|  |  |  |
| --- | --- | --- |
| 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*

*Año 2024*

Trabajo Práctico

N°1

Mamani,Fernando – 135

**FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS**

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

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

Resolución necesaria en Word:

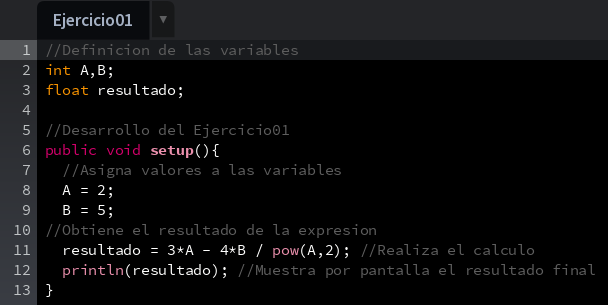
(3\*A) - (4\*B/(A^2))

6 - (4\*B/4)

6 - 5

1

**Captura de Processing:**



**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: Obtener el resultado de la expresión.

**Datos de entrada:**

A,B: entero

**Datos de Salida:**

resultado: float

**Proceso:**

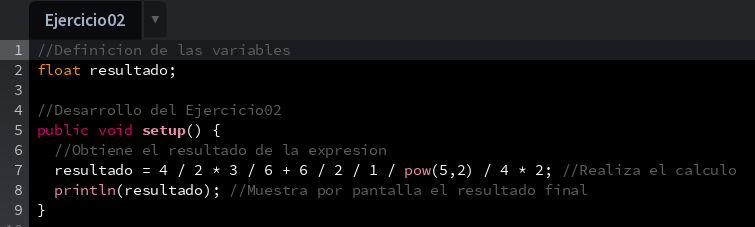
Realizar los calculos

**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  A, B: entero  resultado: float |
| **Nombre Algoritmo:** Ejercicio01  **Algoritmo:**  **Inicio**  A <- 2;  B <- 5;  resultado <- 3\*A - 4\*B / A^2;  escribir resultado  **Fin** |
|  |

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

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: Obtener el resultado de la expresión.

**Datos de entrada:**

Los números usar para calcular: enteros

**Datos de Salida:**

resultado: float

**Proceso:**

Realizar los calculos

**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  resultado: float |
| **Nombre Algoritmo:** Ejercicio02  **Algoritmo:**  **Inicio**  resultado <- 4 / 2 \* 3 / 6 + 6 / 2 / 1 / 5 ^ 2 / 4 \* 2  escribir resultado  **Fin** |
|  |

**Punto 4**: Evaluar las siguientes expresiones aritméticas, para lo cual indicar en el caso de las variables, el valor indicado. Luego escribirlas como expresiones algebraicas.

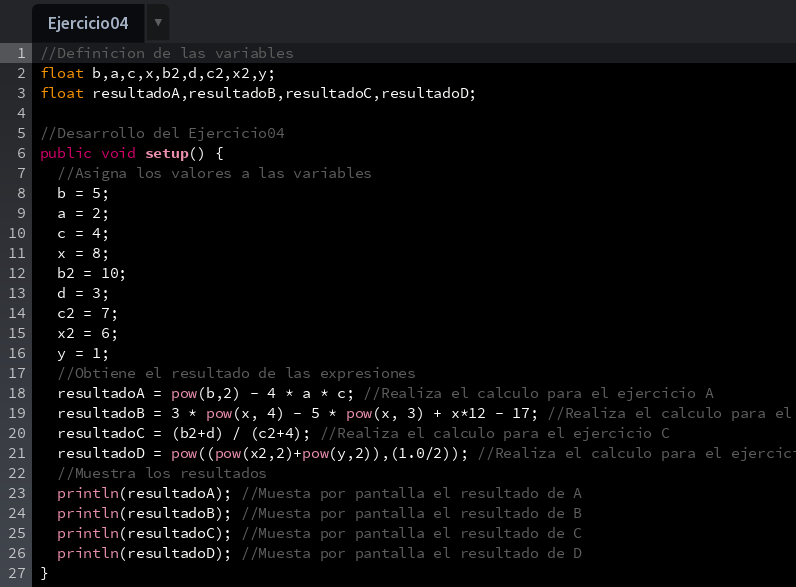
a) b ^ 2 – 4 \* a \* c

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

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

d) (x ^ 2 + y ^ 2) ^ (1 / 2)

**Captura de Processing:**

****

**Desarrollo del punto**

1. b2-4ac
2. 3X4 -5X3+X12-17

**Analisis:**

Descripcion del Problema: Asignar valor a las variables y calcular

**Datos de entrada:**

b,a,c,x,b2,d,c2,x2,y: float

**Datos de Salida:**

resultadoA,resultadoB,resultadoC,resultadoD: float

**Proceso:**

Realizar los calculos

**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  b,a,c,x,b2,d,c2,x2,y: float  resultadoA,resultadoB,resultadoC,resultadoD: float |
| **Nombre Algoritmo:** Ejercicio04  **Algoritmo:**  **Inicio**  b <- 5;  a <- 2;  c <- 4;  x <- 8;  b2 <- 10;  d <- 3;  c2 <- 7;  x2 <- 6;  y <- 1;  resultadoA <- b ^ 2 – 4 \* a \* c;  resultadoB <- 3 \* X ^ 4 – 5 \* X ^ 3 + X 12 – 17;  resultadoC <- (b2 + d) / (c2 + 4);  resultadoD <- (x ^ 2 + y ^ 2) ^ (1 / 2);  escribir resultadoA;  escribir resultadoB;  escribir resultadoC;  escribir resultadoD;  **Fin** |
|  |

