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

Trabajo Práctico N° 1

Valdez, Martin Leonel

LU: TUV000043

*Profesores:*

*Mg. Ing. Ariel Alejandro Vega*

*Año*

**FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS**

ÍNDICE

[**Ejercicio 1:** 3](#_Toc164090078)

[**Ejercicio 2** 3](#_Toc164090079)

[**Ejercicio 4** 3](#_Toc164090080)

[**a)** 3](#_Toc164090081)

[**b)** 4](#_Toc164090082)

[**c)** 4](#_Toc164090083)

[**d)** 4](#_Toc164090084)

[**Ejercicio 5** 5](#_Toc164090085)

[**a)** 5](#_Toc164090086)

[**b)** 5](#_Toc164090087)

[**c)** 5](#_Toc164090088)

[**Ejercicio 6** 5](#_Toc164090089)

[**Ejercicio 7** 5](#_Toc164090090)

[**Ejercicio 8** 6](#_Toc164090091)

[**Ejercicio 9** 6](#_Toc164090092)

[**Ejercicio 10** 6](#_Toc164090093)

[**Ejercicio 11** 6](#_Toc164090094)

[**Ejercicio 12** 6](#_Toc164090095)

[**Ejercicio 13** 7](#_Toc164090096)

[**Ejercicio 14** 8](#_Toc164090097)

[**Ejercicio 15** 9](#_Toc164090098)

[**Ejercicio 16** 11](#_Toc164090099)

[**Ejercicio 17** 11](#_Toc164090100)

[**Ejercicio 18** 13](#_Toc164090101)

[**Ejercicio 19** 15](#_Toc164090102)

[**Ejercicio 20** 16](#_Toc164090103)

[**Ejercicio 21** 18](#_Toc164090104)

[**Ejercicio 22** 19](#_Toc164090105)

[**Fuentes Bibliográficas** 21](#_Toc164090106)

**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**

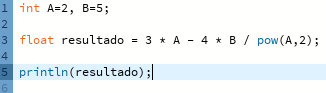
**Desarrollo del punto:**

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

6-(4\*B/4)

