

**Laboratorio de Computación Salas A y B**

Profesor(a): Asignatura:

Fundamentos de programación

Karina García Morales

Grupo:

132

No de practica(s):

practica 3

Integrante(s):

Reyes Landa Gerardo Isaac

No de lista o brigada:

42

primer semestre

Fecha de entrega:

5 de septiembre del 2024

Observaciones:

**Calificación:**

**TRABAJO EN CLASE:**

**OBJETIVOS:**

El alumno elaborará algoritmos correctos y eficientes en la solución de problemas

siguiendo las etapas de Análisis y Diseño pertenecientes al Ciclo de vida del software.

**Conceptos:**

**Contador:** Un contador es una variable de tipo entero que, durante el proceso o ejecución de un programa, va aumentando su valor progresivamente.

**Acumulador:** Los acumuladores son variables que almacenan datos de una suma continua, su característica más común es que su valor suele ser siempre una suma o resta del valor actual con un dato nuevo.

**Análisis de problema:** El propósito del análisis de un problema es ayudar al programador (Analista) para llegar a una cierta comprensión de la naturaleza del problema. Una buena definición del problema, junto con una descripción detallada de las especificaciones de entrada/salida, son los requisitos más importantes para llegar a una solución eficaz.

**Prueba de escritorio:** La prueba de escritorio es un proceso manual a través del cual se busca verificar la programación y la lógica de un algoritmo antes de iniciar un programa.

**Módulo de entrada y módulo de salida:** Los módulos de Entrada / Salida, son dispositivos de estado sólido que permiten adaptar diferentes niveles de tensión desde o hacia un sistema lógico. Módulos de entrada: convierten una señal alterna o continua en una señal de nivel lógico continuo.

**Modulo de procesamiento:** El módulo de procesamiento se puede adquirir en RS como parte de un kit de desarrollo, junto con un módulo E/S que proporciona al usuario toda la conectividad necesaria la creación de prototipos.

**DESARROLLO:**

**EJERCICIOS:**

**Ejercicio 1:**

•Problema: Identificar si un número proporcionado es positivo o negativo.

• Restricciones: El número no debe ser cero.

• Datos de entrada: Un número real.

• Datos de salida: Especificar si el número es negativo o positivo.

• Dominio: Todos los números reales.

• Solución:

1. Inicio.
2. Pide al usuario que te dé un número real y pasa al siguiente paso.
3. Revisa si el número que te dieron es cero. Si es cero, vuelve a pedir un número. Si no es cero, sigue al siguiente paso.
4. Si el número es mayor que cero, significa que es positivo y sigue al paso 6. Si no es mayor, ve al paso 3.
5. Si el número es menor que cero, entonces es negativo y sigue al paso 6.
6. Fin.

Iteración:

|  |  |  |
| --- | --- | --- |
| Iteración | x | salida |
| 1 | -28 | El número es negativo |
| 2 | 0 | - |
| 3 | 30 | El número es positivo |
| 4 | -8 | El número es negativo |

**Ejercicio 2**

•Problema: Determinar cuál de dos números proporcionados es el mayor.

• Restricciones: Los números de entrada deben ser distintos entre sí.

• Datos de entrada: Dos números reales diferentes.

• Datos de salida: Indicar cuál de los dos números es mayor, ya sea el primero o el segundo.

• Dominio: Todos los números reales.

•Solución:

1. Inicia.
2. Pide al usuario que te diga el primer número y guárdalo como "número 1". Luego, pasa al siguiente paso.
3. Pide al usuario que te dé el segundo número y guárdalo como "número 2". Luego, sigue al paso 4.
4. Si "número 1" es igual a "número 2", vuelve a pedir un número. Si son diferentes, pasa al paso 5.
5. Si los números no son iguales, sigue con esto:

Si "número 1" es más grande que "número 2", entonces el primer número es el mayor. Luego, pasa al paso 6.

Si "número 2" es más grande que "número 1", entonces el segundo número es el mayor. Luego, pasa al paso 6.

6- Fin.

Iteración:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Iteración** | **Variable 1** | **Variable 2** | **Salida** |  |  |  |  |
| 1 | 300 | 87 | El primer número es mayor al segundo. |
| 2 | 62 | 83 | El segundo número es mayor al primero. |
| 3 | 47 | 85 | El segundo numero es mayor al primero. |
| 4 | 54 | 54 | - |

Ejercicio del profesor

•Problema: Solicitar un número entero al usuario. Si el usuario proporciona un número mayor a 20, multiplicar 5 por 30 y mostrar el resultado; si es menor o igual a 20, pero mayor a cero, sumar 5 más 10, y mostrar. Si es menor o igual a cero, volver a solicitar el valor.