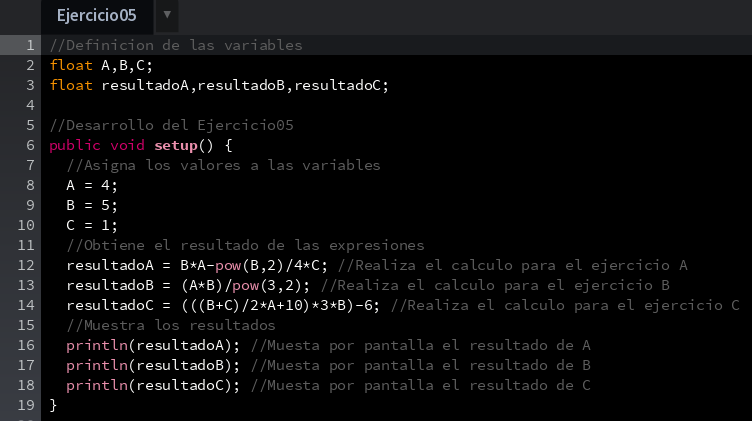
**Punto 5**: Si el valor de A es 4, el valor de B es 5 y el valor de C es 1, evaluar las siguientes expresiones:

a) B \* A – B ^ 2 / 4 \* C

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

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

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: Calcular las ecuaciones

**Datos de entrada:**

A,B,C: float

**Datos de Salida:**

resultadoA,resultadoB,resultadoC,: float

**Proceso:**

Realizar los calculos

**Diseño:**

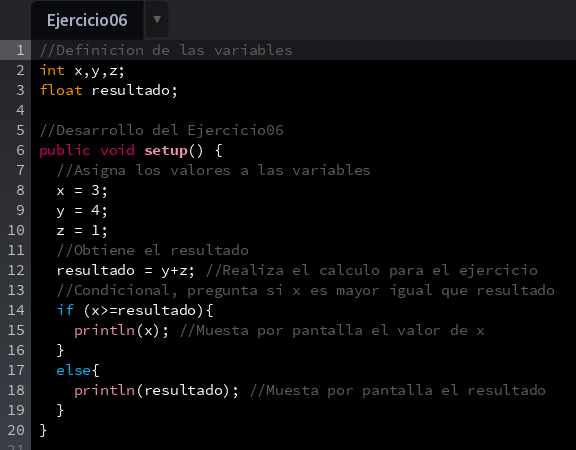
|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  A,B,C: float  resultadoA,resultadoB,resultadoC,: float |
| **Nombre Algoritmo:** Ejercicio05  **Algoritmo:**  **Inicio**  A <- 4;  B <- 5;  C <- 1;  resultadoA <- B \* A – B ^ 2 / 4 \* C;  resultadoB <- (A \* B) / 3 ^ 2;  resultadoC <- (((B + C) / 2 \* A + 10) \* 3 \* B) – 6;  escribir resultadoA;  escribir resultadoB;  escribir resultadoC;  **Fin** |
|  |

**Punto 6**: Para x=3, y=4; z=1, evaluar elresultado de:

R1 = y+z

R2 = x >= R1

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: Calcular la suma y consultar

**Datos de entrada:**

x,y,z:entero

**Datos de Salida:**

resultado: float

**Proceso:**

Realizar suma y consultar

**Diseño:**

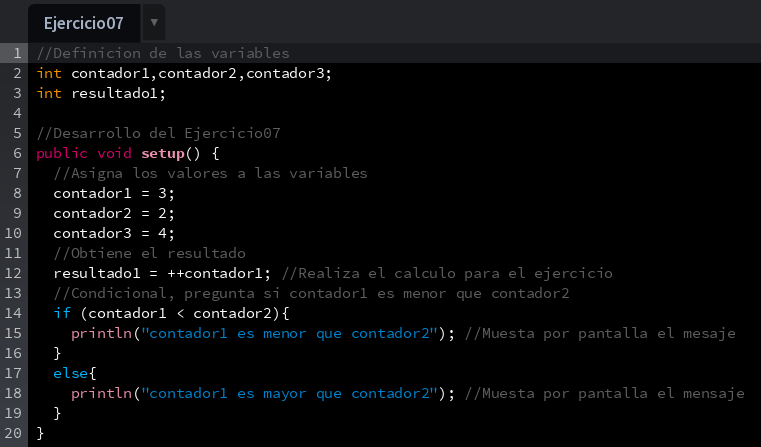
|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  x,y,z: entero  resultado: float |
| **Nombre Algoritmo:** Ejercicio06  **Algoritmo:**  **Inicio**  x <- 3;  y <- 4;  z <- 1;  resultado <- y+z;  **Si** (x > = resultado) Entonces  Escribir x  **SiNo**  Escribir resultado  **FinSi**  **Fin** |
|  |

