $\leftarrow -b / 2 * a$
- Mostrar “la raíz doble es:” + raíz
- Si_no
- Mostrar “no hay raíces reales”



A=1, b=0, c=1

```
1 float a, b, c;
2
3 void setup() {
4   size(400, 200);
5   background(255);
6   float a=1;
7   float b=0;
8   float c=-1;
9
10  // Calcular el discriminante
11  float discriminante = pow(b,2) - 4*a*c;
12
13  // Determinar el número y tipo de raíces
14  if (discriminante > 0) {
15    // Dos raíces reales distintas
16    float x1 = (-b + sqrt(discriminante)) / (2*a);
17    float x2 = (-b - sqrt(discriminante)) / (2*a);
18    println("Las raíces son: " + x1 + " y " + x2);
19  } else if (discriminante == 0) {
20    // Dos raíces reales iguales
21    float x = -b / (2*a);
22    println("La raíz doble es: " + x);
23  } else {
24    // No hay raíces reales (raíces complejas)
25    println("No hay raíces reales (raíces complejas).");
26  }
27 }
```

Las raíces son: 1.0 y -1.0

Consola Errores

A=2, b= -4, c=2

La raíz doble es: 1.0

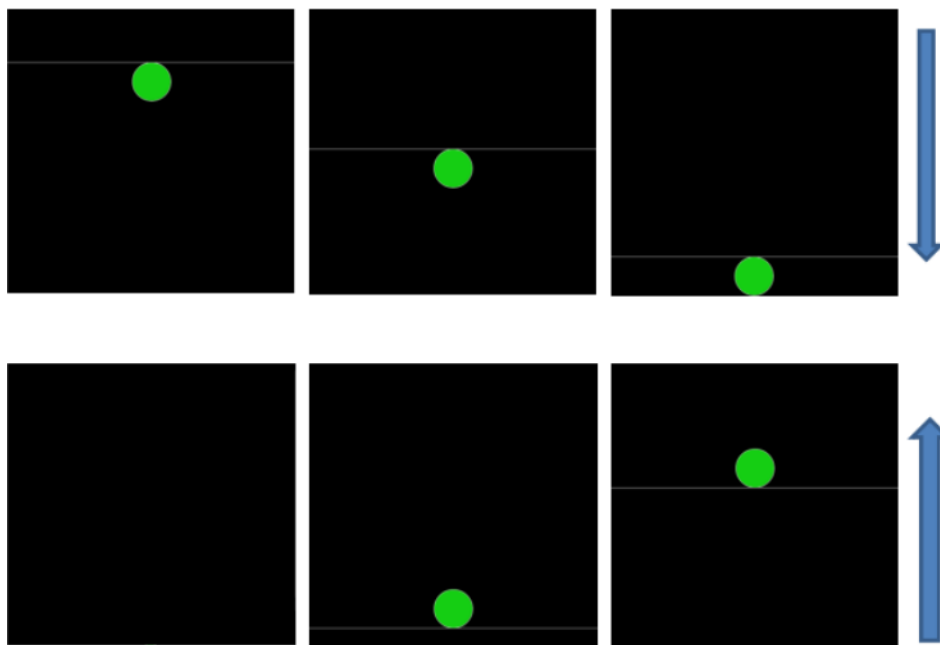
Consola Errores

A=2, b= 2, c=2

No hay raíces reales (raíces complejas).

Consola Errores

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



Análisis:

- ♥ **Datos de entrada:** línea, dir
- ♥ **Datos de salida:** bucle de la línea y círculo
- ♥ **Proceso:**
 - ◆ ¿Quién realiza el proceso? Una computadora
 - ◆ ¿Cuál es el proceso que resuelve?

Diseño:

- + **Entidad que resuelve el problema:** lienzo
- + **Variables:**
 - ❖ Línea: entero // almacena valor entero
 - ❖ Dir: entero // almacena valor entero
- + **Nombre del algoritmo:** línea_circulo_en_movimiento