6-5

1



**Ejercicio 2**

**Evaluar la siguiente expresión 4 / 2 \* 3 / 6 + 6 / 2 / 1 / 5 ^ 2 / 4 \* 2**

**Desarrollo del punto:**

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

2 \* 0,5 + 3 / 1 / 25 / 4 \* 2 =

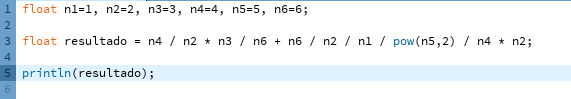
1 + 3 / 25 / 4 \* 2 =

1 + 0,12 / 4 \* 2 =

1 + 0,03 \* 2 =

1 + 0,06 =

1,03



**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 =** b2 – 4ac

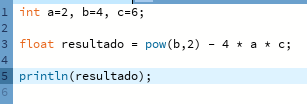
**Desarrollo del punto:** a=2, b=4, c=6

4 ^ 2 – 4 \* 2 \* 6 =

16 – 8 \* 6 =

16 – 48 =

-32



**b)**

**3 \* X ^ 4 – 5 \* X ^ 3 + X 12 – 17** = 3x4 – 5x3 + x12 -17

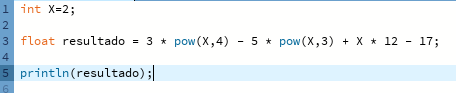
**Desarrollo del punto:** X = 2

3 \* 2 ^ 4 – 5 \* 2 ^ 3 + 2 \* 12 – 17=

3 \* 16 – 5 \* 8 + 24 – 17=

48 – 40 + 7=

15



**c)**

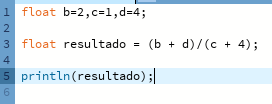
**(b + d) / (c + 4)** =

**Desarrollo del punto:** b=2, c=1, d=4

(2 + 4) / (1 +4) =

6 / 5 =

1,2



**d)**

**(x ^ 2 + y ^ 2) ^ (1 / 2)** = (x2 + y 2)1/2

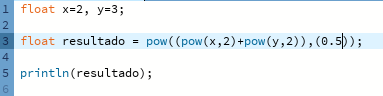
**Desarrollo del punto:** x=2,y=3

(2^2+3^2) ^ (1/2) =

(4+9) ^ (1/2) =

13 ^ (½) =

3.6055512



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

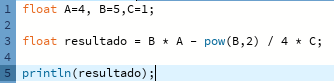
**𝑏2−4.𝑎.𝑐**

**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**

**Desarrollo del punto:**

5 \* 4 – 5 ^ 2 / 4 \* 1=

20 – 25 / 4 =

55/4 = 13,75

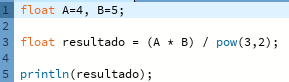
**b)**

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

**Desarrollo del punto:**

(4 \* 5) / 3 ^ 2 =

20 / 9 = 2.2222223



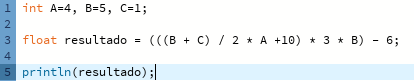
**c)**

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

**Desarrollo del punto:**

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

((6 / 2 \* 4 + 10) \* 15) – 6

((3 \* 4 + 10) \* 15) – 6

((12 + 10) \* 15) – 6

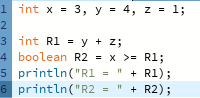
(22 \* 15) – 6

330 – 6

324

**Ejercicio 6**

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

**R1 = y+z**

= 4 + 1

= 5

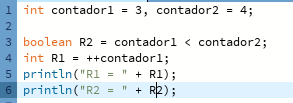
**R2 = x >= R1**

= 3 >= 5

= FALSE

**Ejercicio 7**

**Para contador1=3, contador3=4, evaluar el resultado de**

**R1 = ++contador1**

R1 = 3 + 1

R1 = 4

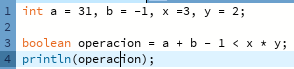
**R2 = contador1 < contador2**

R2 = 3 < 4

R2 = TRUE

**Ejercicio 8**

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

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

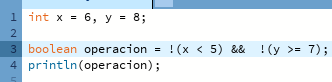
31 - 1 – 1 < 3 \* 2

30 < 6

FALSE

**Ejercicio 9**

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

**!(x < 5) && !(y >= 7)**

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

!(FALSE) && !(TRUE)

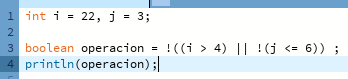
TRUE && FALSE

FALSE

**Ejercicio 10**

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

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

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

!((TRUE) || !(TRUE))

!(TRUE) || FALSE)

!TRUE

FALSE

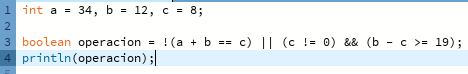
**Ejercicio 11**

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

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

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

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

!(FALSE) || (TRUE) && (FALSE)

TRUE || FALSE

VERDADERO

**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.**

**Desarrollo del punto:**

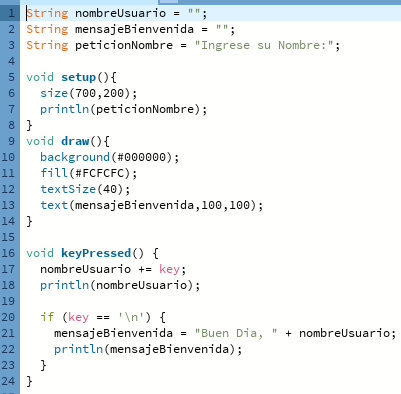
Definición del Problema: Ingresar un nombre y saludarlo

Análisis:

* Datos de Entrada: nombre de la persona
* Datos de Salida: “Buen día ……”… : nombre de la persona
* Proceso:

¿Quién debe realizar el proceso?: Un usuario

¿Cuál es el proceso que realiza …? Unir el mensaje en conjunto con el nombre del usuario

Diseño:

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Usuario |
| **VARIABLES**  nombreUsuario: String // almacena el nombre del usuario que se ingresa  mensajeBienvenida: String // almacena el mensaje de saludo final |
| **NOMBRE ALGORITMO:** saludar\_al\_usuario  **PROCESO DEL ALGORITMO**   1. **Inicio** 2. ***Mostrar***“Ingrese su nombre:” 3. ***Leer*** nombreUsuario 4. mensajeBienvenida ← “Buen día” + nombreUsuario 5. ***Mostrar*** mensajeBienvenida 6. **FIN** |

**Ejercicio 13**

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

**Desarrollo del punto:**

Definición del Problema: Calcule el perímetro y área de un rectángulo

Análisis:

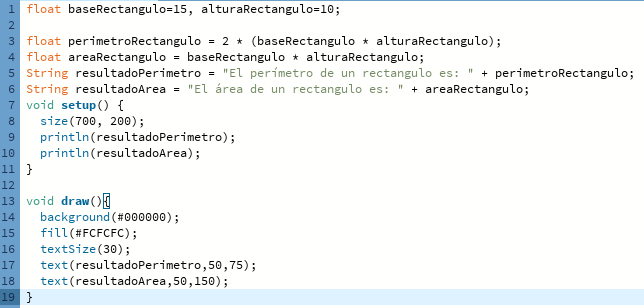
* Datos de Entrada: base rectángulo, altura rectángulo
* Datos de Salida: perímetro rectángulo, área rectángulo
* Proceso:

¿Quién debe realizar el proceso?: rectángulo

¿Cuál es el proceso que realiza …? Calcular el perímetro y el área de un rectángulo con las fórmulas correspondientes.

Diseño:

|  |
| --- |
| ENTIDAD QUE RESUELVE EL PROBLEMA: rectángulo |
| VARIABLES  alturaRectangulo, baseRectangulo: Real // almacenan los números a calcular  resultadoPerimetro: Real // almacena el resultado de  resultadoArea: Real // almacena el resultado de la multiplicación |
| NOMBRE ALGORITMO: perímetro\_rectangulo  PROCESO DEL ALGORITMO   1. **Inicio** 2. ***Leer*** alturaRectangulo 3. ***Leer*** baseRectangulo 4. resultadoPerimetro ← 2 \* alturaRectangulo + 2 \* baseRectangulo 5. resultadoArea ← alturaRectangulo \* baseRectangulo 6. ***Mostrar*** resultadoPerimetro 7. ***Mostrar*** resultadoArea 8. **FIN** |



**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**

**Desarrollo del punto:**

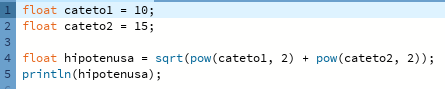
Definición del Problema: Obtenga la hipotenusa de un triángulo rectángulo

Análisis:

* Datos de Entrada: cateto1, cateto2
* Datos de Salida: hipotenusa
* Proceso:

¿Quién debe realizar el proceso?: usuario

¿Cuál es el proceso que realiza …? Calculo algebraico

****Diseño:

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** usuario |
| **VARIABLES**  cateto1, cateto2: Real // almacenan los números a calcular  hipotenusa: Real // almacena el resultado del problema |
| **NOMBRE ALGORITMO:** hipotenusa  **PROCESO DEL ALGORITMO**   1. **Inicio** 2. ***Leer*** cateto1 3. ***Leer*** cateto2 4. hipotenusa ← √ (cateto1^2 + cateto2^2) 5. ***Mostrar*** hipotenusa 6. **FIN** |

**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.**

**Desarrollo del punto:**

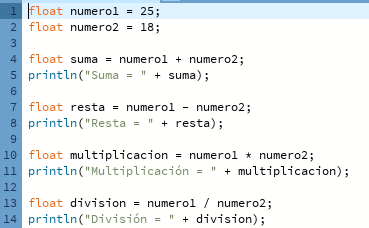
Definición del Problema: Dados dos números calcular la suma, resta, multiplicación y división de estos.

Análisis:

* Datos de Entrada: numero1, numero2
* Datos de Salida: resultado suma, resultado resta, resultado multiplicación, resultado división
* Proceso:

¿Quién debe realizar el proceso?: calculadora

¿Cuál es el proceso que realiza …? Realizar las operaciones matemáticas: sumar, restar, multiplicar y dividir los números ingresados.

Diseño:

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** calculadora |
| **VARIABLES**  numero1, numero2: Real // almacenan los números a calcular  suma, resta, multiplicación, división: Real // almacena el resultado de las operaciones |
| **NOMBRE ALGORITMO:** operación\_suma  **PROCESO DEL ALGORITMO**   1. ***Leer*** numero1 2. ***Leer*** numero2 3. suma ← numero1 + numero2 4. ***Mostrar*** suma |
| **NOMBRE ALGORITMO:** operación\_resta  **PROCESO DEL ALGORITMO**   1. ***Leer*** numero1 2. ***Leer*** numero2 3. resta ← numero1 – numero2 4. ***Mostrar*** resta |
| **NOMBRE ALGORITMO:** operación\_multiplicación  **PROCESO DEL ALGORITMO**   1. ***Leer*** numero1 2. ***Leer*** numero2 3. multiplicación ← numero1 \* numero2 4. ***Mostrar*** multiplicación |
| **NOMBRE ALGORITMO:** operación\_ división  **PROCESO DEL ALGORITMO**   1. ***Leer*** numero1 2. ***Leer*** numero2 3. división ← numero1 / numero2 4. ***Mostrar*** división |

**Ejercicio 16**

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



Definición del Problema: convertir una temperatura Fahrenheit en grados Celsius.

