



TECNICATURA UNIVERSITARIA EN DISEÑO  
INTEGRAL DE VIDEOJUEGOS

FACULTAD DE INGENIERÍA  
Universidad Nacional de Jujuy



# FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS

Trabajo Practico/Actividad



N°1

Alancay Ramon Jorge-LU000666

Profesor

Mg.Ing.Ariel Alejandro Vega

Año:2024

	<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

### Hoja

3.....	Ejercicio 1 2 4
4.....	Ejercicio Ejercicio4
5.....	Ejercicio 5
6.....	Ejercicio 6 7 8
7.....	Ejercicio 9 10 11
8.....	Ejercicio 12
9.....	Ejercicio 13
10.....	Ejercicio 14
11.....	Ejercicio 15
12.....	Ejercicio 16
13.....	Ejercicio 17
14.....	Ejercicio 18
15.....	Ejercicio 18 19
16.....	Ejercicio 19
17.....	Ejercicio 20
18.....	Ejercicio 20
19.....	Ejercicio 21
20.....	Ejercicio 21
21.....	Ejercicio 22
22.....	Ejercicio 22
23.....	Conclusion

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

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

Resolución necesaria en Word:

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

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

$$6 - 5$$

$$1$$

```

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

```

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

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

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

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

$$4/2 * 3/6 + 0,03 * 2$$

$$2 * 0,5 + 0,03 * 2$$

$$1 + 0,06$$

$$1,06$$

```

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

$$b=2 \quad a=3 \quad c=4$$

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

$$(2^2) - (4 * 3 * 4)$$

$$4 - (4 * 3 * 4)$$



$$4 - 48$$

$$-44$$

Captura

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

a)  $b^2 - 4ac$

$$4 - 48$$

$$-44$$

b)  $3 * X^4 - 5 * X^3 + X^{12} - 17$

$$X=2$$

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

$$(3 * 16) - (5 * 8) + (2 * 12) - 17$$

$$48 - 40 + 24 - 17$$

$$15$$

$$3x^4 - 5x^3 + 2x^{12} - 17$$

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

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

$$b=2 \quad d=6 \quad c=4$$

$$(2 + 6) / (4 + 4)$$

$$6 / 8$$

$$1$$

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

```
int b=2,d=6,c=4;  
float resultado= (b + d)/(c + 4);  
println(resultado);
```

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

$$X=2 \quad y=3$$

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



$$(4 + 9)^{(1/2)}$$

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

$$3,60$$

$$\sqrt{x^2 + y^2}$$

```
int x=2,y=3;  
float resultado=sqrt((pow(x,2) + pow(y,2)));  
println(resultado);
```

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

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

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

$$5 * 4 - 25 / 4 * 1$$

$$20 - 25 / 4$$

$$20 - 6,25$$

$$13,75$$

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

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

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

$$20 / 9$$

$$2,222$$

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

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

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

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

$$((12 + 10) * 3 * 5) - 6$$

$$(22 * 3 * 5) - 6$$

330 – 6

324

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

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

R1 = y+z

R1= 4 + 1

R1= 5

R2 = x >= R1

R2 = 3 >= R1

R2= falso

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

Ejercicio 7: Para contador1=3, contador2=4, evaluar el resultado de

A) R1 = ++contador1

R1 = 4

B) R2 = contador1 < contador2

R2 = 3 < 4

R2= true

