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

**FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS**

Trabajo Práctico N°1:

Operadores – Metodología de Programación

Jurado, Leandro Ariel – TUV000745

*Profesores:*

*Mg. Ing. Ariel Alejandro Vega*

*Ing. Carolina Cecilia Apaza*

*Año*

Índice

Ejercicio\_01 3

Ejercicio\_03 4

Ejercicio\_04 5

Ejercicio\_05 6

Ejercicio\_06 7

Ejercicio\_09 8

Ejercicio\_10 10

Ejercicio\_13 11

Ejercicio\_14 12

Ejercicio\_15 13

Ejercicio\_16 14

Ejercicio\_17 18

Ejercicio\_18 19

Ejercicio\_19 21

Ejercicio\_20 24

Ejercicio\_21 26

Fuentes bibliográficas 29

**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 01: 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 processing | Captura Resultado |
|  |  |

Ejercicio 02: Evaluar la siguiente expresión 4 / 2 \* 3 / 6 + 6 / 2 / 1 / 5 ^ 2 / 4 \* 2

|  |  |
| --- | --- |
| Forma Aritmética | Forma Algebraica |
| **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 |  |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

Ejercicio 03: 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=4, b=6, c=2

|  |  |
| --- | --- |
| Forma Aritmética | Forma Algebraica |
| b ^ 2 – 4 \* a \* c  (6 ^ 2) – (4 \* 4 \* 2)  36 – 32  4 |  |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

b) 3 \* X ^ 4 – 5 \* X ^ 3 + X 12 – 17 X=3

|  |  |
| --- | --- |
| Forma Aritmética | Forma Algebraica |
| 3 \* X ^ 4 – 5 \* X ^ 3 + X 12 – 17  (3 \* (3 ^ 4)) – (5 \* (3 ^ 3)) + (3 \* 12) – 17  243 – 135 + 36 – 17  127 |  |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

c) (b + d) / (c + 4) b=5, c=4, d=7

|  |  |
| --- | --- |
| Forma Aritmética | Forma Algebraica |
| (b + d) / (c + 4)  (5 + 7) / (4 + 4)  12 / 8  1.5 |  |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

d) (x ^ 2 + y ^ 2) ^ (1 / 2) x=8, y=6

|  |  |
| --- | --- |
| Forma Aritmética | Forma Algebraica |
| (x ^ 2 + y ^ 2) ^ (1 / 2)  (8 ^ 2 + 6 ^ 2) ^ (1 / 2)  100 ^ (1/2)  10 |  |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

Ejercicio 04: 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

|  |  |
| --- | --- |
| Forma Aritmética | Forma Algebraica |
| B \* A – B ^ 2 / 4 \* C  5 \* 4 – ((5 ^ 2) / 4) \* 1  20 – 6.25  13.75 |  |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

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

|  |  |
| --- | --- |
| Forma Aritmética | Forma Algebraica |
| (A \* B) / 3 ^ 2  (4 \* 5) / 3 ^ 2  20 / 9  2.2….. |  |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

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

|  |  |
| --- | --- |
| Forma Aritmética | Forma Algebraica |
| (((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 |  |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

Ejercicio 05: Para x=3, y=4; z=1, evaluar el resultado de:

R1 = y + z

R2 = x >= R1

|  |
| --- |
| R1 = 4 + 1 = 5  R2 = 3 >= R1  Falso |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

Ejercicio 06: Para contador1=3, contador2=4, evaluar el resultado de:

R1 = ++contador1

R2 = contador1 < contador2

|  |
| --- |
| R1 = 4  R2 = 4 < 4  Falso |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

Ejercicio 07: 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  29 < 6  Falso |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

Ejercicio 08: Para x=6, y=8, evaluar el resultado de !(x<5) CC !(y>=7)

|  |
| --- |
| !(x<5) && !(y>=7)  !(6<5) && !(8>=7)  Falso && Falso  Falso |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

Ejercicio 09: Para i=22, j=3, evaluar el resultado de !((i>4) || !(j<=6))

|  |
| --- |
| !((i>4) || !(j<=6))  !((22>4) || !(3<=6))  !(verdadero || falso)  !(verdadero)  Falso |