Análisis:

* Datos de Entrada: temperatura en Fahrenheit
* Datos de Salida: temperatura en Celsius
* Proceso:

¿Quién debe realizar el proceso?: meteorólogo

¿Cuál es el proceso que realiza …? Ejecutar una fórmula de conversión:

Diseño:

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** meteorólogo |
| **VARIABLES**  temperaturaCelsius: Real // almacenan el valor de temperatura en Celsius  temperaturaFahrenheit: Real // almacena el resultado de la temperatura en Fahrenheit y el resultado de la conversión |
| **NOMBRE ALGORITMO:** conversión\_temperatura  **PROCESO DEL ALGORITMO**   1. **Inicio** 2. ***Leer*** temperaturaCelsius 3. temperaturaFahrenheit ← (temperaturaCelsius - 32) / 1.8 4. ***Mostrar*** temperaturaFahrenheit 5. **FIN** |



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

**Desarrollo del punto:**

Definición del Problema: Determinar la distancia entre dos puntos y dibujar los objetos.

Análisis:

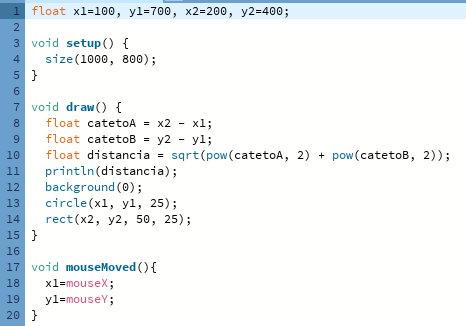
* Datos de Entrada: punto1(x1, y1), punto2(x2, y2)
* Datos de Salida: distancia
* Proceso:

¿Quién debe realizar el proceso?: usuario

¿Cuál es el proceso que realiza …? Calcular la distancia entre dos puntos mediante el teorema de Pitágoras.

Diseño:

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** usuario |
| **VARIABLES**  x1, y1, x2, y2: Real// almacenan los valores de las posiciones de los objetos  distancia: Real // almacena el resultado de la distancia entre objetos |
| **NOMBRE ALGORITMO:** distancia  **PROCESO DEL ALGORITMO**   1. **Inicio** 2. ***Leer*** x1 3. ***Leer*** x1 4. ***Leer*** x2 5. ***Leer*** y2 6. catetoA 🡨 x2 - x1 7. catetoB 🡨 y2 - y1 8. distancia ← √ (catetoA ^ 2 + catetoB ^ 2) 9. ***Mostrar*** distancia 10. ***Dibujar*** circulo (x1, y1, 25) 11. ***Dibujar*** rectángulo (x2.y2,50,25) 12. **FIN** |



**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.**

**Desarrollo del punto:**

Definición del Problema:

Diseñar un algoritmo que permita obtener las raíces de una ecuación de segundo grado.

Análisis:

* Datos de Entrada: a, b, c
* Datos de Salida: x1, x2
* Proceso:

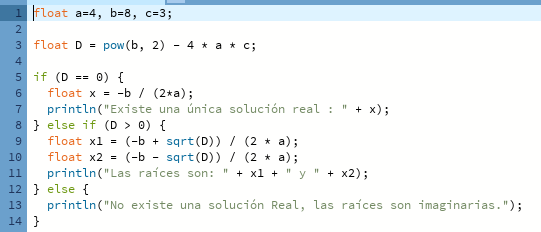
¿Quién debe realizar el proceso?: Software

¿Cuál es el proceso que realiza …? Obtener las raíces de una ecuación de segundo grado mediante el análisis de la discriminante, que está dado por la siguiente fórmula:

* Si D = 0, entonces la ecuación tiene una única solución real, que es .
* Si D > 0, entonces la ecuación tiene dos soluciones reales, que son:
* Si D < 0, entonces la ecuación no tiene soluciones reales.