```
int contador1=3;  
float R1= ++contador1;  
println(R1);
```

```
int contador1=3,contador2=4;  
boolean R2 = contador1 < contador2;  
println(R2);
```

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

a+b-1 < x\*y

---



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

$31-1-1<3*2$

$31-1-1<6$

$29<6$

False

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

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

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

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

true por el ! && False por el !

Y por la conjunción && esto es false

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

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

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

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

$(22>4)$  true

$!(3<=6)$  false

Por el  $||$  esto es true pero al estar encerrado por el ! se vuelve falso

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

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

$!(a+b==c) \ || \ (c!=0) \text{CC} (b-c>=19)$

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

$!(34+12==8)$  = false pero por el ! Se vuelve true

$(8!=0)$  = esto es true por el !

$(12-8>=19)$  = esto es false

---



$(8 \neq 0) \ \&\& \ (12 - 8 \geq 19) =$  esto sabiendo lo anterior y al tener el  $\&\&$  es false

Al final hacemos el  $||$  por lo cual da true como resultado final

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

### 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.

Descripción del problema: Deberá pedir por teclado al usuario un nombre y posteriormente realizará la presentación en pantalla de un saludo con el nombre indicado.

Datos de entrada:

- nombreUsuario

Datos de salida:

- saluO ← Hola + nombreUsuario

Proceso:

Escribir nombre para mostrar un saludo(hola)

Diseño:

Entidad:Programa
Variables nombrUsuario,saludO
Nombre Algoritmo: saludar Algoritmo: 1.Escribir nombreUsuario 2.Mostrar saludO 3. saluO ← Hola + nombreUsuario 4.Mostar en pantalla hola + nombreUsuario 5.Fin





```
String nombreUsuario;  
public void setup(){  
    nombreUsuario="Jorge";  
    salud0();  
}  
public void salud0(){  
    println("Hola" + nombreUsuario);  
}
```

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

Datos de Entrada

base : Real

altura : Real

Datos de salida

Perimetro : Real

Area : Real

Proceso:

Se indica los valores de la basE y alturA

El programa calcula usando los valores de basE y alturA para determinar el

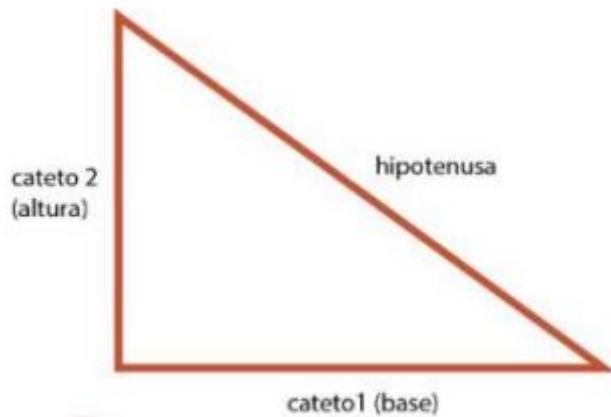
Perimetro  $\leftarrow$  basE + basE + alturA + alturA

Area  $\leftarrow$  basE \* alturA

ENTIDAD QUE RESUELVE EL PROBLEMA: Programa
<b>VARIABLES</b> basE: Real alturA: Real
<b>NOMBRE ALGORITMO:</b> calcular_perimetro 1.basE $\leftarrow$ 20 2.alrurA $\leftarrow$ 10 3.Perimetro $\leftarrow$ basE + basE + alturA + alturA 4.Area $\leftarrow$ basE * alturA 5.Mostrar Perimetro,Area 6.Fin

```
int basE=20,alturA=10;
String Perimetro,Area;
void setup(){
    float calculo= basE + basE+ alturA + alturA;
    //println(calculo);
    println("Perimetro:"+calculo);
    float resultado=basE * alturA;
    //println(resultado);
    println("Area:" + resultado);
}
```

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



Datos de entrada

cateToa, cateTob

Datos de salida

Hipotenusa

Proceso:

Lo resuelve la persona

Se debe calcular la hipotenusa sabiendo el valor de los catetos

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

ENTIDAD QUE RESUELVE EL PROBLEMA: persona

VARIABLES

cateToa: Real

cateTob: Real

Hipotenusa: Muestra Resultado

NOMBRE DEL ALGORITMO: hipotenusa

1. Leer cateToa

2. Leer cateTob

3. Calculo Hipotenusa  $\leftarrow (\text{cateToa} \wedge 2 + \text{cateTob} \wedge 2) \wedge (0,5)$

4. Mostrar hipotenusa

5. Fin

