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

*Profesores:*

*Mg. Ing. Ariel Alejandro Vega*

*Ing. Carolina Cecilia Apaza*

*Año 2024*

**FUNDAMENTOS DE PROGRAMACION ORIENTADA A OBJETOS**

TRABAJO PRACTICO N°1

Quispe Tamara Melisa - 000679

INDICE

Contenido

[SECCIÓN EXPRESIONES ARITMÉTICAS Y LÓGICAS 3](#_Toc164103930)

[Ejercicio 1: 3](#_Toc164103931)

[Ejercicio 2 3](#_Toc164103932)

[Ejercicio 4: 3](#_Toc164103933)

[Ejercicio 5: 5](#_Toc164103934)

[Ejercicio 6: 6](#_Toc164103935)

[Ejercicio 7: 7](#_Toc164103936)

[Ejercicio 8: 7](#_Toc164103937)

[Ejercicio 9: 7](#_Toc164103938)

[Ejercicio 10 8](#_Toc164103939)

[Ejercicio 11 8](#_Toc164103940)

[SECCION: ANALISIS\_DISEÑO Y CODIFICACIÓN DE ALGORITMOS- APLICACIÓN DE ESTRUCTURASDE CONTROL 9](#_Toc164103941)

[Ejercicio 12 9](#_Toc164103942)

[Ejercicio 13 11](#_Toc164103943)

[Ejercicio 14: 12](#_Toc164103944)

[Ejercicio 15: 13](#_Toc164103945)

[Ejercicio 16: 15](#_Toc164103946)

[Ejercicio 17: 16](#_Toc164103947)

[Ejercicio 18: 19](#_Toc164103948)

[Ejercicio 19: 22](#_Toc164103949)

[Ejercicio 20: 24](#_Toc164103950)

[EJERCICIO 21: 26](#_Toc164103951)

[Ejercicio 22: 29](#_Toc164103952)

[FUENTES BIBLIOGRÁFICAS 32](#_Toc164103953)

# 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:

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

(3 \* 2) - (4 \* 5 / (2 ^ 2))

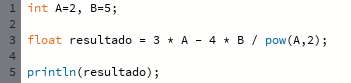
(3 \* 2)– (4 \* 5 / 4)

6 - (20/4)

6- 5

1

Captura de Processing:



**Ejercicio 2:** Evaluar la siguiente expresión:

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

Resolución:

4 / 2 \* 3 / 6 + 6 / 2 / 1 / 25 / 4 \* 2

2 \* 3 / 6 + 3/25/4\*2

6/6 + 0.12 / 4 \* 2

1 + 0.03 \* 2

1 + 0.06

1.06

|  |  |
| --- | --- |
| Captura de Processing: | Resultado: |
| Texto, Carta  Descripción generada automáticamente | Interfaz de usuario gráfica, Texto, Aplicación  Descripción generada automáticamente |

**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

 Resolución:

a) **Para a= 1, b=4 y c=2**

|  |  |
| --- | --- |
| **Resolver expresión aritmética**  b ^ 2 – 4 \* a \* c  4 ^ 2 – 4 \* 1 \* 2  16 – 4 \* 1 \* 2  16 – 8  8 | **Expresión algebraica**  b2 – 4.a.c |

|  |  |
| --- | --- |
| Captura de Processing: | Resultado: |
| Texto  Descripción generada automáticamente con confianza media | Una captura de pantalla de un celular con texto e imagen  Descripción generada automáticamente con confianza media |

b) **Para X = 3**

|  |  |
| --- | --- |
| **Resolver la expresión aritmética**  3 \* X ^ 4 – 5 \* X ^ 3 + X 12 – 17  3 \* 3 ^ 4 – 5 \* 3 ^ 3 + 3 \* 12 - 17  3 \* 81 – 5 \* 27 + 3 \* 12 -17  243 – 135 + 36 – 17  127 | **Expresión algebraica**  3. X4 – 5. X3 + X. 12 - 17 |

|  |  |
| --- | --- |
| CAPTURA DEL PROCESSING: | RESULTADO: |
| Texto  Descripción generada automáticamente con confianza media | Un conjunto de letras blancas en un fondo blanco  Descripción generada automáticamente con confianza media |

c) **Para b=23, c= 2 y d =19**

|  |  |
| --- | --- |
| **Resolver la expresión aritmética**  (b + d) / (c + 4)  (23 + 19) / (2 + 4)  (42) / (6)  42 / 6  7 | **Expresión algebraica** |

|  |  |
| --- | --- |
| CAPTURA DEL PROCESSING | RESULTADO: |
|  | Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamente |

d) **Para x= 4, y=3**

|  |  |
| --- | --- |
| **Resolver la expresión aritmética**  (x ^ 2 + y ^ 2) ^ (1 / 2)  (4 ^ 2 + 3 ^ 2) ^ (1/2)  (16 + 9) ^ (0.5)  25 ^ 0.5  5 | **Expresión algebraica** |

|  |  |
| --- | --- |
| CAPTURA DEL PROCESSING: | RESULTADO: |
|  | Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamente |

**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**

Resolución:

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

|  |  |
| --- | --- |
| Captura del Processing: | Resultado |
|  | Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamente |

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

Resolución:

(4 \* 5) / 3 ^ 2

20 / 9

2.2222

|  |  |
| --- | --- |
| Captura del Processing: | Resultado: |
|  | Interfaz de usuario gráfica  Descripción generada automáticamente |

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

Resolución:

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

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

((3\*4+10) \* 3 \* 5) - 6

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

(22 \* 3 \* 5) - 6

(66 \* 5) -6

330 – 6

324

|  |  |
| --- | --- |
| CAPTURA DEL PROCESSING: | RESULTADO: |
| Imagen que contiene Texto  Descripción generada automáticamente | Interfaz de usuario gráfica, Aplicación, Sitio web  Descripción generada automáticamente |

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

* **R1 = y + z**

Resolución:

R1= y + z

R1= 4 + 1

R1= 5

* **R2 = x >= R1**

Resolución:

R2 = x >= R1

R2 = 3 >= 5

R2 = False

|  |  |
| --- | --- |
| CAPTURA EN PROCESSING | RESULTADO |
| Texto  Descripción generada automáticamente | Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamente |

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

* **R1 = ++contador1**
* **R2 = contador1 < contador2**

Resolución:

R1= ++3

**R1= 4**

R2= 4 < 4

**R2= False**

|  |  |
| --- | --- |
| Captura de Processing: | Resultado: |
|  | Imagen de la pantalla de un celular con letras  Descripción generada automáticamente con confianza baja |

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

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

Resolución:

a+b-1 < x\*y

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

31-1-1 < 6

29 < 6

**False**

|  |  |
| --- | --- |
| CAPTURA DEL PROCESSING: | RESULTADO: |
|  | Interfaz de usuario gráfica, Texto, Aplicación  Descripción generada automáticamente |

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

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

Resolución:

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

!(Falso)CC !(Verdadero)

Verdadero CC Falso

**Falso**

|  |  |
| --- | --- |
| CAPTURA DE PROCESSING: | RESULTADO: |
| Imagen que contiene Texto  Descripción generada automáticamente | Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamente |

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

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

Resolución:

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

! ((Verdadero) ||! (Verdadero))

(False || False)

**False**

|  |  |
| --- | --- |
| Captura del Processing: | Resultado: |
| Imagen que contiene Texto  Descripción generada automáticamente | Interfaz de usuario gráfica, Texto, Aplicación  Descripción generada automáticamente |

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

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

Resolución:

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

! (46==8) || (Verdadero)CC (4>=19)

! (Falso) || (Verdadero) CC (Falso)

Verdadero||Verdadero CC Falso

Verdadero || Falso

**Verdadero**

|  |  |
| --- | --- |
| CAPTURA DEL PROCESSING: | RESULTADO: |
| Imagen que contiene Diagrama  Descripción generada automáticamente | Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamente |

# SECCION: ANALISIS\_DISEÑO Y CODIFICACIÓN DE ALGORITMOS- APLICACIÓN DE ESTRUCTURASDE 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.

**FASE DE ANÁLISIS**

**Definición del problema:** Pedir por teclado al usuario un nombre y saludarlo.

**Análisis:**

* Datos de Entrada: nombre de la persona.

nombreUsuario: String

* 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 de saludo con el nombre de la persona o usuario.

mensajeSaludo: “Buen día” + nombreUsuario

**FASE DE DISEÑO**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Usuario |
| **VARIABLES**  nombreUsuario: String //almacena el nombre del usuario  mensajeSaludo: String //almacena el mensaje de saludo |
| **NOMBRE DEL ALGORITMO:** saludar\_usuario  **PROCESO DEL ALGORITMO**   1. *Mostrar* “Ingrese su nombre: “ 2. *Leer* nombreUsuario 3. mensajeSaludo ← “Buen día “+ nombreUsuario 4. *Mostrar* mensajeSaludo |

***CAPTURA DEL PROCESSING:***

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

***RESULTADO:***

|  |  |
| --- | --- |
| Texto  Descripción generada automáticamente | Interfaz de usuario gráfica, Texto, Aplicación  Descripción generada automáticamente |

***OTRA FORMA MÁS SENCILLA:***

|  |  |
| --- | --- |
| Interfaz de usuario gráfica, Texto, Aplicación  Descripción generada automáticamente | Interfaz de usuario gráfica, Texto, Aplicación, Sitio web  Descripción generada automáticamente |

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

**FASE DEL ANÁLISIS**

**Descripción del problema:** Calcular el perímetro y área de un rectángulo

**Análisis:**

* Datos de Entrada: los lados del rectángulo: base del rectángulo, altura del rectángulo.
* Datos de Salida: el perímetro y área de un rectángulo
* Proceso:

**¿Quién debe realizar el proceso?:** El rectángulo

**¿Cuál es el proceso que realiza?:** Realiza las operaciones necesarias para calcular el perímetro y el área de un rectángulo utilizando los lados dados.