•Restricciones: El número entero no puede ser menor a cero.

Datos de entrada: Un número entero mayor que cero.

• Datos de salida: El resultado de la operación correspondiente al número proporcionado.

• Dominio: Todos los números enteros positivos.

• Solución:

1. Inicio.
2. Pide al usuario que ingrese un número entero y sigue con el siguiente paso.
3. Si el número es cero o negativo, vuelve a pedir un número. Si es positivo, sigue al paso 4.
4. Si el número es mayor que 20, pasa al paso 6. Si no, ve al paso 5.
5. Si el número es mayor que cero, pero 20 o menos, pasa al paso 6.

6- Haz lo siguiente:

6- Si el número es mayor que 20, haz la operación "5 x 30" y muestra el resultado. Luego, pasa al paso

6- Si el número es 20 o menos pero mayor que cero, haz la operación "5 + 10" y muestra el resultado. Luego, pasa al paso 7.

7- Fin.

Iteración:

|  |  |  |
| --- | --- | --- |
| Iteración | Numero entero | Salida |
| 1 | 37 | 5 x 30 = 150 |
| 2 | 25 | 5 x 30 = 150 |
| 3 | 12 | 5 + 10 = 15 |
| 4 | -1 | - |

**Ejercicios con la hoja:**

**Ejercicio 1**

PROBLEMA: Seguir el algoritmo para obtener una figura

ENTRADA: Hoja tamaño carta en limpio, regla y lápiz.

SALIDA: Figura correcta.

Algoritmo

**1.** Dibuja una V invertida. Empieza desde el lado izquierdo, sube, y baja hacia el

lado derecho, no levantes el lápiz.

**2.** Ahora dibuja una línea en ángulo ascendente hacia la izquierda. Debe cruzar la

primera línea más o menos a 1/3 de la altura. Todavía no levantes el lápiz del

papel.

**3.** Ahora, dibuja una línea horizontal hacia la derecha. Debe cruzar la V invertida

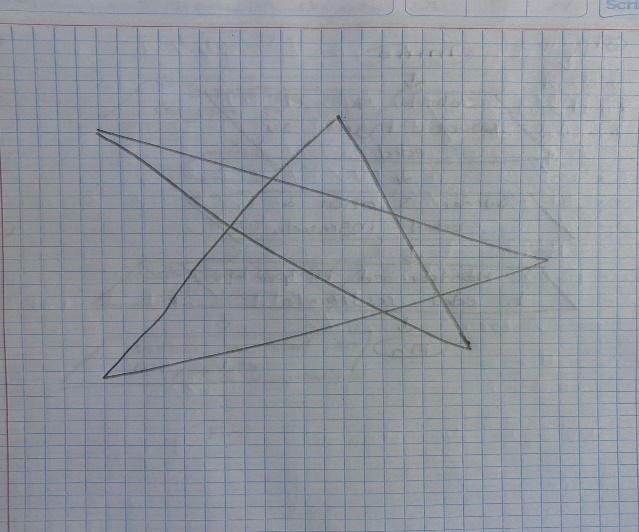
más o menos a 2/3 de la altura total. Sigue sin levantar el lápiz.

**4.** Dibuja una línea en un ángulo descendente hasta el punto de inicio. Las líneas

deben unirse.

**5.** Ahora ya puedes levantar el lápiz del papel. Has terminado la estrella de 5

puntas.



**Ejercicio 2**

PROBLEMA: Seguir el algoritmo para obtener una figura

ENTRADA: Hoja tamaño carta en limpio, regla y lápiz.

SALIDA: Figura correcta.

Algoritmo

**1.** Empieza dibujando un círculo con un compás. Coloca un lápiz en el compás.

Coloca la punta del compás en el centro de una hoja de papel.

**2.** Ahora gira el compás, mientras mantienes la punta apoyada en el papel. El lápiz

dibujará un círculo perfecto alrededor de la punta del compás.

**3.** Marca un punto en la parte superior del círculo con el lápiz. Ahora, coloca la

punta del compás en la marca. No cambies el radio del compás con que hiciste el

círculo.

**4.** Gira el compás para hacer una marca en el propio círculo hacia la izquierda. Haz

una marca también en el lado derecho.

**5.** Ahora, coloca la punta del compás en uno de los puntos. Recuerda no cambiar el

radio del compás. Haz otra marca en el círculo.