```
1 int cateToa=8,cateTob=9;
2 void setup(){
3     float calculo = pow(pow(cateToa,2) + pow(cateTob,2),0.5);
4     //println(calculo);
5     println("Hipotenusa=" + calculo);
6 }
```

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.

Datos de Entrada

numA,numB

Datos de Salida

SUMA,RESTA,MULTIPLICACION,DIVISION

Proceso:

Lo resuelve la calculadora

Se calcula la suma,resta,multiplicacion,division

ENTIDAD QUE RESUELVE EL PROBLEMA: calculadora

VARIABLES:

numA:Entero

numB: Entero

Suma: almacena resultado

Resta: almacena resultado

Multiplicacion: almacena resultado

Division: almacena resultado

NOMBRE DEL ALGORITMO: calculos\_basicos

1.Leer numA

2.Leer numB

3.Suma  $\leftarrow$  numA + numB

4.Resta  $\leftarrow$  numA - numB

5.Multiplicacion  $\leftarrow$  numA \* numB

6.Division  $\leftarrow$  numA / numB

7.Mostrar

8.Resultado de la suma

9.Resultado de la resta

10.Resultado de la multiplicacion

11.Resultado de la division

12.fin

```
int numA=10,numB=5;
float suma = numA + numB;
println(" Suma = " + suma);
float resta = numA - numB;
println(" Resta = " + resta);
float multiplicacion = numA * numB;
println(" Multiplicacion = " + multiplicacion);
float division = numA / numB;
println(" Division = " + division);
```

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

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

Datos de Entrada:

temperaturaFahrenheit

Datos de Salida:

Grados Celsius

Proceso:

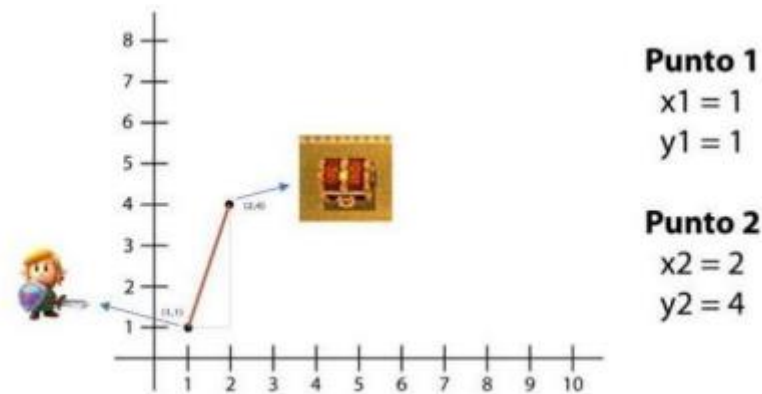
Lo resuelve el programa

Se hace una conversión de temperaturaFahrenheit a Grados Celsius usando la formula dada

ENTIDAD QUE RESUELVE EL PROBLEMA: programa
VARIABLES: temperaturaFahrenheit Grados Celsius
NOMBRE DEL ALGORITMO: conversión 1. Leer temperaturaFahrenheit 2. $\text{Conversion} \leftarrow (\text{temperaturaFahrenheit} - 32) / 1.8$ 3. Mostrar Grados Celsius 4. Fin