Ecuación para calcular el perímetro:

resultadoPerimetro = 2 \* alturaRect + 2 \* baseRect

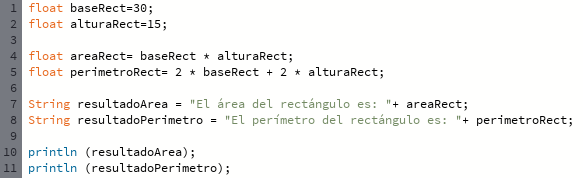
Ecuación para calcular el área:

resultadoArea= baseRect\*alturaRect

**FASE DEL DISEÑO**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Rectángulo |
| **VARIABLES**  baseRect: Real //almacena la base del rectángulo  alturaRect: Real //almacena la altura del rectángulo  areaRect: Real //almacena el área del rectángulo  perimetroRect: Real //almacena el perímetro del rectángulo  resultadoPerimetro: String // mensaje que muestra el resultado del perímetro  resultadoArea: String // mensaje que muestra el resultado del área |
| **NOMBRE DEL ALGORITMO:** calcular\_perimetro\_y\_area  **PROCESO DEL ALGORITMO**   1. inicio 2. *Leer* baseRect 3. *Leer* alturaRect 4. areaRect ← baseRect \* alturaRect 5. perimetroRect ← 2\*baseRect + 2 \*alturaRect 6. resultadoArea ← “El área del rectángulo es: “+ areaRect 7. resultadoPerimetro ← “El perímetro del rectángulo es: “+ perimetroRect 8. *Mostrar* resultadoArea 9. *Mostrar* resultadoPerimetro 10. Fin |

***CAPTURA DEL PROCESSING:***



***RESULTADO:***

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**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

Imagen que contiene Diagrama

Descripción generada automáticamente

**FASE DE ANÁLISIS**

**Descripción del problema:** Calcular la hipotenusa de un triángulo rectángulo conociendo sus catetos.

**Análisis:**

* Datos de Entrada: los catetos del triángulo rectángulo: catetoOpuesto y catetoAdyacente.
* Datos de Salida: la hipotenusa del triángulo rectángulo.
* Proceso:

¿Quién debe realizar el proceso?: El triángulo rectángulo

¿Cuál es el proceso que realiza?: Dados los catetos realiza las correspondientes operaciones con la siguiente ecuación:

resultadoHipotenusa = (catetoOpuesto^2 + catetoAdyacente^2) ^ (0.5)

**FASE DE DISEÑO**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Triángulo Rectángulo |
| **VARIABLES**  catetoOpuesto, catetoAdyacente, resultadoHipotenusa: Real |
| **NOMBRE DEL ALGORITMO:** calcular\_hipotenusa  **PROCESO DEL ALGORITMO**   1. Inicio 2. *Leer* catetoOpuesto 3. *Leer* catetoAdyacente 4. resultadoHipotenusa ← (catetoOpuesto^2 + catetoAdyacente^2) ^ (0.5) 5. *Mostrar* “El resultado de la hipotenusa es: “+ resultadoHipotenusa 6. Fin |

***CAPTURA DEL PROCESSING:***

Texto

Descripción generada automáticamente

***RESULTADO:***

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**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.

**FASE DE ANÁLISIS**

**Descripción del problema:** Dados dos números calcular la suma, resta, multiplicación y división.

**Análisis:**

* Datos de Entrada: dos números, numero1, numero2
* Datos de Salida: resultado de la suma, de la resta, de la multiplicación y de la división.
* Proceso:

¿Quién debe realizar el proceso?: Calculadora

¿Cuál es el proceso que realiza?: Calcula las operaciones de suma, resta, multiplicación y división.