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

Ejercicio 10: 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) && (b-c>=19)  !(34+12==8) || (8!=0)&&(12-8>=19)  !(46==8) || (8!=0)&&(4>=19)  verdadero || verdadero && falso  verdadero|| falso  Verdadero |

|  |  |
| --- | --- |
| Captura processing | Captura 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 11: 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: Realizar un algoritmo donde el usuario escriba su nombre y reciba un saludo

ANALISIS

Datos de Entrada:

nombre: cadena

Datos de Salida

Saludo: cadena de texto

Proceso

Saludar con el nombre ingresado

DISEÑO

|  |
| --- |
| Entidad: Algoritmo |
| Variables:  nombre: cadena  saludo: cadena de texto |
| Nombre del Algoritmo: saludar\_nombre  **Proceso del Algoritmo**   1. Inicio 2. Mostrar “Escribe tu nombre” 3. Leer nombre 4. saludo 🡨 "Hola " + nombre + "¿Cómo estás?" 5. Mostrar saludo 6. Fin |

CODIFICACION

|  |
| --- |
| Captura processing |
|  |

|  |  |  |
| --- | --- | --- |
| Mensa que se muestra al Iniciar el programa | Al escribir el programa va leyendo cada tecla presionada y la registra, al presionar enter se presenta el saludo | En la ventana del dibujo se aprecia como aparece el saludo |
|  |  |  |

Ejercicio 12: Será común resolver problemas utilizando variables. Calcule el perímetro y área de un rectángulo dada su base y su altura.

Descripción del problema: Obtener el perímetro y el área de un rectángulo

ANALISIS

Datos de Entrada:

altura, base: enteros

Datos de Salida

Perímetro, área: real

Proceso

Calcular el perímetro y el área con sus respectivas formulas

DISEÑO

|  |
| --- |
| Entidad: Persona |
| Variables:  altura: entero  base: entero  perimetro: real  area: real  preimetro\_area: real |
| Nombre del Algoritmo: perimetro\_area\_Rect  **Proceso del Algoritmo**   1. Inicio 2. altura 🡨 12 3. base 🡨 8 4. perimetro 🡨 2\*(base + altura) 5. area 🡨 base \* altura 6. perimetro\_area 🡨 “El perímetro de su rectángulo es: “ +perimetro+ “y el 7. área es: “+area 8. Mostrar perimetro\_area 9. Fin |

CODIFICACION

|  |
| --- |
| Captura processing |
|  |
| Captura Resultado |
|  |

Ejercicio 13: 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.

Descripción del problema: Obtener la hipotenusa de un triángulo rectángulo

ANALISIS

Datos de Entrada:

catetoA, catetoB: enteros

Datos de Salida

hipotenusa: real

Proceso

Calcular la hipotenusa con la formula correspondiente

DISEÑO

|  |
| --- |
| Entidad: Persona |
| Variables:  catetoA: entero  catetoB: entero  hipotenusa: real |
| Nombre del Algoritmo: resultado  **Proceso del Algoritmo**   1. Inicio 2. Leer catetoA 3. Leer catetoB 4. hipotenusa 🡨 (catetoA^2 + catetoB^2) ^ (0.5) 5. Mostrar hipotenusa 6. Fin |

CODIFICACION

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  |  |

Ejercicio 14: 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.

Descripción del problema: Obtener la hipotenusa de un triángulo rectángulo

ANALISIS

Datos de Entrada:

num1, num2: enteros

Datos de Salida

suma, resta, mult, div: real

Proceso

Calcular las operaciones correspondientes

DISEÑO

|  |
| --- |
| Entidad: calculadoraBasica |
| Variables:  num1: entero  num2: entero  suma: entero  resta: entero  mult: entero  div: entero |
| Nombre del Algoritmo: calculadora\_simple  **Proceso del Algoritmo**   1. Inicio 2. Leer num1 3. Leer num2 4. suma 🡨 num1 + num2 5. Mostrar 🡨 “El resultado de la suma es: “+ suma 6. resta 🡨 num1 - num2 7. Mostrar 🡨 “El resultado de la resta es: “+ resta 8. mult 🡨 num1 \* num2 9. Mostrar 🡨 “El resultado de la multiplicación es: “+ mult 10. div 🡨 num1 / num2 11. si (num2 != 0) entonces 12. Mostrar 🡨 “El resultado de la división es: “+ div 13. si\_no 14. Mostrar 🡨 “La división por cero no está definida” 15. Fin |

CODIFICACION

|  |
| --- |
| Captura processing |
|  |
| Captura Resultado |
|  |

Ejercicio 15: 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.