```
int temperaturaFahrenheit= 60;
float conversion = (temperaturaFahrenheit-32)/1.8;
println("Grados Celsius = " + conversion);
```

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  $(x_1, y_1)$ , mientras que las de la caja de tesoro se halla en la posición  $(x_2, y_2)$ . 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

Datos de Entrada

Coordenadas de link, tesoro

Datos de salida

Distancia entre Link y el tesoro

Proceso

Lo resuelve el programador

Hallamos la distancia entre link y el tesoro

ENTIDAD QUE RESUELVE EL PROBLEMA: programador

VARIABLES:

X1 cordenada

X2 cordenada

Y1 cordenada

Y2 cordenada

Cateto1, cateto2

Distancia entre Link y el Tesoro almacena calculo

NOMBRE DEL ALGORITMO: calcular\_distancia

1. Leer x1 , x2 , y1 , y2

2. Distancia entre link y el tesoro  $\leftarrow ((coordenadaX)^2 + (coordenadaY)^2)^2$

3. Mostrar distancia

4. Fin

```
int x1=1,y1=1,x2=2,y2=4;
int cateto1, cateto2;
int x , y;
public void setup(){
  size(700,600);
  distanciaLinkyTesoro();
}
public void distanciaLinkyTesoro(){
  cateto1=x2 - x1;
  cateto2= y2 - y1;
  println("Distancia entre Link y el Tesoro es" , pow((pow(cateto1,2) + pow(cateto2,2)
}
public void draw(){
  background(220);
  x = mouseX - 15;
  y = mouseY - 15;
  Link();
  Tesoro();
}
```

```
}
}
public void Link(){
  fill(#151CCB);
  ellipse(x,y,30,30);
}
public void Tesoro(){
  fill(#76531E);
  rect(200,200,40,40);
}
```

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.

Datos de entrada

A ← 2

B ← 2

C ← -4

Datos de salida

Raíces de la ecuación dependiendo de si la discriminante es > 0, ==0, imaginario

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

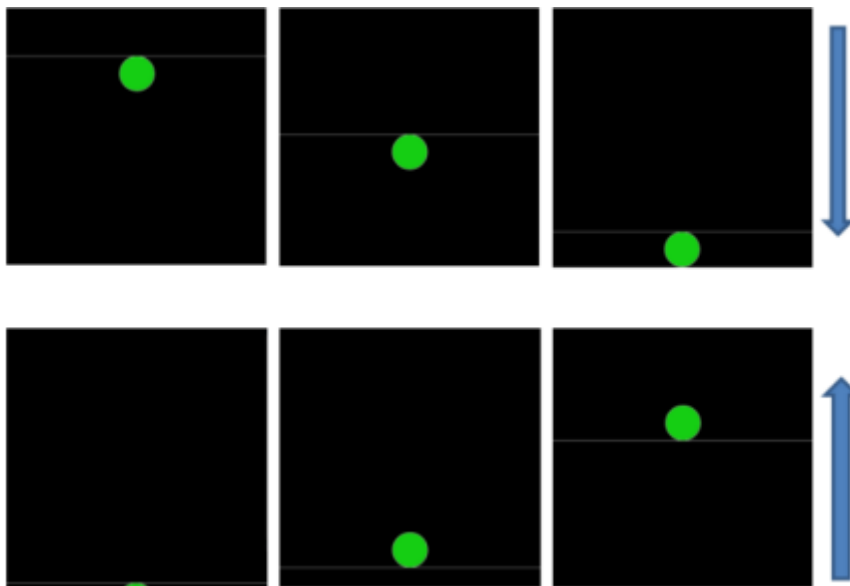
ENTIDAD QUE RESUELVE EL PROBLEMA: persona
VARIABLES: a: float // valor real b: float // valor real c: float // valor real Discriminante: float// valor de calculo
NOMBRE DEL ALGORITMO: calcular_raices 1. Leer a 2. Leer b 3. Leer c

4. discriminante  $\leftarrow b^2 - 4 * a * c$
5. Si ( discriminante > 0) entonces
6. Raiz1  $\leftarrow (-b + (\text{discriminante})^{0,5} / (2*a)$
7. Raiz2  $\leftarrow (-b - (\text{discriminante})^{0,5} / (2*a)$
8. Mostrar "Raíces = " + x1 + "y" + x2
9. Si\_no ( discriminante == 0 ) entonces
10. Raiz  $\leftarrow -b / (2*a)$
11. Mostrar "Raices iguales" + x
12. Si\_no
13. Mostrar "Raices imaginarias"
14. fin