Diseño:

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA**: Software |
| **VARIABLES**  a, b, c: Real // almacenan los números a calcular.  D: Real // almacena el valor de la Discriminante.  x, x1, x2: Real // almacenan las raíces calculadas.  mensaje: string // almacena el mensaje resultante |
| **NOMBRE ALGORITMO**: raíces\_ecuacion  **PROCESO DEL ALGORITMO**   1. ***Leer*** a 2. ***Leer*** b 3. ***Leer*** c 4. D ← b^2 – 4\*a\*c 5. **si** (D == 0) **entonces** 6. x1 = x2 ← -b/(2\*a) 7. mensaje ← “Existe una única solución real:” + x 8. **si\_no si (D > 0)** 9. x1 ← (-b + √D) / (2 \* a) 10. X2 ← (-b - √D) / (2 \* a) 11. mensaje ← “Las raíces son:” + x1 “,” + x2 12. **si\_no (D < 0)** 13. mensaje ← “No existe una solución Real, las raíces son imaginarias:” 14. **fin** 15. ***Escribir*** mensaje |



**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**

**Desarrollo del punto:**

Definición del Problema: Determinar la distancia entre dos puntos y dibujar los objetos.

Análisis:

* Datos de Entrada: posY, distanciaRadio
* Datos de Salida: dibujo del elipse y línea en bucle
* Proceso:

¿Quién debe realizar el proceso?: programa

¿Cuál es el proceso que realiza …? Dibujara un elipse y una línea que se moverán de arriba abajo en bucle.

Diseño:

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** programa |
| **VARIABLES**  posY: Real// almacenan el valor de la posición en Y de los objetos  distanciaRadio: Real // almacena el resultado de la distancia entre el centro del círculo con la línea  incremento: Real // almacena el valor con el cual alteraremos la posición en Y de los objetos |
| **NOMBRE ALGORITMO:** bucle  **PROCESO DEL ALGORITMO**   1. **Inicio** 2. ***Leer*** posY} 3. ***Leer*** distanciaRadio 4. ***Leer*** incremento 5. ***Dibujar*** línea (0, posY, width, posY) 6. ***Dibujar*** elipse (width/2, posY+distanciaRadio,80,80) 7. posY += incremento 8. **si** (posY>=height || posY<=0) **entonces** 9. incremento\*=-1; 10. distanciaRadio\*=-1; 11. **fin\_si** 12. **FIN** |



**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:**

**Desarrollo del punto:**

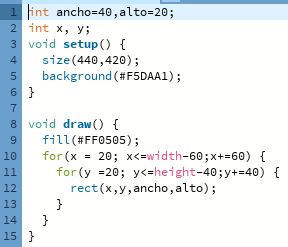
Definición del Problema: Dibujar en un lienzo varios rectángulos de idénticas medidas y que mantengan la misma distancia unos de otros.

Análisis:

* Datos de Entrada: valores de los rectángulos (ancho, alto) y la posición de ubicación de los rectángulos(x,y)
* Datos de Salida: el dibujo del lienzo con los rectángulos distribuidos correctamente.
* Proceso:

¿Quién debe realizar el proceso?: Un programa

¿Cuál es el proceso que realiza …? Crear un lienzo donde dibujaremos un rectángulo y mediante la estructura de control repetitiva “for” para que se dupliquen a lo largo del lienzo equitativamente.

Diseño:

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Usuario |
| **VARIABLES**  x, y: int // almacena los valores de la posición de los rectángulos  ancho, alto: int // almacena las medidas de los rectangulos |
| **NOMBRE ALGORITMO:** saludar\_al\_usuario  **PROCESO DEL ALGORITMO**   1. **Inicio** 2. ***Leer*** x 3. ***Leer*** y 4. ***Leer*** ancho 5. ***Leer*** alto 6. ***Hacer for*** (x = 20; x <= width - 60; x += 60) 7. ***Hacer for*** (y = 20; y <= height - 40; y += 40***)*** 8. ***Dibujar*** rectángulo (x,y,ancho,alto) 9. ***fin\_for*** 10. ***fin\_for*** 11. **fin** |

**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.**

**Desarrollo del punto:**

Definición del Problema: Utilizando la estructura de control repetitiva while() dibuje líneas que forman escalones y sobre cada borde de escalón se dibuje un punto de color rojo