Descripción del problema: Convertir una temperatura Fahrenheit en grados Celsius

ANALISIS

Datos de Entrada:

tempFa: real

Datos de Salida

tempCe: real

Proceso

Convertir la temperatura en Fahrenheit a Celsius con su formula

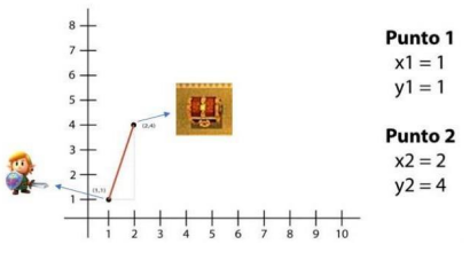
DISEÑO

|  |
| --- |
| Entidad: Persona |
| Variables:  tempFa: real  tempCe: real |
| Nombre del Algoritmo: conversor\_a\_temperatura\_celsius  **Proceso del Algoritmo**   1. Inicio 2. Leer temFa 3. tempCe 🡨 (tempFa – 32) / 1.8 4. Mostrar “Temperatura en grados Celsius es: “+tempCe 5. Fin |

CODIFICACION

|  |
| --- |
| Captura processing |
|  |
| Captura Resultado |
|  |

Ejercicio 16: 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.

Descripción del problema: Hallar la distancia entre Link y el tesoro

ANALISIS

Datos de Entrada:

linkX, linkY, tesoroX; tesoroY: real

Datos de Salida

distEntreLinkTe: real

Proceso

Calcular la hipotenusa con la formula correspondiente

DISEÑO

|  |
| --- |
| Entidad: Persona |
| Variables:  x1: real  y1: real  x2: real  y2: real  catetoX: real  catetoY: real  distancia: real  umbral: real |
| Nombre del Algoritmo: distancia\_Link\_tesoro  **Proceso del Algoritmo**   1. Inicio 2. Leer x1 3. Leer x2 4. Leer y1 5. Leer y2 6. umbral 🡨 30 7. catetoX 🡨 x2 - x1 8. catetoY 🡨 y2 - y1 9. distancia 🡨 ((catetoX)^2 + (catetoY)^2)^0.5 10. Mostrar “Distancia: “+distancia 11. si (distancia <= umbral) entonces 12. Mostrar cambio de color en el tesoro 13. fin\_si 14. Fin |

CODIFICACION

|  |
| --- |
| Captura processing |
|  |
| Captura pantalla |
|  |

Ejercicio 17: 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.

Descripción del problema: Hallar las rices de una ecuación de segundo grado

ANALISIS

Datos de Entrada:

a, b, c: real

Datos de Salida

R1, R2: real

Proceso

Hallar las raíces con la formula correspondiente

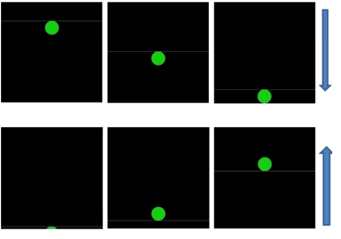
DISEÑO

|  |
| --- |
| Entidad: Persona |
| Variables:  a: real  b: real  c: real  R1: real  R2: real  discriminante: real |
| Nombre del Algoritmo: hallar\_raices  **Proceso del Algoritmo**   1. Inicio 2. Leer a 3. Leer b 4. Leer c 5. discriminante 🡨 b^2 – 4\*a\*c 6. si (discriminante > 0) entonces 7. R1 🡨 (-b + (discriminante))^0.5 /(2\*a) 8. R2 🡨 (-b - (discriminante))^0.5 /(2\*a) 9. mostrar “Las raíces son: ” + R1 + “ y ” + R2 10. si\_no si (discriminante == 0) entonces 11. raiz ← -b / (2\*a) 12. mostrar “La raíz doble es: “ + raiz 13. si\_no 14. mostrar “No hay raíces reales” 15. fin\_si 16. Fin |