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

R1 = ++contador1

R2 = contador1 < contador2

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: Calcular la suma y consultar

**Datos de entrada:**

contador1, contador2, contador3: entero

**Datos de Salida:**

resultado1: entero

**Proceso:**

Realizar suma y consultar

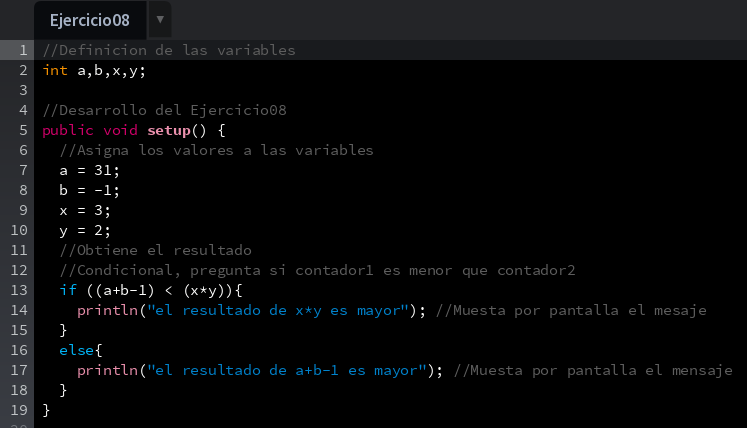
**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  contador1, contador2, contador3: entero  resultado1: entero |
| **Nombre Algoritmo:** Ejercicio07  **Algoritmo:**  **Inicio**  contador1 <- 3;  contador2 <- 2;  contador3 <- 4;  resultado1 <- ++contador1;  **Si** (contador1 < contador2) Entonces  Escribir “contador1 es menor que contador2”;  **SiNo**  Escribir “contador1 es mayor que contador2”;  **FinSi**  **Fin** |
|  |

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

a+b-1 < x\*y

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: Calcular y consultar

**Datos de entrada:**

a,b,x,y: entero

**Datos de Salida:**

mensaje: caracter

**Proceso:**

Realizar calculos y consultar

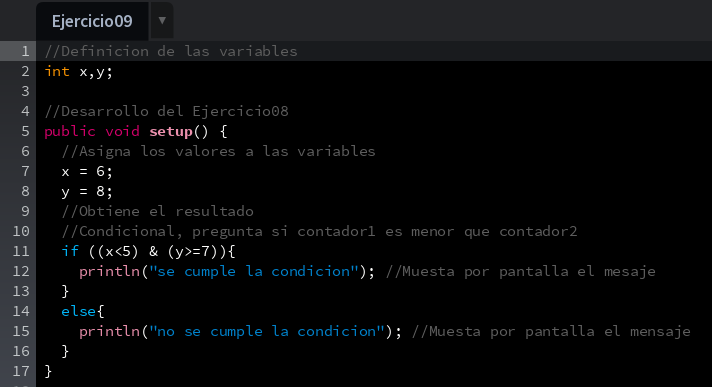
**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  a,b,x,y: entero |
| **Nombre Algoritmo:** Ejercicio08  **Algoritmo:**  **Inicio**  a <- 31;  b <- -1;  x <- 3;  y <- 2;  **Si** ((a+b-1) < (x\*y)) Entonces  Escribir "el resultado de x\*y es mayor";  **SiNo**  Escribir "el resultado de a+b-1 es mayor";  **FinSi**  **Fin** |
|  |

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

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

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: consultar

**Datos de entrada:**

x,y: entero

**Datos de Salida:**

mensaje: caracter

**Proceso:**

Realizar consultar

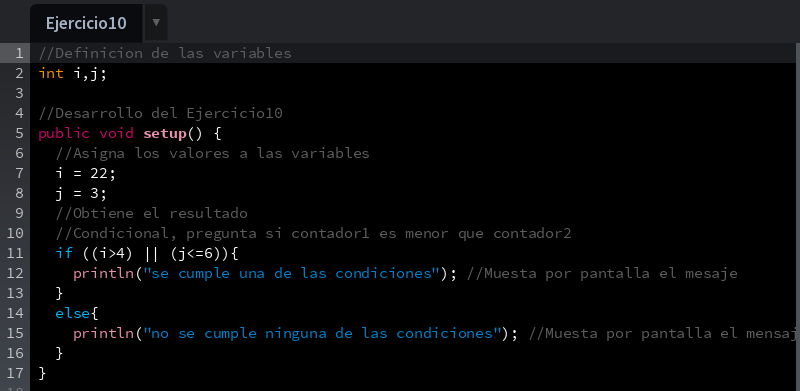
**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  x,y: entero |
| **Nombre Algoritmo:** Ejercicio09  **Algoritmo:**  **Inicio**  x <- 6;  y <- 8;  **Si** ((x<5) && (y>=7)) Entonces  Escribir "se cumple la condicion";  **SiNo**  Escribir "no se cumple la condicion";  **FinSi**  **Fin** |
|  |