Ecuación de la suma= numero1+numero2

Ecuación de la resta= numero1-numero2

Ecuación de la multiplicación= numero1 \* numero2

Ecuación de la división= numero1/numero2

**FASE DE DISEÑO**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Calculadora |
| **VARIABLES**  numero1, numero2: Real  suma, resta, multiplicación, división: Real |
| **NOMBRE DEL ALGORITMO:** calcular\_suma  **PROCESO DEL ALGORITMO**   1. Inicio 2. *Leer* numero1 3. *Leer* numero2 4. suma ← numero1 + numero2 5. *Mostrar* “La suma de estos dos números es: “+ suma 6. Fin |
| **NOMBRE DEL ALGORITMO:** calcular\_resta  **PROCESO DEL ALGORITMO**   1. Inicio 2. *Leer* numero1 3. *Leer* numero2 4. resta ← numero1 - numero2 5. *Mostrar* “La resta de estos dos números es: “+ resta 6. Fin |
| **NOMBRE DEL ALGORITMO:** calcular\_multiplicacion  **PROCESO DEL ALGORITMO**   1. Inicio 2. *Leer* numero1 3. *Leer* numero2 4. multiplicación ← numero1 \* numero2 5. *Mostrar* “La multiplicación de estos dos números es: “+ multiplicación 6. Fin |
| **NOMBRE DEL ALGORITMO:** calcular\_division  **PROCESO DEL ALGORITMO**   1. Inicio 2. *Leer* numero1 3. *Leer* numero2 4. **si** (numero2==0) **entonces** 5. Mostrar “No se puede dividir por cero” 6. **fin\_si** 7. división ← numero1 / numero2 8. *Mostrar* “La división de estos dos números es: “+ división 9. Fin |

***CAPTURA DEL PROCESSING:***

|  |
| --- |
| Texto  Descripción generada automáticamente |
|  |

***RESULTADO***

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**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



**FASE DE ANÁLISIS**

**Descripción del problema:** Convertir una temperatura Fahrenheit en grados Celsius

**Análisis:**

* Datos de Entrada: temperaturaFahrenheit: Real
* Datos de Salida: temperaturaCelsius: Real
* Proceso:

¿Quién debe realizar el proceso?: Calculadora

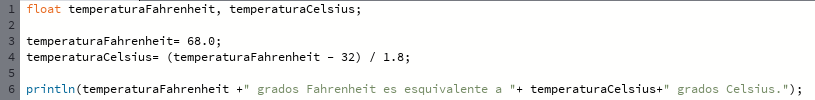
¿Cuál es el proceso que realiza?: Convierte el grado Fahrenheit a grado Celsius con la siguiente formula:

temperaturaCelsius= (temperaturaFahrenheit-32) / 1.8

**FASE DE DISEÑO**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Calculadora |
| **VARIABLES**  temperaturaFahrenheit: Real //almacena el grado Fahrenheit  temperaturaCelsius: Real //almacena el grado Celsius calculado por la formula |
| **NOMBRE DEL ALGORITMO:** convertir\_a\_celsius  **PROCESO DEL ALGORITMO**   1. Inicio 2. *Lee*r temperaturaFahrenheit 3. temperaturaCelsius ← (temperaturaFahrenheit -32) / 1.8 4. *Mostrar* “Esta temperatura Fahrenheit en temperatura Celsius es” + temperaturaCelsius |

***CAPTURA DEL PROCESSING:***



***RESULTADO:***



**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.

Gráfico

Descripción generada automáticamente

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.

**FASE DE ANÁLISIS**

**Descripción del problema:**

**Análisis:**

* **Datos de Entrada**: posicionLink, posicionTesoro
* **Datos de Salida:** la distancia entre Link y el tesoro, la representación de Link con un círculo (mover con el mouse), la representación del cofre con un cuadrado
* **Proceso:**

**¿Quién debe realizar el proceso?:** Pitágoras

**¿Cuál es el proceso que realiza?:** Para calcular la distancia necesitaremos de los catetos, que lo obtendremos según la posición de Link y el tesoro:

catetoA= posicionTesoro.y – posicionLink.y

catetoB= posicionTesoro.x – posicionLink.x

Después de calcular los catetos aplicaremos el teorema de Pitágoras que determinara la distancia entre ellos.

distancia= (catetoA^2 + catetoB^2) ^ (0.5)

Dibujaremos un círculo en: posicionLink.x, posicionLink.y

Dibujaremos un cuadrado en: posicionTesoro.x, posicionTesoro.y

Si la distancia es menor a un valor que elijamos, el tesoro desaparecerá.

**FASE DE DISEÑO**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Pitágoras |
| **VARIABLES**  posicionLink, posicionTesoro: coordenadas cartesianas  catetoA, catetoB, distancia: Real |
| **NOMBRE DEL ALGORITMO:** calcular\_distancia  **PROCESO DEL ALGORITMO**   1. Inicio 2. catetoA← posicionTesoro.y – posicionLink.y 3. catetoB← posicionTesoro.x – posicionLink.x 4. distancia← (catetoA^2 + catetoB^2) ^ (0.5) 5. **si** (distancia<20) **entonces** 6. //el tesoro desaparece 7. **sino** (distancia>20) **entonces** 8. dibujarTesoro() 9. **fin\_si** 10. fin |
| **NOMBRE DEL ALGORITMO:** dibujar\_link  **PROCESO DEL ALGORITMO**   1. Inicio 2. posicionLink.x ←mouseX 3. posicionLink.y ← mouseY 4. Dibujar un círculo en la posición: posicionLink.x, posicionLink.y 5. Fin |
| **NOMBRE DEL ALGORITMO:** dibujar tesoro  **PROCESO DEL ALGORITMO**   1. Inicio 2. anchoLienzo ← 500 3. altoLienzo←500 4. Dibujará un rectángulo, en el centro, en la posición: anchoLienzo/2, altoLienzo/2 5. Fin |