CODIFICACION

|  |  |
| --- | --- |
| Captura processing | Captura Resultado |
|  | Valores a=1, b=0, c=-1    Valores a=2, b=-4, c=2    Valores a=1, b=0, c=1 |

Ejercicio 18: 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



Descripción del problema: Dibuja una línea con un circulo que suba y baje

ANALISIS

Datos de Entrada

linea, elipse, direc: entero

Datos de Salida

Línea y circulo en bucle

Proceso

Dibujar una línea y un círculo en movimiento

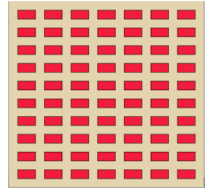
DISEÑO

|  |
| --- |
| Entidad: Lienzo |
| Variables  linea: entero  elipse: entero  direc: entero |
| Nombre Algoritmo: linea\_circulo\_mov  Algoritmo   1. Inicio 2. Leer linea 3. Leer elipse 4. Leer direc 5. anchoLienzo ⃪ 400 6. altoLiezo ⃪ 400 7. para i 🡨 o hasta alto incremento 1 hacer 8. linea 🡨 linea + direc 9. fin\_para 10. si ((linea >= anchoLienzo) O (linea <= 0)) entonces 11. direc 🡨 direc \* (-1) 12. fin\_si 13. Mostrar linea 14. Dibujar linea en (direc, linea, altoLienzo, linea) 15. Dibujar circulo en (altoLienzo/2, (direc==1) ? linea + elipse : linea – elipse, 80, 80) 16. Fin |

CODIFICACION

|  |
| --- |
| Captura processing |
|  |
| Captura Resultado |
|  |

Ejercicio 19: 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:



Descripción del problema: Dibuja en el lienzo rectángulos usando estructuras iterativas

ANALISIS

Datos de Entrada

coordenadasRect: coordenadas cartesianas

ancho, alto, distEntreRect: entero

Datos de Salida

Los rectángulos dibujados

Proceso

Dibujar los rectángulos

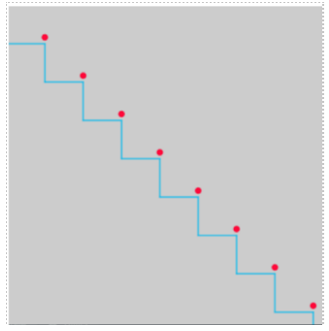
DISEÑO

|  |
| --- |
| Entidad: Lienzo |
| Variables  coordenadaRect: Coordenadas  ancho, alto, distEntreRect: Enteros  anchoLienzo, altoLienzo: Enteros |
| Nombre Algoritmo: dibujar rectangulos  **Proceso del algoritmo**   1. Inicio 2. anchoLienzo ⃪ 440 3. altoLienzo ⃪ 420 4. distanciEntreRect ⃪ 20 5. ancho ⃪ 40 6. alto ⃪ 20 7. Para x 🡨 coordenadasRect.x hasta anchoLienzo con paso (ancho + distEntreRect) hacer 8. Para y 🡨 coordenadasRect.y hasta altoLienzo con paso (alto + distEntreRect) hacer 9. Dibujar un rectángulo en (x,y,ancho,alto) 10. Fin Para 11. Fin Para 12. Fin |

CODIFICACION

|  |
| --- |
| Captura processinng |
|  |
| Captura pantalla |
|  |

Ejercicio 20: Utilizando la estructura de control repetitiva while() dibuje la siguiente imagen utilizando líneas que forman escalones y sobre cada borde de escalón se dibuje un punto de color rojo



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.