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

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

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: consultar

**Datos de entrada:**

i,j: entero

**Datos de Salida:**

mensaje: caracter

**Proceso:**

Realizar consultar

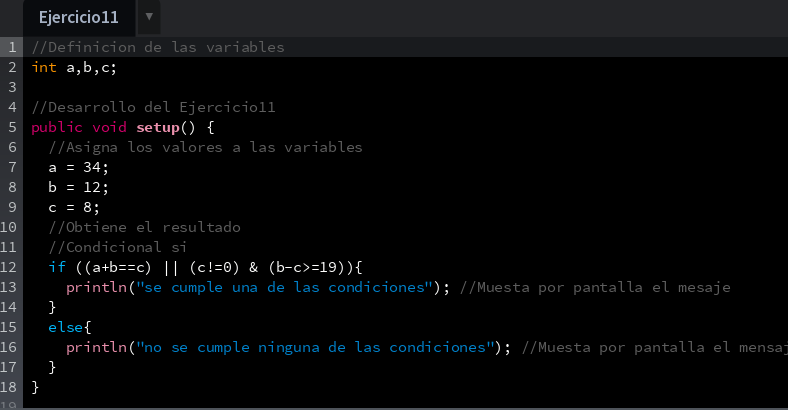
**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  i,j: entero |
| **Nombre Algoritmo:** Ejercicio10  **Algoritmo:**  **Inicio**  i <- 22;  j <- 3;  **Si** ((i>4) || (j<=6)) Entonces  Escribir "se cumple una de las condiciones";  **SiNo**  Escribir "no se cumple ninguna de las condiciones";  **FinSi**  **Fin** |
|  |

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

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

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: consultar

**Datos de entrada:**

a,b,c: entero

**Datos de Salida:**

mensaje: caracter

**Proceso:**

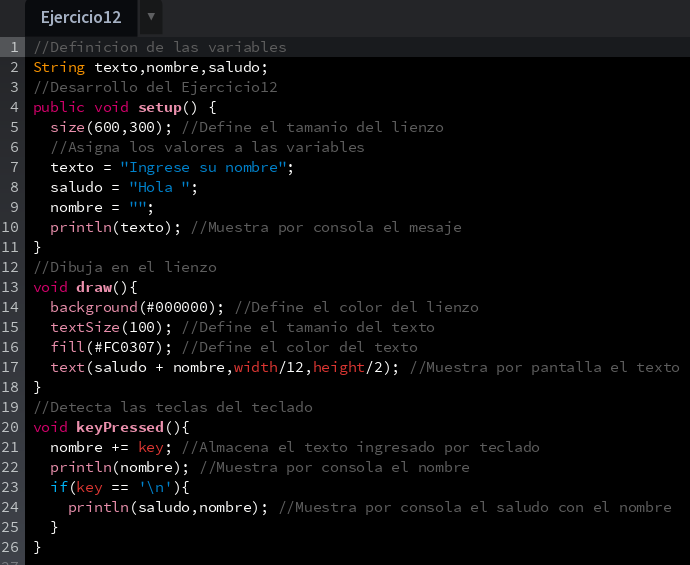
Realizar consultar

**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  a,b,c: entero |
| **Nombre Algoritmo:** Ejercicio11  **Algoritmo:**  **Inicio**  a <- 34;  b <- 12;  c <- 8;  **Si** ((a+b==c) || (c!=0) && (b-c>=19)) Entonces  Escribir "se cumple una de las condiciones";  **SiNo**  Escribir "no se cumple ninguna de las condiciones";  **FinSi**  **Fin** |
|  |

**Punto 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.

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: realizar un saludo con nombre de usuario

**Datos de entrada:**

nombre: carácter

texto:caracter

**Datos de Salida:**

saludo: caracter

**Proceso:**

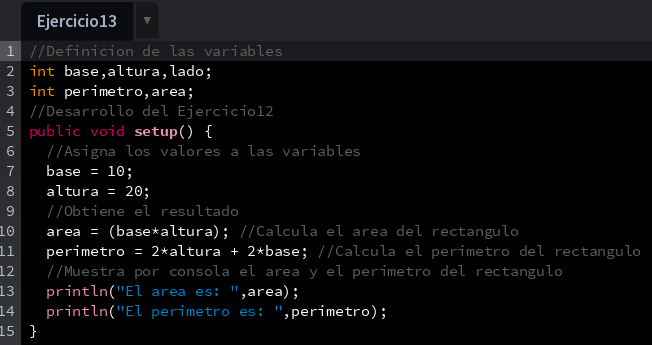
Pedir por teclado nombre y realizar saludo

**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  nombre,texto,saludo: carácter; |
| **Nombre Algoritmo:** Ejercicio12  **Algoritmo:**  **Inicio**  texto <- “Ingrese su nombre”;  escribir texto;  leer nombre;  escribir saludo,nombre;  **Fin** |
|  |

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

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: realizar calculo de área y perímetro de un rectangulo