Captura del Processing:

Texto

Descripción generada automáticamente con confianza media

Texto

Descripción generada automáticamente

***RESULTADO:***

|  |  |
| --- | --- |
| **APARECE EL TESORO** | **DESAPARECE EL TESORO** |
| Icono  Descripción generada automáticamente | Imagen en blanco y negro  Descripción generada automáticamente con confianza media |

**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.

**FASE DE ANÁLISIS**

**Descripción del problema:** Obtener las raíces de una ecuación de segundo grado y analizar el discriminante

**Análisis:**

* Datos de Entrada:

a: Real

b: Real

c: Real

* Datos de Salida: las raíces de la ecuación

resultadoRaiz1: Real

resultadoRaiz2: Real

mensajeDiscriminante: String

* Proceso:

¿Quién debe realizar el proceso?: Calculadora

¿Cuál es el proceso que realiza?: Para calcular las raíces de una ecuación de segundo grado utiliza la siguiente fórmula:

x1= (-b + ((b^2-4\*a\*c) ^ (0.5))) /(2\*a)

x2= (-b - ((b^2-4\*a\*c) ^ (0.5))) /(2\*a)

Para analizar el determinante usaremos esta ecuación:

determinante= b^2-4\*a\*c

Para conocer que tipos de raíces tendremos se tomará en cuenta lo siguiente:

discriminante>0 //raíces reales diferentes

discriminante==0 //raíces reales iguales

discriminante<0 //raíces complejas diferentes

**FASE DE DISEÑO**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Calculadora |
| **VARIABLES**  a, b, c: Real //constantes de la ecuación de segundo grado  x1, x2: Real //calcula las raíces  resultadoRaiz1: Real //almacena la primera raíz  resultadoRaiz2: Real //almacena la segunda raíz  discriminante: Real //almacena el discriminante  opción: Entero //almacena una opción según el valor de la discriminante  mensajeDiscriminante: String //almacena el tipo de raíces que tiene la ecuación |
| **NOMBRE DEL ALGORITMO:** calcular\_raices  **PROCESO DEL ALGORITMO**   1. Inicio 2. *Leer* a 3. *Leer* b 4. *Leer* c 5. x1 ← (-b + ((b^2-4\*a\*c) ^ (0.5))) /(2\*a) 6. x2 ← (-b - ((b^2-4\*a\*c) ^ (0.5))) /(2\*a) 7. resultadoRaiz1 ← x1 8. resultadoRaiz2 ← x2 9. *Mostrar* “Primera raíz: “+ resultadoRaiz1 10. *Mostrar* “Segunda raíz: “+ resultadoRaiz2 11. Fin |
| **NOMBRE DEL ALGORITMO:** analizar\_discriminante   1. Inicio 2. *Leer* a 3. *Leer* b 4. *Leer* c 5. discriminante ← b^2-4\*a\*c 6. **si** (discriminante >0) **entonces** 7. opción ← 1 8. **si\_no** (discriminante==0) **entonces** 9. opción ← 2 10. **si\_no** (discriminante<0) **entonces** 11. opción ← 3 12. **fin\_si** 13. **según\_sea** (opción) **hacer** 14. **caso** 1: 15. mensajeDiscriminante ←"Son dos raíces reales diferentes" 16. **caso** 2: 17. mensajeDiscriminante ←"Son dos raíces reales iguales" 18. **caso** 3: 19. mensajeDiscriminante ←"Son dos raíces complejas diferentes" 20. **otros:** 21. mensajeDiscriminante ←"ERRROR" 22. **fin\_segun** 23. *Mostrar* mensajeDiscriminante 24. Fin |

***CAPTURA EN PROCESSING:***

|  |
| --- |
| Texto  Descripción generada automáticamente |
| Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico  Descripción generada automáticamente |

***RESULTADO:***

|  |  |  |
| --- | --- | --- |
| RAÍCES REALES DIFERENTES | RAÍCES REALES IGUALES | RAÍCES COMPLEJAS DIFERENTES |
| Texto  Descripción generada automáticamente | Texto  Descripción generada automáticamente | Texto  Descripción generada automáticamente |

**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:

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

**FASE DE ANÁLISIS**

**Descripción del problema:** Mover la línea, desde su inicio, en dirección hacia abajo arrastrando la ellipse, y al superar la posición de la altura del lienzo, debe dirigirse hacia arriba arrastrando la ellipse y así sucesivamente, es decir, que invierte su sentido.