```
float a=2,b=2,c=-4;
void setup(){
    size(400,200);
    background(255);
    float discriminante = pow(b,2) - 4 * a * c;
    println("Discriminante=" + discriminante);
    if (discriminante > 0){
        float x1 = (-b + sqrt(discriminante)) / (2*a);
        float x2 = (-b - sqrt(discriminante)) / (2*a);
        println("Raices =" + x1 + " y " + x2);
    }else if(discriminante == 0){
        float x = -b / (2 * a);
        println("Raices iguales" + x);
    }else{
        println("Raices imaginarias");
    }
}
```

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





Datos de entrada

direcCion

Linea

Datos de salida

Bucle y circulo

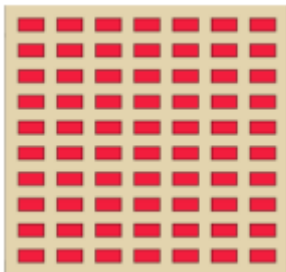
Lo realiza el programa

Un circulo encima de una linea que baja y sube

ENTIDAD QUE RESUELVE EL PROBLEMA:programa
VARIABLES Linea:entero DirecCion:entero
NOMBRE DEL ALGORITMO:linea_y_circulo <ol style="list-style-type: none"> <li>1. Leer direcCion</li> <li>2. Leer linea</li> <li>3. Ancho lienzo <math>\leftarrow</math> 600</li> <li>4. Alto lienzo <math>\leftarrow</math> 600</li> <li>5. Para <math>i \leftarrow 0</math> hasta alto incremento 1 hacer</li> <li>6. Linea <math>\leftarrow</math> linea + direcCion</li> <li>7. Si ((linea<math>\geq</math>ancho lienzo) o (linea<math>\leq</math>0)) entonces</li> <li>8. DirecCion <math>\leftarrow</math> direcCion * (-1)</li> <li>9. Mostrar linea</li> <li>10. Dibujar linea ( direcCion,linea,alto lienzo,linea)</li> <li>11. Dibujar circulo en (alto lienzo/2, linea + 40,80,80)</li> <li>12. Fin</li> </ol>

```
int direcCion=1;
int linea;
void setup(){
    size(600,600);
    linea = 300;
}
void draw(){
    background(0);
    for(int i = 0; i < 1; i++){
        linea = linea + direcCion;
    }
    if (linea >= height || linea <=0) {
        direcCion = direcCion * -1;
    }
    println(linea);
    stroke(255);
    fill(0,255,0);
    line(direcCion,linea,width,linea);
    ellipse(width/2, linea + 40 , 80 , 80);
}
```

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:



Datos de entrada

Los rectangulos en el lienzo segun los valores dados

Datos de salida

Los rectangulos en el lienzo segun los valores dados

Proceso

Lo realiza el progrma

Dibuja una serie de rectangulos con tamaños específicos

ENTIDAD QUE RESUELVE EL PROBLEMA: programador

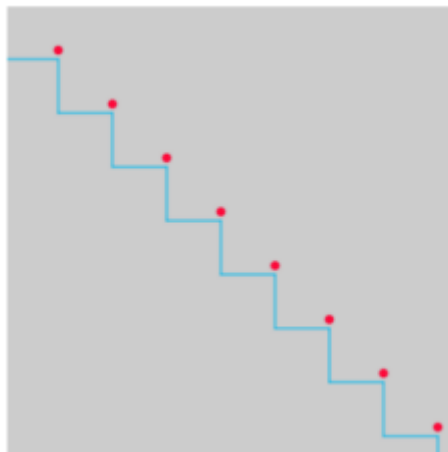
VARIABLES: CordenadasRect cordenadas Ancho, alto, distanciaRect  
AnchoLienzo,altoLienzo

NOMBRE DEL ALGORITMO: rectangulos

1. AnchoLienzo  $\leftarrow$  440
2. AltoLienzo  $\leftarrow$  420
3. AnchO  $\leftarrow$  40
4. AltO  $\leftarrow$  20
5. DisTancia  $\leftarrow$  20
6. Para x  $\leftarrow$  coordenadas.x hasta AnchoLienzo con paso ( anchO + disTancia)
7. Hacer
8. Para y  $\leftarrow$  cordenadas.y hasta AltoLienzo con paso ( altO + disTancia)
9. Hacer
10. Dibujar rectangulo en (x,y,anchO,altO)
11. Fin