**Datos de entrada:**

base,altura : entero

**Datos de Salida:**

perímetro,área : entero

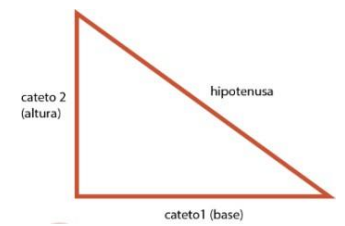
**Proceso:**

Calcular área y perímetro del rectangulo

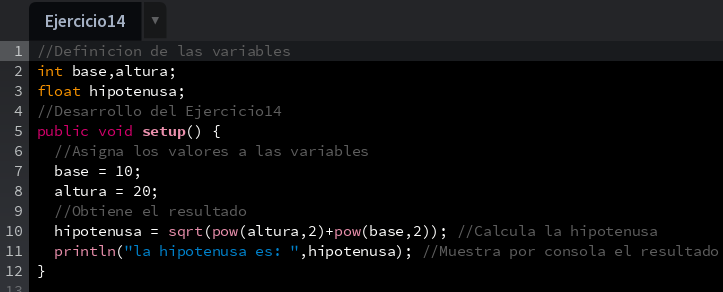
**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  base,altura: entero;  perímetro,área: entero |
| **Nombre Algoritmo:** Ejercicio13  **Algoritmo:**  **Inicio**  base <- 10;  altura <-20;  área <- (base\*altura);  perímetro <- 2\*base + 2\*altura;  escribir “El área es: “,área;  escribir “El perímetro es: “,perímetro;  **Fin** |
|  |

**Punto 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.



**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: realizar calculo de la hipotenusa de un triangulo rectangulo

**Datos de entrada:**

base,altura : entero

**Datos de Salida:**

hipotenusa : float

**Proceso:**

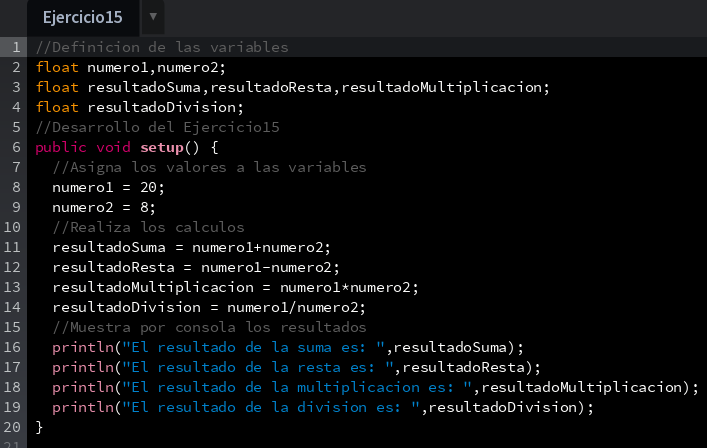
Calcular la hipotenusa del triangulo rectangulo

**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  base,altura: entero;  hipotenusa: float |
| **Nombre Algoritmo:** Ejercicio14  **Algoritmo:**  **Inicio**  base <- 10;  altura <-20;  hipotenusa <- ;  escribir “la hipotenusa es: “,hipotenusa;  **Fin** |
|  |

**Punto 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.

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: realizar los calculos

**Datos de entrada:**

numero1,numero2 : entero

**Datos de Salida:**

resultadoSuma,resultadoResta,resultadoMultiplicacion: entero

resultadoDivision:float

**Proceso:**

Calcular los problemas

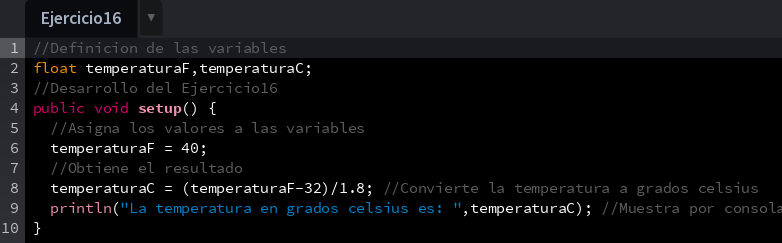
**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  numero1,numero2: entero;  resultadoSuma,resultadoResta,resultadoMultiplicacion:entero  resultadoDivision: float |
| **Nombre Algoritmo:** Ejercicio15  **Algoritmo:**  **Inicio**  numero1 <- 20;  numero2 <-8;  resultadoSuma <- numero1 + numero2;  resultadoResta <- numero1 – numero2;  resultadoMultiplicacion <- numero1 \* numero2;  resultadoDivision <- numero1/numero2;  escribir “El resultado de la suma es: “,resultadoSuma;  escribir “El resultado de la resta es: “,resultadoResta;  escribir “El resultado de la multiplicacion es: “,resultadoMultiplicacion;  escribir “El resultado de la division es: “,resultadoDivision;  **Fin** |
|  |