**Análisis:**

* Datos de Entrada:

posicionLinea: coordenadas cartesianas

anchoEllipse, altoEllipse: entero

* Datos de Salida: El desplazamiento de la línea arrastrando una ellipse hacia arriba y hacia abajo sucesivamente
* Proceso:

¿Quién debe realizar el proceso?: El lienzo

¿Cuál es el proceso que realiza?: La posición de la línea es: 0, 0. Se dibujará primero la línea en: (posicionLinea.x, posicionLinea.y, anchoLienzo, posicionLinea.y) y también se dibujará la ellipse en la mitad de la línea: (ancho/2, posicionLinea.y-distancia, anchoEllipse, altoEllipse)

Para que se desplace por el eje y será necesario que la posicionLinea.y vaya aumentando de uno en uno. Esto se realizará con el incremento:

incremento=1

posicionLinea.y+=incremento

Cuando la posicionLinea.y sea mayor o igual al alto del lienzo, la línea tiene que volver al inicio, a 0. Entonces en vez de aumentar, tendremos que disminuir en 1 en 1 hasta que la posición sea =0

incremento\*=-1

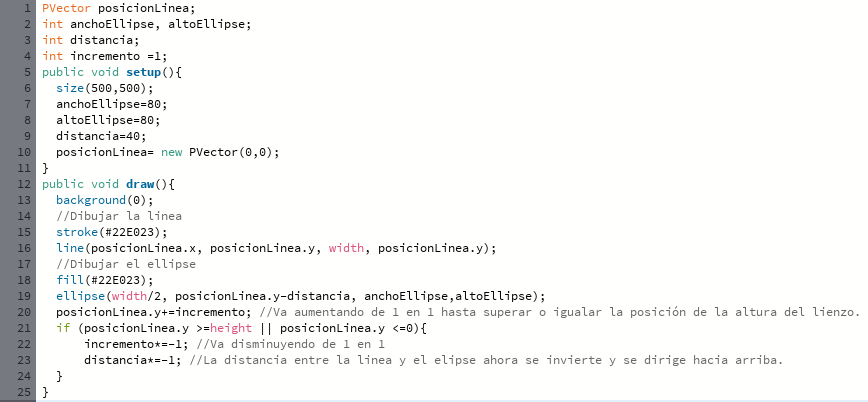
Y para que la elipse se invierta tendremos que multiplicar la distancia por -1

distancia\*=-1

**FASE DE DISEÑO**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA: Lienzo** |
| **VARIABLES**  posicionLinea: coordenadas cartesianas  anchoEllipse, altoEllipse, altoLienzo, anchoLienzo  incremento: entero |
| **NOMBRE DEL ALGORITMO:** dibujar\_linea  **PROCESO DEL ALGORITMO**   1. Inicio 2. anchoLienzo ←500 3. altoLienzo ←500 4. Dibujar una línea en la posicionLinea.x, posicionLinea.y, anchoLienzo, posicionLinea.y 5. dibujarEllipse() 6. incremento ← 1 7. posicionLinea.y ←posicionLinea.y + incremento 8. **sí**(posicionLinea.y< anchoLienzo) **entonces** 9. incremento ←incremento\*-1 10. distancia\*=-1 11. **fin\_si** 12. fin |
| **NOMBRE DEL ALGORITMO:** dibujar\_ellipse  **PROCESO DEL ALGORITMO**   1. anchoEllipse ←80 2. altoEllipse ←80 3. distancia ←40 4. Dibujar una ellipse en la mitad de la línea: x=anchoLienzo/2, y=posicionLinea.y-distancia, anchoEllipse, altoEllipse |

***CAPTURA DEL PROCESSING:***



***RESULTADO:***

|  |  |
| --- | --- |
| Interfaz de usuario gráfica  Descripción generada automáticamente con confianza media | Interfaz de usuario gráfica  Descripción generada automáticamente con confianza media |

**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:

Imagen que contiene Patrón de fondo

Descripción generada automáticamente

**FASE DE ANÁLISIS:**

**Definición del problema:** Dibujar en el lienzo rectángulos usando estructuras iterativas.

**Análisis**:

* Datos de Entrada:

coordenadasRect: coordenadas cartesianas

anchoLienzo, altoLienzo, ancho, alto, distanciaEntreRect: entero

* Datos de Salida:

Los rectángulos dibujados en el lienzo

* Proceso:

**¿Quién debe realizar el proceso?** El lienzo

**¿Cuál es el proceso que realiza?**

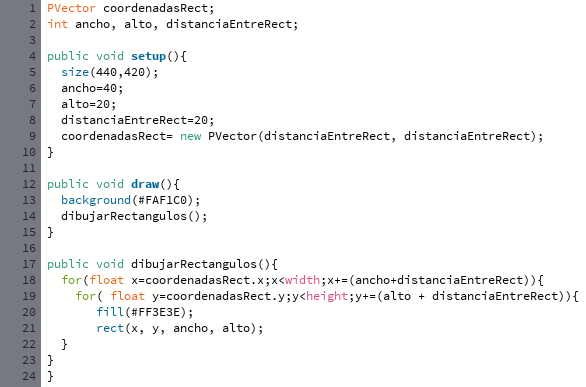
Dibujar los rectángulos en el lienzo. Para dibujarlos, como está en la imagen, se necesitará una estructura iterativa llamada for.