```
PVector coordenadas;  
int alt0,anch0,disTancia;  
  
void setup(){  
    size(440,420);  
    disTancia = 20;  
    anch0= 40;  
    alt0= 20;  
    coordenadas= new PVector(disTancia,disTancia);  
}  
  
void draw(){  
    background(#BCAE74);  
    fill(#C11010);  
    stroke(#EA0707);  
    dibujarRec();  
}  
  
void dibujarRec(){  
    for(float x=coordenadas.x;x<width;x+=(anch0+disTancia)){  
        for(float y=coordenadas.y;y<height;y+=(alt0+disTancia)){  
            rect(x,y,anch0,alt0);  
        }  
    }  
}
```

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.

Datos de entrada

PuntoA,puntoB,puntoC,puntoD,distancia

Datos de salida

Imagen del resultado

ENTIDAD QUE RESUELVE EL PROBLEMA:programa
<b>VARIABLES</b> PuntoA,puntoB,puntoC,puntoD distancia
<b>NOMBRE DEL ALGORITMO:puntos_escalones</b> <ol style="list-style-type: none"> <li>1. AnchoLienzo <math>\leftarrow</math> 500</li> <li>2. AltoLienzo <math>\leftarrow</math> 500</li> <li>3. Distancia <math>\leftarrow</math> 60</li> <li>4. Mientras (puntoA.y sea menor o igual que anchoLienzo) hacer</li> <li>5. Dibujar linea horizontal en (puntoA.x,puntoA.y,puntoBx,puntoB.y)</li> <li>6. Dibujar linea vertical en (puntoB.x,puntoB.y,puntoC.x,puntoC.y)</li> <li>7. Dibujar circulo en (puntoD.x,puntoD.y)</li> <li>8. PuntoA.x <math>\leftarrow</math> puntoC.x</li> <li>9. PuntoA.y <math>\leftarrow</math> puntoC.y</li> <li>10. Fin</li> </ol>

```
int distancia;
PVector puntoA, puntoB, puntoC, puntoD;

public void setup (){
    size(500,500);
    distancia=60;
    puntoA = new PVector(0,distancia);

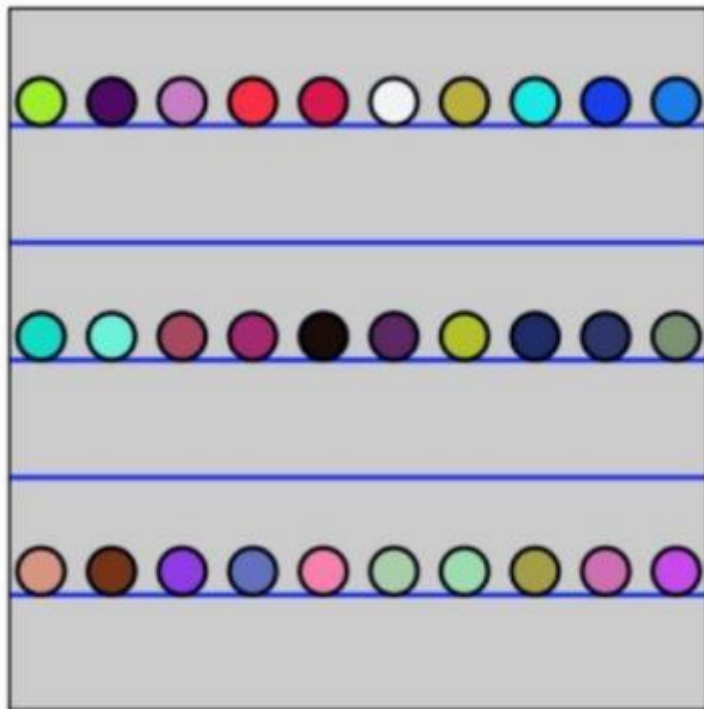
    while(puntoA.y <= height){
        escalon();
        circulo();
        repeticion();
    }
}