**Punto 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



**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: convertir temperatura

**Datos de entrada:**

temperaturaF: float

**Datos de Salida:**

temperaturaC:float

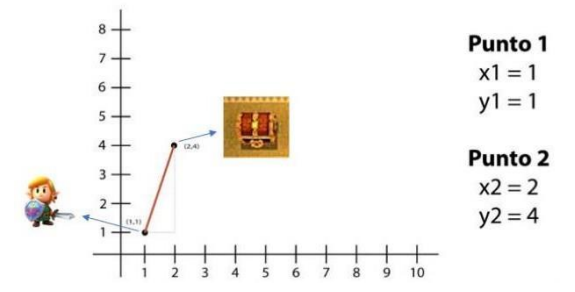
**Proceso:**

Convertir la temperatura

**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  temperaturaF,temperaturaC: float; |
| **Nombre Algoritmo:** Ejercicio16  **Algoritmo:**  **Inicio**  temperaturaF <- 40;  temperaturaC <- (temperaturaF-32)/1.8;  escribir “La temperatura en grados Celsius es: “,temperaturaC;  **Fin** |
|  |

**Punto 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.

**Captura de Processing:**

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema:

**Datos de entrada:**

**Datos de Salida:**

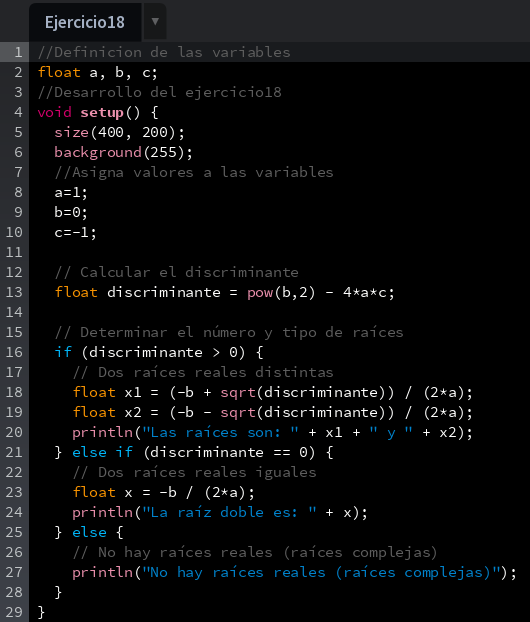
**Proceso:**

**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:** |
| **Nombre Algoritmo:** Ejercicio17  **Algoritmo:**  **Inicio**  **Fin** |
|  |

**Punto 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.

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: convertir temperatura

**Datos de entrada:**

a, b ,c:float

**Datos de Salida:**

raices

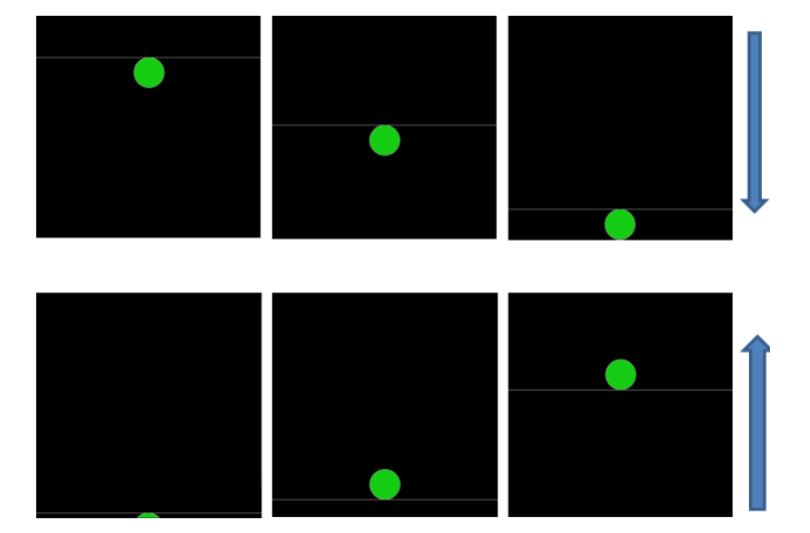
**Proceso:**

Realizar y calcular las raices

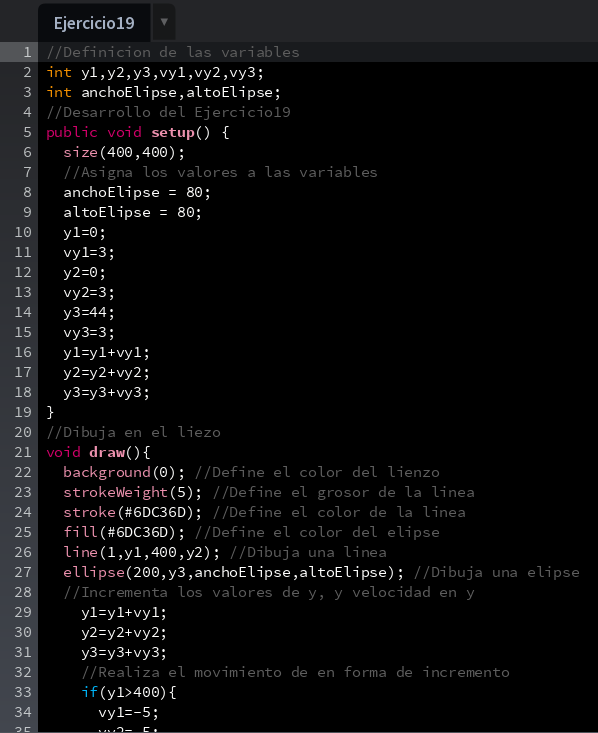
**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  a, b ,c:float  discriminante:float |
| **Nombre Algoritmo:** Ejercicio18  **Algoritmo:**  **Inicio**  *inicio*  Leer a  Leer b  Leer c  discriminant*e* ← b^2 – 4\*a\*c  ***si*** (discriminante > 0) **entonces**  raiz1 ← (-b + (discriminante))^0.5 /(2\*a)  raiz2 ← (-b - (discriminante))^0.5 /(2\*a)  mostrar “las raíces son: ” + raiz1 + “ y ” + raiz2  **si\_no si** (discriminante == 0) **entonces**  raiz *← -b / (2\*a)*  mostrar “la raíz doble es: “ + raiz  **si\_no**  mostrar “no hay raíces reales”  FinSi  **Fin** |
|  |