Estos rectángulos serán dibujados con un ancho de 40 y un alto de 20, manteniendo una distancia entre ellos que es 20.

**FASE DE DISEÑO:**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Lienzo |
| **VARIABLES**  coordenadasRect: coordenadas  ancho, alto, distanciaEntreRect: entero  anchoLienzo, altoLienzo: entero |
| **NOMBRE DEL ALGORITMO:** dibujar\_rectangulos  **PROCESO DEL ALGORITMO**   1. inicio 2. anchoLienzo ← 440 3. altoLienzo ← 420 4. ancho ← 40 5. alto ← 20 6. distanciaEntreRect ← 20 7. **para** x ← coordenadasRect.x **hasta** anchoLienzo **con paso** (ancho + distanciaEntreRect) **hacer** 8. **para** y ← coordenadasRect.y **hasta** altoLienzo **con paso** (alto + distanciaEntreRect) **hacer** 9. dibujar un rectángulo en (x, y, ancho, alto) 10. **fin\_para** 11. **fin\_para** 12. fin |

***CAPTURA DEL PROCESSING:***



***RESULTADO:***

Imagen que contiene Patrón de fondo

Descripción generada automáticamente

**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 .

Gráfico, Gráfico de líneas, Gráfico de cajas y bigotes

Descripción generada automáticamente

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.

**FASE DE ANÁLISIS**

**Descripción del problema:** Dibujar escalones sobre el lienzo y colocar sobre cada escalón un punto rojo.

**Análisis:**

* **Datos de Entrada:**

puntoA, puntoB, puntoC, puntoD: coordenadas cartesianas

distanciaLinea: Entero

* **Datos de Salida:**

El dibujo de la línea horizontal

El dibujo de la línea vertical

El dibujo del punto rojo

* **Proceso:**

**¿Quién debe realizar el proceso?:** El escalón

**¿Cuál es el proceso que realiza?:**

Dibujar una línea horizontal entre los puntos A y B, con distancia igual a distanciaLinea.

Dibujar una línea vertical entre los puntos B y C, con distancia igual a distanciaLinea.

Dibujar un punto en la siguiente posición: x= posición en x de B, y =

posición en y de B – 10 unidades.

Actualizar las coordenadas del puntoA con las de puntoC

Repetir desde el principio hasta que la coordenada en y de puntoA sea mayor que el alto del lienzo

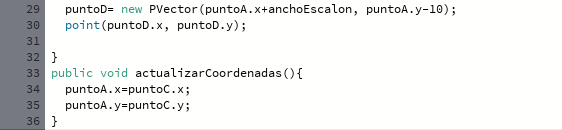
**FASE DE DISEÑO**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA:** Escalón |
| **VARIABLES**  puntoA, puntoB, puntoC, puntoD: coordenadas cartesianas  anchoEscalon, altoEscalon: Entero |
| **NOMBRE DEL ALGORITMO:** dibujar\_escalon  **PROCESO DEL ALGORITMO**   1. Inicio 2. Dibujar una línea horizontal entre los puntos A y B, con distancia igual a anchoEscalon. 3. Dibujar una línea vertical entre los puntos B y C, con distancia igual a altoEscalon 4. dibujar\_circulo () 5. Fin |
| **NOMBRE DEL ALGORITMO:** dibujar\_circulo  **PROCESO DEL ALGORITMO**   1. Inicio 2. Dibujar un punto en la siguiente posición: x= posición en x de B, y =posición en y de B – 10 unidades. 3. Fin |
| **NOMBRE DEL ALGORITMO:** actualizar\_coordenadas\_A  **PROCESO DEL ALGORITMO**   1. Inicio 2. puntoA.y <- puntoC.x 3. puntoA.y <- puntoC.y 4. Fin |

***CAPTURA DE PROCESSING:***

Texto

Descripción generada automáticamente



***RESULTADO:***

Gráfico, Gráfico de líneas, Gráfico de cajas y bigotes

Descripción generada automáticamente

**Ejercicio 22:** Utilizando la estructura de control repetitiva do-while. Replique la siguiente imagen:

Diagrama

Descripción generada automáticamente

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.