public void escalon(){
    stroke(#00BEDE);
    strokeWeight(5);
    puntoB = new PVector(puntoA.x+distancia, puntoA.y);
    line(puntoA.x, puntoA.y,puntoB.x,puntoB.y);
    puntoC = new PVector(puntoB.x,puntoB.y+60);
    line(puntoB.x,puntoB.y,puntoC.x,puntoC.y);
}

public void circulo(){
    stroke(#FC030B);
    strokeWeight(9);
    puntoD = new PVector(puntoB.x, puntoB.y-8);
    point(puntoD.x,puntoD.y);
}

public void repeticion(){
    puntoA.x = puntoC.x;
    puntoA.y = puntoC.y;
}
```

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.

Datos de entrada

Numero de lineas y círculos

Datos de salida

Círculos con colores aleatorios sobre líneas

Se debe programar una secuencia de círculos con diferentes colores encima de líneas

Lo realiza el programa

ENTIDAD QUE RESUELVE EL PROBLEMA: programa
<b>VARIABLES:</b> DistanciaCirculo LineaX, lineaY, circuloX, circuloY AnchoLienzo, altoLienzo
<b>NOMBRE DEL ALGORITMO:</b> circulos_y_lineas 1. AnchoLienzo ← 600

2. AltoLienzo  $\leftarrow$  600
3. LineaX  $\leftarrow$  0
4. LineaY  $\leftarrow$  100
5. DistanciaCirculo  $\leftarrow$  30
6. CirculoY  $\leftarrow$  75
7. Hacer
8. CirculoX  $\leftarrow$  distanciaCirculo
9. Hacer
10. Dibujar Linea en (lineaX,lineaY,anchoLienzo,lineaY)
11. Dibujar circuloX,circuloY,50,50
12. CirculoX  $\leftarrow$  circuloX + distanciaCirculo\*2
13. Fin\_hacer
14. Mientras(circuloX sea menor que anchoLienzo)
15. LineaY  $\leftarrow$  lineaY+100
16. CirculoY  $\leftarrow$  circuloY+200
17. Fin\_hacer
18. Mientras (lineaY sea menor que alto lienzo)
19. Fin

```
void setup(){
    size(600,600);
    int lineaX = 0;
    int lineaY = 100;
    int circuloY = 75;
    int distanciaCirculo = 30;
    do{
        int circuloX = distanciaCirculo;
    do{
        stroke(#008DFC);
        line(lineaX,lineaY,width,lineaY);
        fill(random(255), random(255), random(255));
        stroke(0);
        strokeWeight(2);
        ellipse(circuloX,circuloY,50,50);
        circuloX += distanciaCirculo*2;

    }while(circuloX < width);
        lineaY += 100;
        circuloY += 200;

    }while(lineaY < height);
}
```

## Conclusión

Se me complico bastante en los ultimos 4 ejercicios y use los materiales dados en clase.

---

Busque en google palabras como while,height,whdth porque no entendia bien su funcionamiento.

También formulas matemáticas

<https://www.youtube.com/watch?v=60FGh4NQoRs>

