|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Rodríguez Espino Claudia |
| *Asignatura:* | Fundamentos de Programación |
| *Grupo:* | 1102 |
| *No de Práctica(s):* | 3 |
| *Integrante(s):* | Morales Luna Emilio |
|  |  |
|  |  |
| *Semestre:* | 2018-1 |
| *Fecha de entrega:* | 19 de agosto de 2017 |
| *Observaciones:* |  |
|  |  |

**Guía práctica de estudio 03: Solución de problemas y Algoritmos.**

**Objetivo:**

Elaborar algoritmos correctos y eficientes en la solución de problemas siguiendo las etapas de Análisis y Diseño pertenecientes al Ciclo de vida del software.

**Actividades:**

* A partir del enunciado de un problema, identificar el conjunto de entrada y el conjunto de salida.
* Elaborar un algoritmo que resuelva un problema determinado (dado por el profesor), identificando los módulos de entrada, de procesamiento y de salida.

**Introducción**

Un problema informático se puede definir como el conjunto de instancias al cual corresponde un conjunto de soluciones, junto con una relación que asocia para cada instancia del problema un subconjunto de soluciones (posiblemente vacío).

Para poder solucionar un problema nos apoyamos en la Ingeniería de Software que de acuerdo con la IEEE se define como “La aplicación de un enfoque sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento del software". Por lo que el uso y establecimiento de principios de ingeniería sólidos, son básicos para obtener un software que sea económicamente fiable y funcione eficientemente.

La Ingeniería de Software provee métodos que indican cómo generar software. Estos métodos abarcan una amplia gama de tareas:

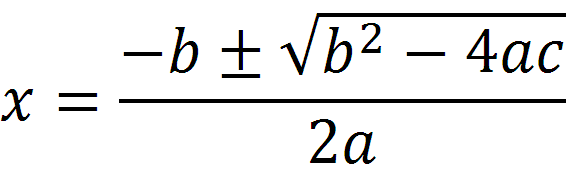
 Planeación y estimación del proyecto.

 Análisis de requerimientos del sistema y software.

 Diseño de la estructura de datos, la arquitectura del programa y el procedimiento algorítmico.

 Codificación.

 Pruebas y mantenimiento (validación y verificación).

**Desarrollo**

Un algoritmo para realizar la formula general.

1.- Inicio

2.- Pedir el valor **a**. e implicar que no debe ser 0

3.- Si **a** = cero, regresar al paso 2. Si **a** no es 0 seguir al paso número 4.

4.- Pedir el valor de b y c

5.- Crear la variable d la cual vale []

6.- Si d < 0 ir al paso 7, si d > 0 ir al paso 8.

7.- Si d < 0 entonces realizar = Z con unidad i, hacer operación ,

8.- Imprimir x1.1= r + ni y y x1.2= r – ni, en ese caso contrario, ir al paso 9.

9.- = x1 , = x2

10.- Imprimir x1 y x2

11.-Fin

Variables demostradas

El usuario asigna los siguientes valores

A=1

B=1

C=1

x1.1= r + ni

x1.2= r – ni

X1=

X2=

D=

Prueba de escritorio

Después de pedir las variables, se realiza d:

(1)²- 4(1)(1) = -3 ……Es menor a 0

Como es menor entonces realizar z= √d(-1) con unidad i de imaginario.

√-3(-1) = √3

=1.77

Realiza la operación n y r:

Finalmente realiza la operación:

x1.1= r + ni

x1.2= r – ni

Algoritmo para realizar una factorial de 1 – 10

1.- Inicio

2.- Agregar las variables contador= 1 y factorial = 1

2.- Pedir número: m y decir que el número debe ser mayor a 0 y menor de 11

3.- Si 0 < m < 11 ir al paso 4, si no, regresar al 2.

4.- Mientras el contador sea igual o menor a 0 y menor o igual a m. Si no, ir al paso 7.

5.- fact = fact(contador)

6.- contador ++

7.- Termina

El contador= 1 y la factorial = 1

El usuario asigna el valor m = 5

fact = fact(contador)

contador = contador +1=2

fact= 1\*2 = 2

fact= 2\*3 = 6

fact= 6\*4 = 24

fact= 24\*5 = 120

Algoritmo que realiza 2 operaciones dependiendo del valor de y

Si y es menor a 2 x1= y² + 4y - 25

Si y es mayor a 2 x2= 4y² - 3y + 0

1.- Inicio

2.- Pedir y

3.- Si y es menor a 2 ir al paso 4, si es, mayor a 2 ir al paso 6

4.- x1= y² + 4y - 25

5.- Imprimir x1 y el valor de y, ir al paso 8

6.- x2= 4y² - 3y + 0

5.- Imprimir x2 con valor de y, ir al paso 8

8.- Fin

Y= 3

Y es mayor a 2, entonces se hace x2

x2= 4(3) ² - 3(3) + 0

x2=4(9)-9

x2= 36-9

x2=27

Terminar

Ejercicio 1

PROBLEMA: Seguir el algoritmo para obtener una figura

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

SALIDA: Figura correcta.