**Punto 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



**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: realizar el ejercicio utilizando el incremento y decremento de los valores x,y de la pelota y línea.

**Datos de entrada:**

y1,y2,y3,vy1,vy2,vy3:entero;

anchoElipse,altoElipse:entero;

**Datos de Salida:**

Dibujo en el lienzo

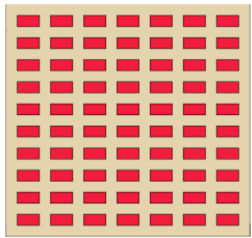
**Proceso:**

Sibujar y cambiar valores en x,y

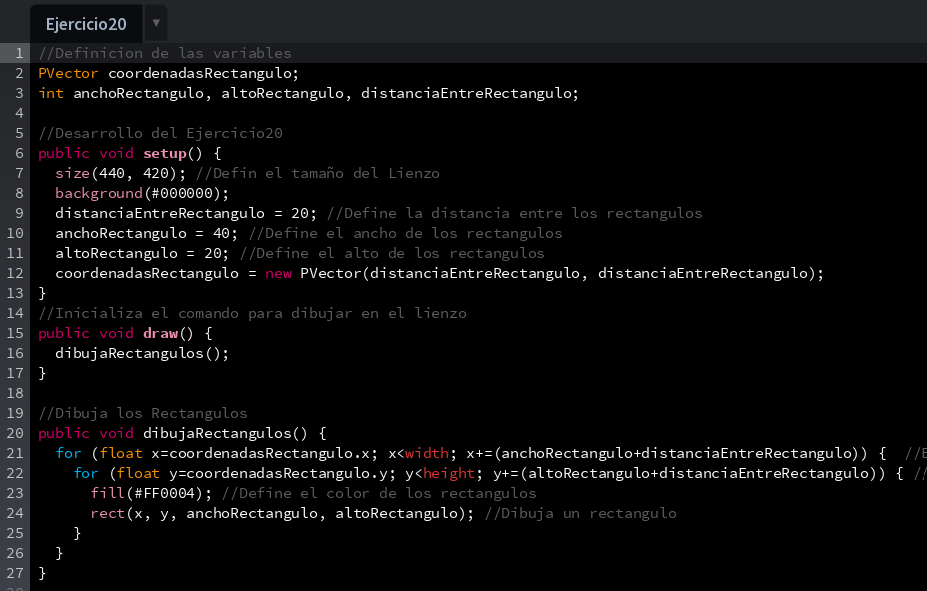
**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  y1,y2,y3,vy1,vy2,vy3:entero;  anchoElipse,altoElipse:entero; |
| **Nombre Algoritmo:** Ejercicio19  **Algoritmo:**  **Inicio**  y1<-0;  vy1<-3;  y2<-0;  vy2<-3;  y3<-44;  vy3<-3;  y1<-y1+vy1;  y2<-y2+vy2;  y3<-y3+vy3;  **SI** (y1>400) entonces  vy1<- -5;  vy2<- -5;  vy3<- -5;  y3<-y3-80;  **FinSI**  **Si** (y1<0)entonces  vy1<-5;  vy2<-5;  vy3<-5;  y3<-y3+80;  **Fin** |
|  |

**Punto 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:



**Captura de Processing:**



**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: Dibuja en el lienzo rectángulos utilizando estructuras iterativas

**Datos de entrada:**

coordenadasRectangulo: coordenadas cartesianas

ancho,alto,distanciaEntreRectangulo: entero

**Datos de Salida:**

Los rectángulos dibujados.

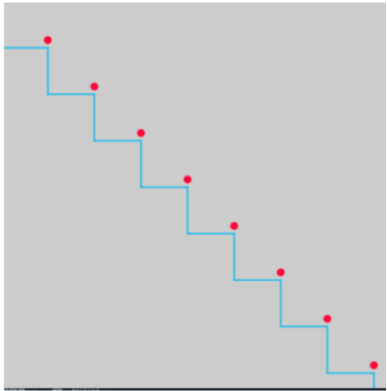
**Proceso:**

Dibujar los rectángulos.