Descripción del problema: Hallar las rices de una ecuación de segundo grado

ANALISIS

Datos de Entrada:

distancia: entero

puntoA, puntoB, puntoC, puntoD: coordenadas

Datos de Salida

El dibujo de la línea horizontal

El dibujo de la línea vertical

El dibujo del punto

Proceso

Confeccionar escalones con puntos en los bordes

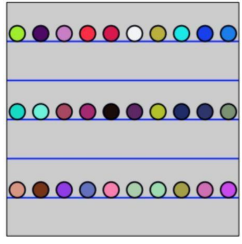
DISEÑO

|  |
| --- |
| Entidad: Persona |
| Variables:  distancia: entero  puntoA, puntoB, puntoC, puntoD: coordenadas |
| Nombre del Algoritmo: escalones\_circulos  **Proceso del Algoritmo**   1. Inicio 2. anchoLienzo 🡨 500 3. altoLienzo 🡨 500 4. distancia 🡨 60 5. mientras (puntoA.y <= anchoLienzo) hacer 6. dibujar línea horizontal en (puntoA.x,puntoA.y,puntoB.x,puntoB.y) 7. dibujar línea vertical en (puntoB.x,puntoB.y,puntoC.x,puntoC.y) 8. dibujar circulo en (pintoD.x,puntoD.y) 9. puntoA.x 🡨 puntoC.x 10. puntoA.y 🡨 puntoC.y 11. fin\_mientras 12. Fin |

CODIFICACION

|  |
| --- |
| Captura processing |
|  |
| Captura Pantalla |
|  |

Ejercicio 21: Utilizando la estructura de control repetitiva do-while. Replique la siguiente imagen



La imagen debe ser construida desde la función setup(). Defina el tamaño del lienzo en size(600,600), verticalmente se divide el lienzo en 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.

Descripción del problema: Hallar las rices de una ecuación de segundo grado

ANALISIS

Datos de Entrada:

distCirculo: entero

lineaX, lineaY, circuloX, circuloY: enteros

Datos de Salida

El dibujo de las líneas

El dibujo de los circulos

Proceso

Confeccionar líneas con círculos de colores aleatorios

DISEÑO

|  |
| --- |
| Entidad: Persona |
| Variables:  distCirculo: entero  lineaX, lineaY, circuloX, circuloY: entero  anchoLienzo, altoLienzo: entero |
| Nombre del Algoritmo: escalones\_circulos  **Proceso del Algoritmo**   1. Inicio 2. anchoLienzo 🡨 600 3. altoLienzo 🡨 600 4. lineaX 🡨 0 5. lineaY 🡨 100 6. distCirculo 🡨 30 7. circuloY 🡨 75 8. hacer 9. circuloX 🡨 distCirculo 10. hacer 11. dibujar linea en (lineaX, lineaY, anchoLienzo, lineaY) 12. dibujar circulo en (cirucloX, circuloY, 50, 50) 13. circuloX 🡨 circuloX + distCirculo\*2 14. fin\_hacer 15. mientras (circuloX <= anchoLienzo) 16. lineaY 🡨 lineaY + 100 17. circuloY 🡨 circuloY + 200 18. fin\_hacer 19. mientras (lineaY <= altoLienzo) 20. Fin |

CODIFICACION

|  |
| --- |
| Captura processing |
|  |
| Captura Pantalla |
|  |

Fuentes bibliográficas

Convertir Expresiones Matemática en Expresiones Algorítmicas

https://youtu.be/zVAf1NnsM6I?si=GNQYyfnV3wcvGxM9

<https://youtu.be/rGLatJ45_Tw?si=qgnY1_O8sxWsTa37>

Estructuras de control selectivas en Processing

<https://www.youtube.com/watch?v=Nr3NdAxjqsE>

Aplicaciones de las matemáticas en Processing

<https://www.youtube.com/watch?v=LUfoF-tG8xw>

Expresiones lógicas y estructuras selectivas

<https://www.youtube.com/watch?v=EAclc8NcnOo>

Estructura while en Processing

<https://www.youtube.com/watch?v=dysxlptjAx0>

Estructura do while en Processing

<https://www.youtube.com/watch?v=4RQV0gMl8y8>

Estructura for en Processing

<https://www.youtube.com/watch?v=kApd8u4l0tU>