Algoritmo

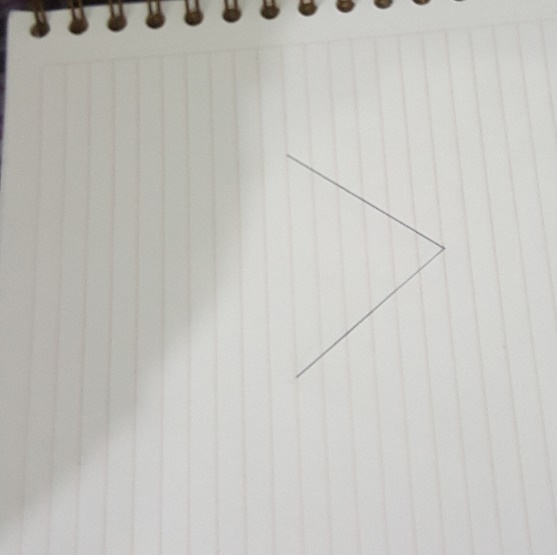
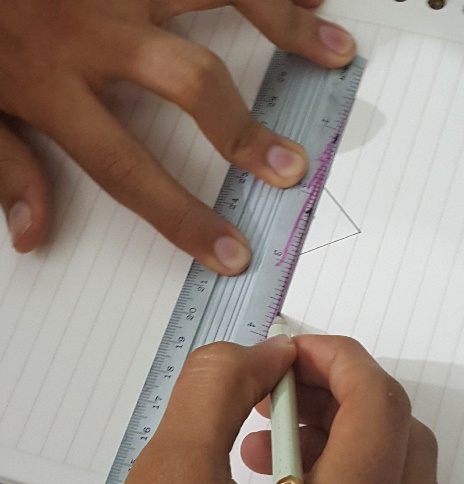
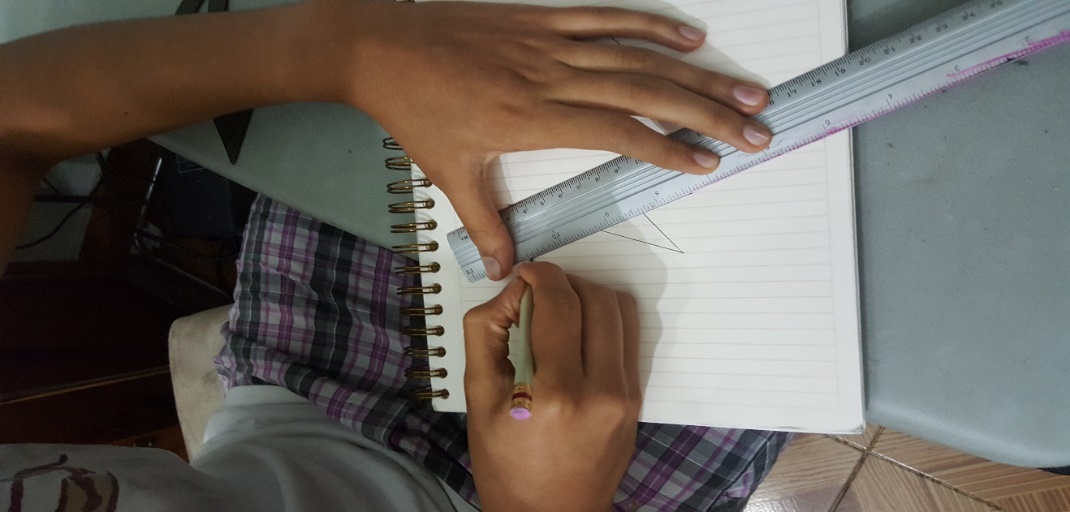
1.Dibuja una V invertida. Empieza desde el lado izquierdo, sube, y baja hacia el lado derecho, no levantes el lápiz.

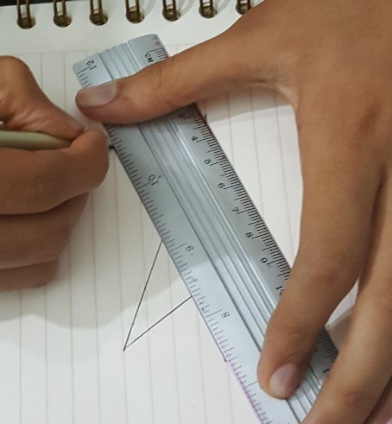
2. Ahora dibuja una línea en ángulo ascendente hacia la izquierda. Debe cruzar la primera línea más o menos a 1/3 de la altura. Todavía no levantes el lápiz del papel.

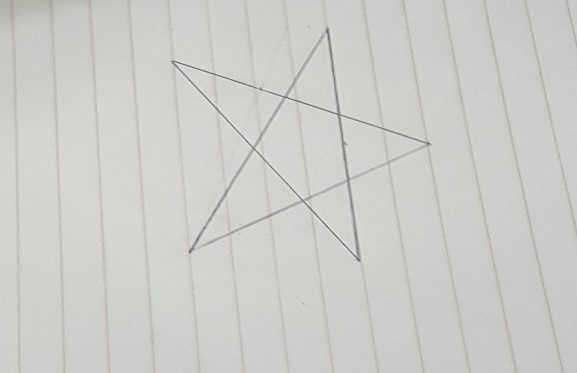
3. Ahora, dibuja una línea horizontal hacia la derecha. Debe cruzar la V invertida más o menos a 2/3 de la altura total. Sigue sin levantar el lápiz.

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.







Conclusiones

El hecho de poder realizar algoritmos para poder solventar cualquier tipo de problema de manera ordenada y específica, claramente significa que podemos tomar a cabo la programación sin problemas.

Para los que van a empezar a programar esta manera de empezar es una buena base, para formar su manera de pensar. Que sea lógica, sin ambigüedades y directa.

Los ejercicios que hicimos fueron supervisados en clase para resolver dudas y al final uno de tarea que revisamos superficialmente.

Por último, hicimos un ejercicio de escritorio que consistía en un dibujo, tomé fotos del procedimiento de manera canónica y mostrando el resultado final.

Esta práctica me sirvió para pulir mis habilidades lógicas, debido a que no practico mucho la programación.