**Diseño:**

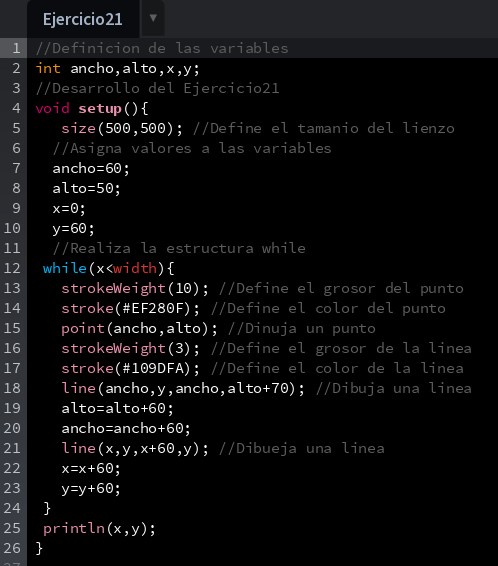
|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  coordenadasRectangulo: Coordenadas  ancho,alto,distanciaEntreRectangulo: entero  anchoLienzo,altoLienzo: entero |
| **Nombre Algoritmo:** Ejercicio20  **Algoritmo:**  Inicio  anchoLienzo <- 440;  altoLienzo <- 420;  distanciaEntreRectangulos <- 20;  anchoRectangulo <- 40;  altoRectangulo <- 20;  **Para** x<-coordenadasRectangulo.x hasta anchoLienzo con paso (ancho+distanciaEntreRectangulo) hacer  **Para** y<-coordenadasRectangulo.y hasta altoLienzo con paso (alto+distanciaEntreRectangulo) hacer  Dibujar un rectángulo en (x,coordenadasRectangulo.y)condimensiones ancho y alto  **Fin para**  **Fin para**  **Fin** |
|  |

**Punto 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.

**Captura de Processing:**

****

**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: dibujar imagen usando while

**Datos de entrada:**

ancho,alto,x,y:entero;

**Datos de Salida:**

Dibujo en el lienzo

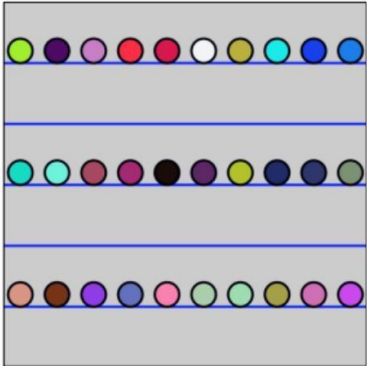
**Proceso:**

Realizar el dibuja en el lienzo usando while

**Diseño:**

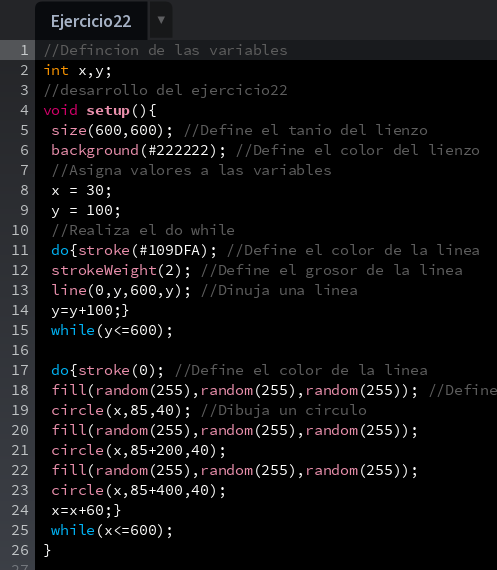
|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  ancho,alto,x,y:entero |
| **Nombre Algoritmo:** Ejercicio21  **Algoritmo:**  **Inicio**  ancho <- 60;  alto <-50;  x <-0;  y<-60;  **Mientras** (x<ancholienzo)hacer  dibujapunto <- ancho,alto);  dibujalinea <- ancho,y,ancho,alto+70;  alto <- alto+60  ancho <- ancho+60;  dibujalinea <- x,y,x+60,y;  x<-x+60;  y+y+60;  **FinMientras**  **Fin** |
|  |

**Punto 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.

**Captura de Processing:**



**Desarrollo del punto**

**Analisis:**

Descripcion del Problema: Dibuja en el lienzo círculos y líneas usando estructuras iterativas

**Datos de entrada:**

x,y:entero

**Datos de Salida:**

Los círculos y lineas

**Proceso:**

Dibujar los círculos y lineas

**Diseño:**

|  |
| --- |
| **Entidad:**  Lienzo |
| **Variables:**  x,y:entero |
| **Nombre Algoritmo:** Ejercicio22  **Algoritmo:**  **Inicio**  x<-30;  y<-100;  repetir  dibujalinea<-0,y600,y;  y<-y+100  hasta que(y<=600);  repetir  dibujacirculo<- x,85,40;  dibujacirculo<- x,85+200,40;  dibujacirculo<- x,85+400,40;  x=x+60;  hasta que(x<=600);  **Fin** |
|  |