**FASE DE ANÁLISIS**

**Descripción del problema:** Dibujar círculos sobre cada línea de por medio

**Análisis:**

* Datos de Entrada: tamaño del lienzo
* Datos de Salida: Círculos dibujado sobre cada línea de por medio en el lienzo
* Proceso:

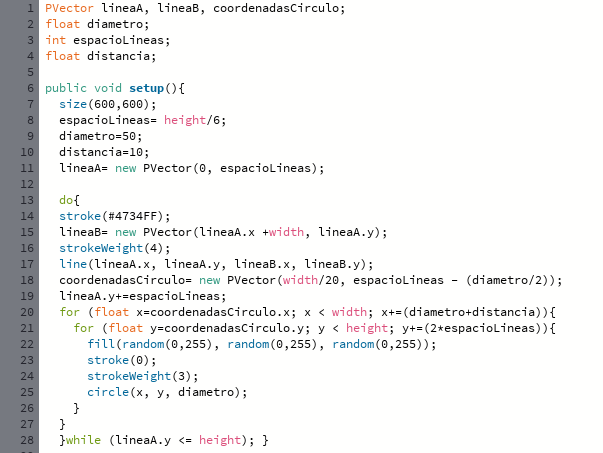
¿Quién debe realizar el proceso?: Lienzo

¿Cuál es el proceso que realiza?: Primero divide el altoLienzo por 6, y en esas medidas se dibujarán las líneas. Sobre esas líneas se dibujarán los círculos con un do while(), estos círculos tendrán colores aleatorios y la línea será de color azul, pero hay que tener en cuenta que no se dibujan en todas líneas. Si se dibujan en la línea 1 los círculos, entonces en la línea 2 no se dibujarán, y así sucesivamente

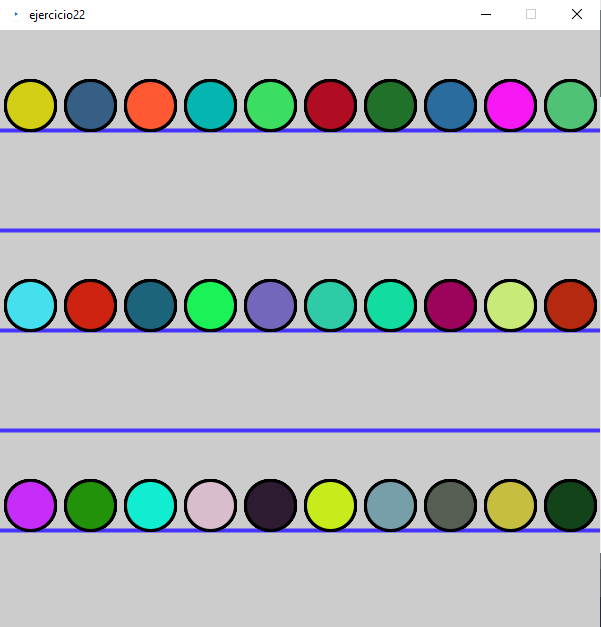
**FASE DE DISEÑO**

|  |
| --- |
| **ENTIDAD QUE RESUELVE EL PROBLEMA: Lienzo** |
| **VARIABLES**  anchoLienzo, altoLienzo, diámetro: Real  lineaA, lineaB, coordenadasCirculos: coordenadas cartesianas  espacioLineas: entero |
| **NOMBRE DEL ALGORITMO:** dibujar\_circulos  **PROCESO DEL ALGORITMO**   1. **Inicio** 2. anchoLienzo← 600 3. altoLienzo ← 600 4. espacioLineas ← altoLienzo /6 5. diámetro ← 50 6. distancia←10 7. **hacer** 8. Dibujar línea en (lineaA.x, lineaA.y, lineaB.x, lineaB,y); 9. lineaA.y ← lineaA.y + espacioLineas 10. **para x** ←coordenadasCirculo.x **hasta** anchoLienzo con paso (diámetro + distancia) 11. **hacer** 12. **para y** ← coordenadasCirculo.y **hasta** altoLienzo con paso (2\*espacioLineas) 13. **hacer** 14. dibujar un círculo en (x,y,diámetro) 15. **Fin\_para** 16. **Fin\_para** 17. **mientras** (lineaA.y < = altoLienzo) 18. **Fin** |

***CAPTURA DEL PROCESSING:***



***RESULTADO:***



# FUENTES BIBLIOGRÁFICAS

<https://youtu.be/Nr3NdAxjqsE?si=k3S_It3l23b4mBfz>

<https://youtu.be/Nr3NdAxjqsE?si=k3S_It3l23b4mBfz>

<https://youtu.be/m6oSw8IBfLQ?si=GWT8rejX1Tz0PPyB>

<https://youtu.be/EAclc8NcnOo?si=zBrv9HNE7_93pe0e>

<https://www.youtube.com/watch?v=l3xL2MA8hSs&t=345s>

Libro: Processing Un Lenguaje Al Alcance De Todos Edicion2013 por Ignacio Buioli y Jaime Perez Marin

https://processing.org/reference/

https://www.youtube.com/watch?v=dysxlptjAx0&t=5s

Apuntes del aula virtual

07 fases en la resolución de problemas mediante algoritmos,

11 bifurcaciones- Expresiones Lógicas- Estructuras Selectivas

14 estructuras de Control Iterativas,