Análisis:

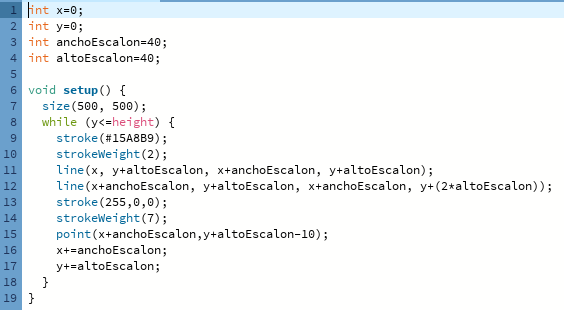
* Datos de Entrada: x, y, anchoEscalon, altoEscalon
* Datos de Salida: el dibujo de los escalones con puntos sobre cada escalon.
* Proceso:

¿Quién debe realizar el proceso?: Un programa

¿Cuál es el proceso que realiza …? Crear un lienzo donde dibujaremos mediante la estructura de control repetitiva “while” líneas formando escalones y círculos sobre estos estos mismos

Diseño:

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Usuario |
| **VARIABLES**  x, y: int // almacena los valores de la posición de los objetos  anchoEscalon, altoEscalon: int // almacena las medidas de los escalones |
| **NOMBRE ALGORITMO:** escaleras  **PROCESO DEL ALGORITMO**   1. **Inicio** 2. ***Leer*** x 3. ***Leer*** y 4. ***Leer*** anchoEscalon 5. ***Leer*** altoEscalon 6. ***Mientras*** (y<=height) **hacer** 7. ***Dibujar*** línea(x, y + altoEscalon, x + anchoEscalon, y + altoEscalon) 8. ***Dibujar*** línea ***(***x + anchoEscalon, y + altoEscalon, x + anchoEscalon, y + (2 \* altoEscalon)) 9. ***Dibujar*** punto (x + anchoEscalon, y + altoEscalon - 10) 10. *x += anchoEscalon* 11. *y += altoEscalon* 12. ***fin\_mientras*** 13. ***FIN*** |



**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.**

**Desarrollo del punto:**

Definición del Problema: Utilizando la estructura de control repetitiva while() dibuje líneas que forman escalones y sobre cada borde de escalón se dibuje un punto de color rojo

Análisis:

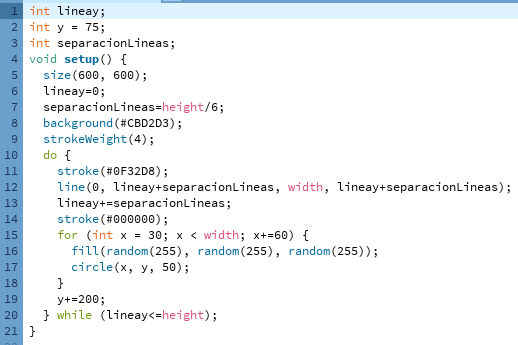
* Datos de Entrada: lineay, y, separaciónLineas
* Datos de Salida: círculos sobre cada línea de por medio es decir en la línea 1 se dibujan círculos con distanciamiento, en la línea 2 no
* Proceso:

¿Quién debe realizar el proceso?: Un programa

¿Cuál es el proceso que realiza …? Defina un lienzo, 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.

Diseño:

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Usuario |
| **VARIABLES**  y: int // almacena el valor de la posición en Y de los circulos  lineay: int // almacena el valor de la posición en Y de las líneas  separacionLineas: int // almacena el valor de la distancia entre las lineas |
| **NOMBRE ALGORITMO:** cirulos\_lineas  **PROCESO DEL ALGORITMO**   1. **Inicio** 2. ***Leer*** y 3. ***Leer*** *lineay* 4. ***Leer*** separacionLineas 5. ***Hacer*** 6. ***Dibujar*** línea (0, lineay + separacionLineas, width, lineay + separacionLineas) 7. lineay+=separacionLineas 8. ***hacer* *for*** (int x = 30; x < width; x+=60) 9. ***Dibujar*** circulo(x, y, 50) 10. ***fin\_for*** 11. y += 200; 12. ***fin\_hacer*** 13. ***FIN*** |



# **Fuentes Bibliográficas**

PDFs y videos de el Aula Virtual:

<https://virtual.unju.edu.ar/course/view.php?id=2537&section=2>

Videos de YouTube:

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