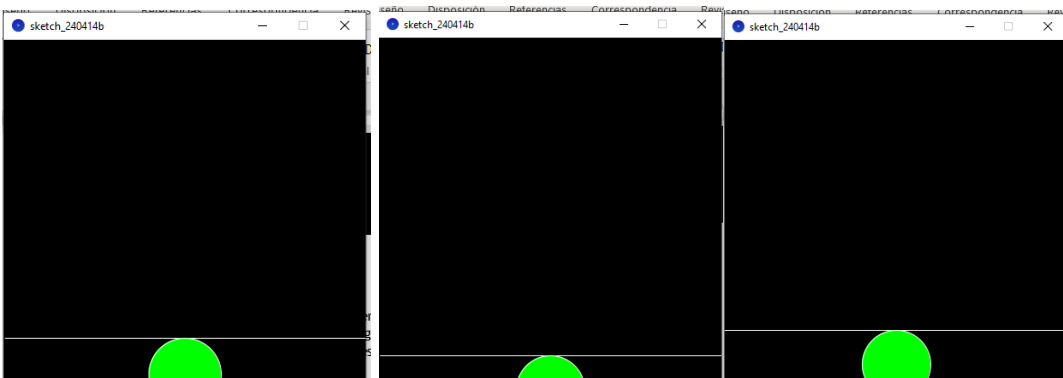
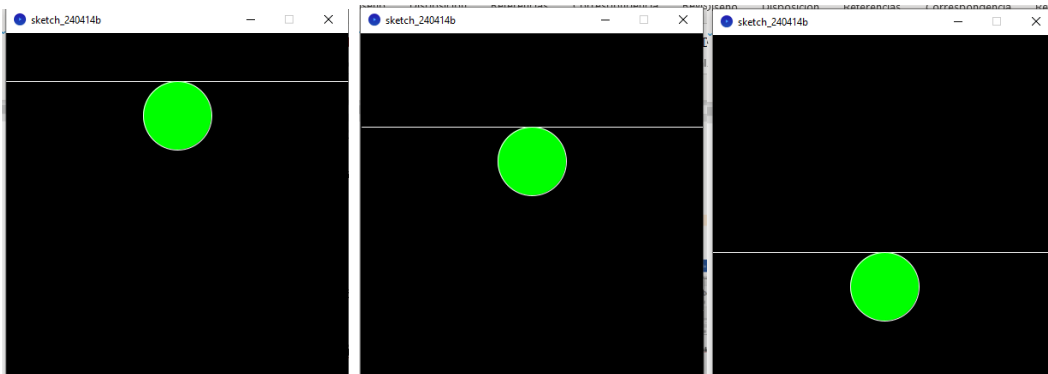
Proceso:

- Inicio
- Leer línea
- Leer dir
- anchoLienzo ← 400
- altoLienzo ← 400
- para $i \leftarrow 0$ hasta alto incremento 1 hacer
- línea ← línea + dir
- fin para
- si $((\text{línea} \geq \text{anchoLienzo}) \text{ O } (\text{línea} \leq 0))$ **entonces**
- dir ← dir * (− 1)
- fin si
- mostrar línea
- dibujar linea en (dir, linea, altoLienzo, linea)
- dibujar circulo en (altoLienzo/2, linea + 40, 80, 80)

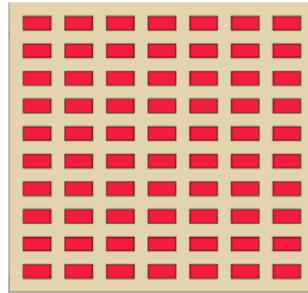


Trabajo Práctico N° 1: Operadores -
Metodología de Programación

```
sketch_240414b
1 int linea;
2 int dir = 1;
3
4 void setup() {
5   size(400, 400);
6   linea = 200;
7 }
8
9 void draw() {
10  background(0);
11
12  for (int i = 0; i < 1; i++) {
13    linea = linea + dir; // incrementa
14  }
15  if (linea >= height || linea <= 0) {
16    dir = dir * -1;
17  }
18  println(linea);
19
20  stroke(255);
21  fill(0,255,0);
22  line(dir, linea, width, linea);
23  ellipse(width/2, linea + 40, 80, 80);
24 }
25
```



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:

- ❖ anchoLienzo: Entero
- ❖ altoLienzo: Entero
- ❖ anchoRect: Entero
- ❖ altoRect: Entero
- ❖ distanciaEntreRect: Entero

♥ Datos de salida: Dibujar rectángulos en todo el lienzo

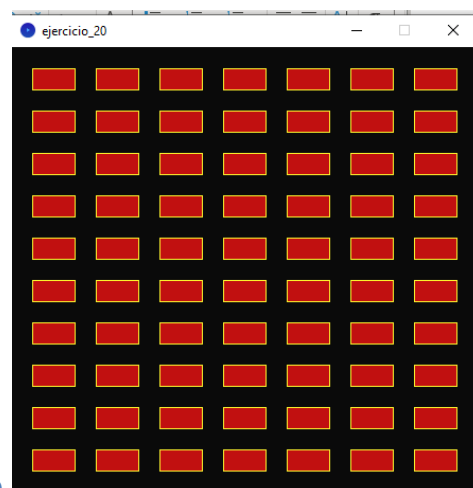
Diseño:

- ✚ Entidad que resuelve el problema: lienzo
- ✚ Nombre del algoritmo: rectángulos_repetidos

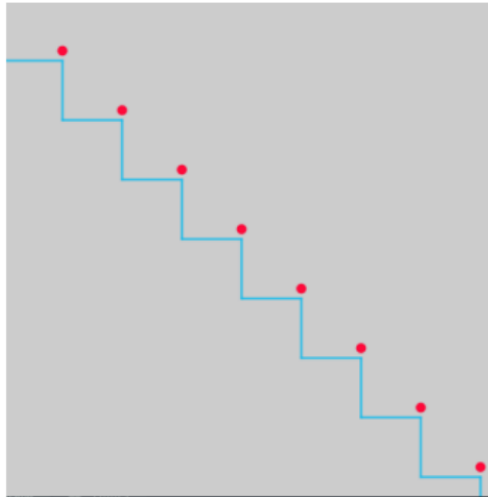
Proceso:

- Inicio
- Leer línea coordenadasRect: float //almacena un valor de coordenadas
- ancho, alto, distanciaEntreRect : int //almacena un valor entero
- anchoLienzo, altoLienzo: int //almacenan valores enteros

```
ejercicio 20
1 PVector coordenadasRect;
2 int alto, ancho, distRect;
3
4 void setup(){
5     size(440,420);
6     distRect = 20;
7     ancho= 40;
8     alto= 20;
9     coordenadasRect= new PVector(distRect,distRect);
10 }
11
12 void draw(){
13     background(10);
14     fill(#C11010);
15     stroke(#FCF32E);
16     dibujarRec();
17 }
18
19 void dibujarRec(){
20     for(float x=coordenadasRect.x;x<width;x+=(ancho+distRect)){
21         for(float y=coordenadasRect.y;y<height;y+=(alto+distRect)){
22             rect(x,y,ancho,alto);
23         }
24     }
25 }
```



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, distancia
- ♥ **Datos de salida:** una imagen que consiste en escalones con puntos rojos en los bordes
- ♥ **Proceso:**
 - ◆ ¿Quién realiza el proceso? El programa mediante el código en processing
 - ◆ ¿Cuál es el proceso que resuelve? Consiste en iterar mediante while() para dibujar los puntos rojos y los escalones donde estarían ubicados en los bordes

Diseño:

- ✚ **Entidad que resuelve el problema:** programa
- ✚ **Variables:**
 - ❖ puntoA, puntoB, puntoC, distancia: int // almacena un vector
 - ❖ distancia: int // almacena un valor entero
- ✚ **Nombre del algoritmo:** escalones_puntos

Proceso:

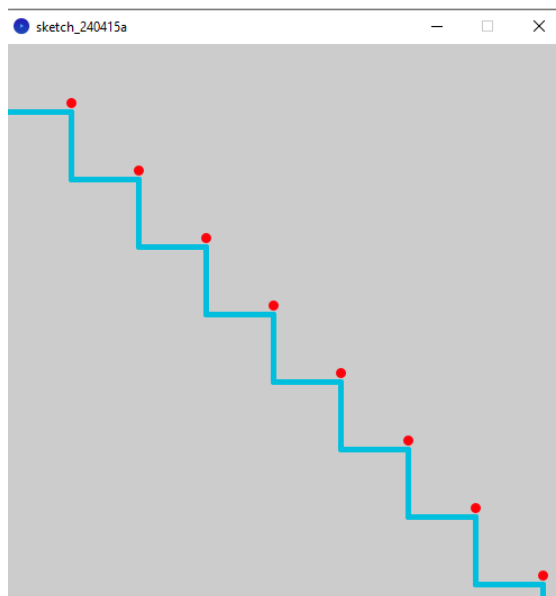
- Inicio
- anchoLienzo ← 500
- altoLienzo ← 500
- distancia ← 60