**6.** Continúa moviendo la punta del compás a las otras marcas, y continúa hasta que

tengas 6 marcas a la misma distancia unas de otras. Ahora, ya puedes dejar tu

compás a un lado.

**7.** Usa una regla para crear un triángulo que empiece en la marca superior del

círculo. Coloca el lápiz en la marca superior. Ahora dibuja una línea hasta la

segunda marca por la izquierda. Dibuja otra línea, ahora hacia la derecha,

saltándote la marca de la parte más baja. Complementa el triángulo con una línea

hacia la marca superior. Así completarás el triángulo.

**8.** Crea un segundo triángulo empezando en la marca en la base del círculo. Coloca

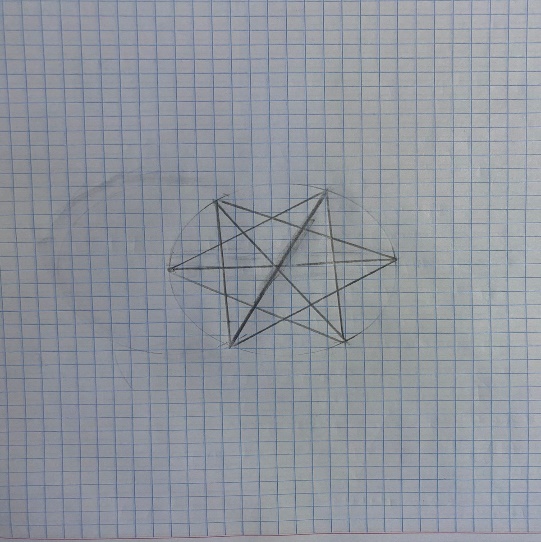
el lápiz en la marca inferior. Ahora conéctala con la segunda marca hacia la

izquierda. Dibuja una línea recta hacia la derecha, saltándote el punto superior.

Completa el segundo triángulo dibujando una línea hasta la marca en la parte

inferior.

**9.** Borra el círculo. Has terminado de dibujar tu estrella de 6 puntos.



**TAREA:**

Problema: Genera un algoritmo para resolver una ecuación de segundo grado.

Datos de entrada: Valores a, b, c.

Datos de salida: Deben salir 2 resultados.

Dominio: Todos los números reales y no reales.

1. Inicio.
2. Pedirle al usuario que introduzca los valores a,b,c.
3. Calcular el discriminante, si el discriminante es mayor o igual a 0 pasar al siguiente paso y su resultado será real, si el discriminante es menor a 0 pasar al siguiente paso y su resultado será no real.
4. Si el discriminante es mayor o igual a 0 dará como resultado x1 y x2 serán reales.
5. Si el discriminante es menor a 0 dará como resultado x1 y x2 serán no reales.
6. Mostrar los resultados al usuario.
7. Fin.

**Referencias:**

Universidad Autónoma de Ciudad Juárez. (s.f.). Fundamentos de programación: Unidad 4.1. Recuperado de <https://www.uacj.mx/CGTI/CDTE/JPM/Documents/IIT/fund_programacion/U4-1.html>

Platzi. (n.d.). Contadores, acumuladores y centinelas. Recuperado de <https://platzi.com/tutoriales/1050-basico-programacion/934-contadores-acumuladores-y-centinelas/>

Domingo, J. (s.f.). Análisis de problemas y diseño de algoritmos. PLEDIN. Recuperado el 1 de septiembre de 2024, de <https://plataforma.josedomingo.org/pledin/cursos/curso_cpp1/curso/u01/>

Industria Embebida Hoy. (n.d.). Módulo de procesamiento. Recuperado de <https://www.industriaembebidahoy.com/modulo-de-procesamiento/>.

• Raghu Singh (1995). International Standard ISO/IEC 12207 Software Life Cycle

Processes. Agosto 23 de 1996, de ISO/IEC. Consulta: Junio de 2015. Disponible en:

<http://www.abelia.com/docs/12207cpt.pdf>

• Carlos Guadalupe (2013). Aseguramiento de la calidad del software (SQA). [Figura

1]. Consulta: Junio de 2015. Disponible en:

https://www.mindmeister.com/es/273953719/aseguramiento-de-la-calidad

delsoftware-sqa

• Andrea S. (2014). Ingeniería de Software. [Figura 2]. Consulta: Junio de 2015.

Disponible en: <http://ing-software-verano2014.blogspot.mx>

• Michael Littman. (2012). Intro to Algorithms: Social Network Analysis. Consulta

Junio de 2015, de Udacity. Disponible en:

<https://www.udacity.com/course/viewer#!/c-cs215/l-48747095/m-48691609>