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

Trabajo Práctico

N° 1

Pérez, Ezequiel Alejandro

LU: TUV000625

**FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS**

Indice

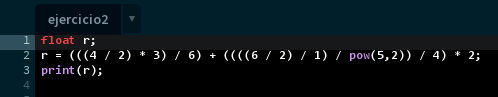
**Sección Expresiones aritméticas y lógicas**

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

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

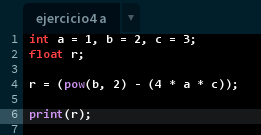
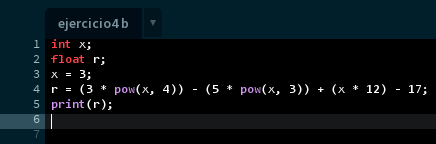
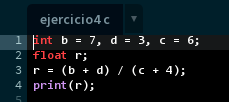
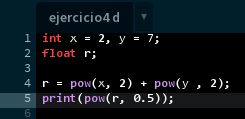
1 + 0.06 = 1.06

Código:



~~Ejercicio 3: Escribir las siguientes expresiones algebraicas como expresiones algorítmicas (en su forma aritmética dentro del algoritmo). En este caso no se pide evaluarlas ni programarlas.~~

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

1. b ^ 2 – 4 \* a \* c  
     
   a = 1, b =2 , c = 3  
     
   **Aritmética:**b ^ 2 – 4 \* a \* c  
   (2 ^ 2) – (4 \* 1 \* 3)  
   4 – 12  
   -8 **Algebraica:**  
     
    b2 – 4ac  
   22 – 4 . 1 . 3  
   4 – 12  
   -8  
     
   **Código:**
2. 3 \* X ^ 4 – 5 \* X ^ 3 + X 12 – 17   
     
   X = 3  
     
   **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 **Algebraica:**3x4 – 5x3 + 12x – 17  
   3 . 34 – 5 . 33 + 12 . 3 – 17  
   3 . 81 – 5 . 27 – 12 . 3 – 17  
   243 – 135 + 36 – 17  
   127  
     
   **Código:  
     
   **
3. (b + d) / (c + 4)   
     
   b = 7, d = 3, c = 6  
     
   **Aritmética:**(b + d) / (c + 4)  
   (7 + 3) / (6 + 4)  
   10 / 10  
   1 **Algebraica:  
     
   Código:**
4. (x ^ 2 + y ^ 2) ^ (1 / 2)  
     
   x =2 , y = 7  
     
   **Aritmética:**(x ^ 2 + y ^ 2) ^ (1 / 2)  
   (2 ^ 2 + 7 ^ 2) ^ (1 / 2)  
   (4 + 49) ^ (1 / 2)  
   53 ^ (1 / 2)  
   7.2801098893  
     
    **Algebraica:**  
   **Código:**

**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

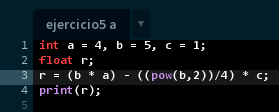
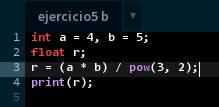
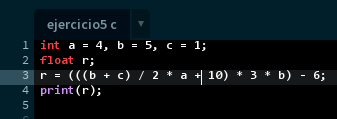
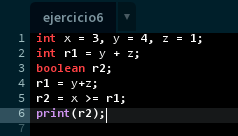
b) (A \* B) / 3 ^ 2

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

**Resolución:**  
  
a) (5 \* 4) – ((5 ^ 2) / 4) \* 1

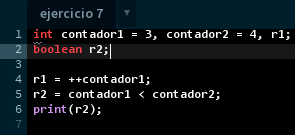
20 – (25 / 4) \* 1

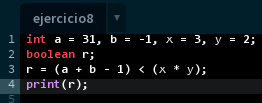
20 – 6.25

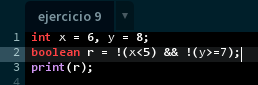
13.75  
  
  
**Código:**b) (A \* B) / 3 ^ 2  
  
(4 \* 5) / 3 ^ 2  
  
(4 \* 5) / 9  
  
20 / 9  
  
2.22  
  
**Código:**c) (((B + C) / 2 \* A + 10) \* 3 \* B) – 6  
  
(((5 + 1) / 2 \* 4 + 10) \* 3 \* 5) – 6  
((6 / 2 \* 4 + 10) \* 3 \* 5) – 6  
(3 \* 4 + 10) \* 3 \* 5) – 6  
(12 + 10) \* 3 \* 5) – 6  
(22 \* 3 \* 5) – 6  
330 – 6  
324  
  
**Código:**  
  
Ejercicio 6: Para x=3, y=4; z=1, evaluar el resultado de:   
  
R1 = y+z   
R2 = x >= R1  
  
**R1** = 4 + 1  
**R2** = x >= **R1** 🡪 Falso  
**Código:**  
  


Ejercicio 7: Para contador1=3, contador3=4, evaluar el resultado de   
  
R1 = ++contador1   
R2 = contador1 < contador2

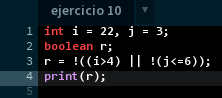
**R1** = ++contador1  
**R2** = contador1<contador2 🡪 Falso

**Código:**  
  
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 – 1 < 6  
  
29 < 6

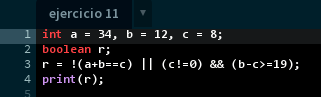
Falso  
  
**Código:**

Ejercicio 9: Para x=6, y=8, evaluar el resultado de:  
  
!(x<5)CC !(y>=7)  
  
!(6<5) && !(8>=7)  
  
!(False) && !(True)  
  