- mientras (puntoA.y sea menor o igual que anchoLienzo) Hacer
- dibujar línea horizontal en (puntoA.x, puntoA.y, puntoB.x, puntoB.y)
- dibujar línea vertical en (puntoB.x, puntoB.y, puntoC.x, puntoC.y)
- dibujar círculo en (puntoD.x, puntoD.y)
- puntoA.x ← puntoC.x
- puntoA.y ← puntoC.y
- fin_mientras

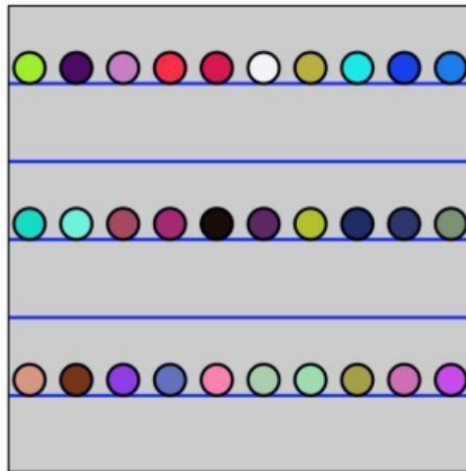
```

1  int distancia;
2  PVector puntoA, puntoB, puntoC, puntoD;
3
4
5  public void setup () {
6    size(500,500);
7    distancia=60;
8    puntoA = new PVector(0,distancia);
9
10   while(puntoA.y <= height){
11     escalon();
12     circulo();
13     repeticion();
14   }
15 }
16
17 public void escalon(){
18   stroke(#00BEDE);
19   strokeWeight(5);
20   puntoB = new PVector(puntoA.x+distancia, puntoA.y);
21   line(puntoA.x, puntoA.y,puntoB.x,puntoB.y);
22   puntoC = new PVector(puntoB.x,puntoB.y+60);
23   line(puntoB.x,puntoB.y,puntoC.x,puntoC.y);
24 }
25
26 public void circulo(){
27   stroke(#FC030B);
28   strokeWeight(9);
29   puntoD = new PVector(puntoB.x, puntoB.y-8);
30   point(puntoD.x,puntoD.y);
31 }
32
33 public void repeticion(){
34   puntoA.x = puntoC.x;
35   puntoA.y = puntoC.y;
36 }

```



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:** número de líneas y círculos

♥ **Datos de salida:** círculo de colores aleatorios sobre línea de un color con distancia de por medio

♥ **Proceso:**

- ♦ ¿Quién realiza el proceso? El lienzo se divide verticalmente en franjas de igual medida, donde se dibujan líneas en todas ellas. En cada línea de forma alternada, se dibujan círculos con colores aleatorios, los cuales están espaciados uniformemente a lo largo de la línea.
- ♦ ¿Cuál es el proceso que resuelve? El programa utilizado, actualmente utilizado processing

Diseño:

✚ **Entidad que resuelve el problema:** Processing

✚ **Variables:**

- ❖ `distanciaCirculo: int` // almacena un valor entero
- ❖ `lineaX, lineaY, circuloX, circuloY, distanciaCirculo : int` //almacena un valor entero
- ❖ `anchoLienzo,altoLienzo: int` // almacenan valores enteros

✚ **Nombre del algoritmo:** círculos_repetidos

Proceso:

- Inicio
- `anchoLienzo ← 600`
- `altoLienzo ← 600`



- lineaX ← 0
- lineaY ← 100
- distanciaCirculo ← 30;
- circuloY ← 75
- circuloX ← distanciaCirculo
- dibujar linea en (lineaX, lineaY, anchoLienzo, lineaY)
- dibujar circulo en circuloX, circuloY, 50, 50)
- circuloX ← circuloX + distanciaCirculo*2
- mientras (circuloX sea menor que anchoLienzo)
- LineaY ← lineaY + 100;
- circuloY ← circuloY + 200;
- mientras (lineaY sea menor que altoLienzo)

```

1 void setup() {
2   size(600,600);
3   int lineaX = 0;
4   int lineaY = 100;
5   int circuloY = 75;
6   int distanciaCirculo = 30;
7
8   do{
9     int circuloX = distanciaCirculo;
10
11   do{
12     stroke(#008DFC);
13     line(lineaX,lineaY,width,lineaY);
14     fill(random(255), random(255), random(255));
15     stroke(0);
16     strokeWeight(2);
17     ellipse(circuloX,circuloY,50,50);
18     circuloX += distanciaCirculo*2;
19
20   }while(circuloX < width);
21   lineaY += 100;
22   circuloY += 200;
23
24 }while(lineaY < height);
25 }

```