True && False  
  
False  
  
**Código:  
  
**

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  
  
**Código:**

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

!(a+b==c) || (c!=0)CC(b-c>=19)  
!(46==8) || (8!=0)CC(12 - 8>=19)  
!(False) || (True) && (4>=19)  
True || (True) && (False)  
True || False  
True  
  
**Código:**

**Sección Análisis – Diseño y Codificación de algoritmos – Aplicación de estructuras de control**

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:**

* Especificación del problema: Pedir al usuario su nombre y mostrarlo por pantalla
* Análisis:

Datos de entrada:

nombreUsuario: **String**

Datos de salida:

darMensaje: **String**

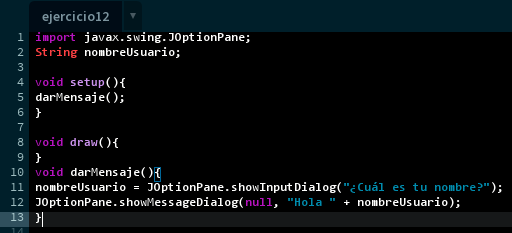
Proceso:

// se le pedirá al usuario que escriba su nombre

// con el nombre del usuario, se creará el mensaje saludándolo  
darMensaje = “Hola ” + nombreUsuario

**Fase de diseño:**

|  |
| --- |
| **Entidad que resuelve el problema:** El programador |
| **Variables:**  nombreUsuario: **String** // va a almacenar lo que el usuario ingrese |
| **Nombre del algoritmo:** darMensaje  **Proceso:**   1. **Inicio** 2. *Mostrar* “¿Cuál es tu nombre?” 3. *Leer nombreUsuario* 4. darMensaje 🡨 “Hola ” + nombreUsuario 5. *Mostrar* darMensaje 6. **fin** |

**Código:**

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

**Fase de Análisis:**

* Especificación del problema: problema
* Análisis:

Datos de entrada:

Dato

Datos de salida:

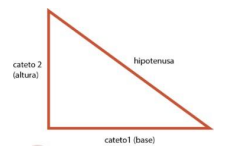
Salida:

Proceso:

**Fase de diseño:**

|  |
| --- |
|  |
|  |
|  |

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



**Fase de Análisis:**

* Especificación del problema: problema
* Análisis:

Datos de entrada:

Dato

Datos de salida:

Salida:

Proceso:

**Fase de diseño:**

|  |
| --- |
|  |
|  |
|  |

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:**

* Especificación del problema: problema
* Análisis:

Datos de entrada:

Dato

Datos de salida:

Salida:

Proceso:

**Fase de diseño:**

|  |
| --- |
|  |
|  |
|  |

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:**

* Especificación del problema: problema
* Análisis:

Datos de entrada:

Dato

Datos de salida:

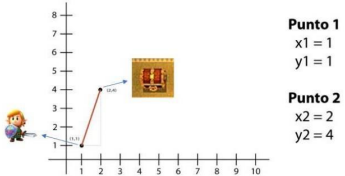
Salida:

Proceso:

**Fase de diseño:**

|  |
| --- |
|  |
|  |
|  |

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.

**Fase de Análisis:**

* Especificación del problema: problema
* Análisis:

Datos de entrada:

Dato

Datos de salida:

Salida:

Proceso:

**Fase de diseño:**

|  |
| --- |
|  |
|  |
|  |

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:**

* Especificación del problema: problema
* Análisis:

Datos de entrada:

Dato

Datos de salida:

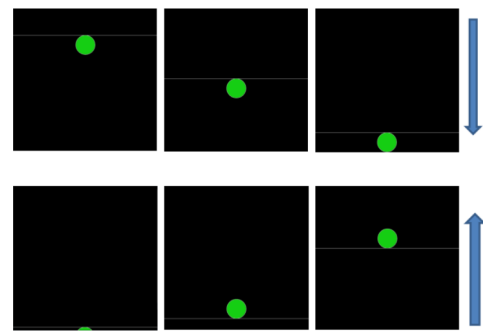
Salida:

Proceso:

**Fase de diseño:**

|  |
| --- |
|  |
|  |
|  |

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

****

**Fase de Análisis:**

* Especificación del problema: problema
* Análisis:

Datos de entrada:

Dato

Datos de salida:

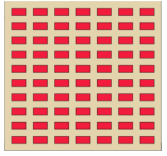
Salida:

Proceso:

**Fase de diseño:**

|  |
| --- |
|  |
|  |
|  |

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:

****

**Fase de Análisis:**

* Especificación del problema: problema
* Análisis:

Datos de entrada:

Dato

Datos de salida:

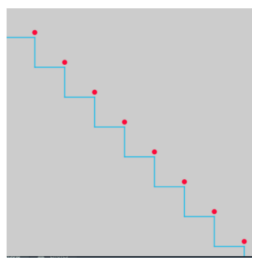
Salida:

Proceso:

**Fase de diseño:**

|  |
| --- |
|  |
|  |
|  |

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.

**Fase de Análisis:**

* Especificación del problema: problema
* Análisis:

Datos de entrada:

Dato

Datos de salida:

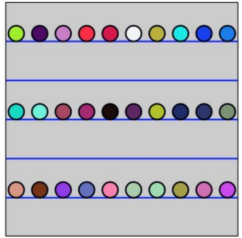
Salida:

Proceso:

**Fase de diseño:**

|  |
| --- |
|  |
|  |
|  |

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.

**Fase de Análisis:**

* Especificación del problema: problema
* Análisis:

Datos de entrada:

Dato

Datos de salida:

Salida:

Proceso:

**Fase de diseño:**

|  |
| --- |
|  |
|  |
|  |

Conclusión

Párrafos de las conclusiones

Fuentes bibliográficas

Se deben enunciar las fuentes (apuntes de la materia, páginas web, videos de youtube, libro (nombre, autores, año), etc)