UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

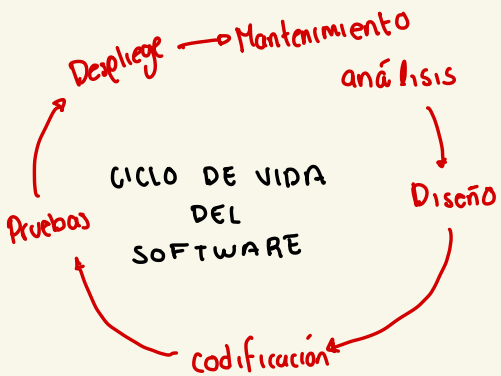
Fundamentos de Programación

Serie III

Nombre del alumno: Anhoar Villanueva Noguedas Grupo: 22

Número de cuenta: _____

1.- Dibuja el ciclo de vida del software y describe sus partes



Análisis: Como su nombre lo indica se encarga de analizar el planteamiento del problema o necesidad para despues diseñar la idea principal de como resolverlo. También llamado Definición de necesidades

Diseño: Se encarga de plasmar la idea ya sea en pseudocódigo o diagrama de flujo.

Codificación: Trasladar la idea a un lenguaje de programación real

Pruebas: Las llamadas pruebas de escritorio se encarga de verificar puntos críticos que pueda tener el software.

Mantenimiento: Cambiar o mejorar el proyecto trabajado

2.- ¿Qué es un algoritmo? Describe sus características y da un ejemplo de cada una

Es una serie de pasos que tienen como objetivo final resolver alguna necesidad

Tiene pasos precisos, definidos y finitos

- Características:
  - Definido: no importa cuantas veces lo repitas, siempre tendrás el mismo resultado

    Ejemplo: Alarma diaria; siempre sonará tu alarma en el día y hora precisa que tu establezcas
  - Preciso: Tiene un orden preciso a cumplir.

    Ejemplo: Seguir una receta.
  - Finito: Tiene un número de pasos exactos

    Ejemplo: Receta de cocina, Instrucción para armar algo, etc.
  - Eficiente: Cumple con su función

    Ejemplo: Un automóvil, una taza, etc.
  - Legible: Su descripción es clara para cualquier usuario.

    Ejemplo: Un texto con buena escritura.

3.- Describe la teoría de la computabilidad

Es aquella que se encarga de estudiar los modelos computacionales y como estos son esenciales en la resolución de problemas

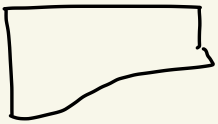
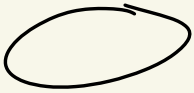
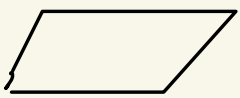
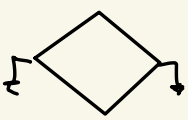
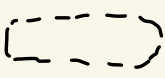
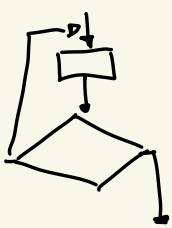
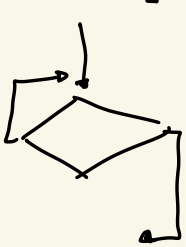
¿Qué son los modelos computacionales?

- Es un modelo matemático en las ciencias de la computación que requiere extensos recursos computacionales para el estudio de un sistema complejo.

4.- Características del diagrama de flujo

- Es una forma visual de representar un modelo algorítmico.
- Contiene una simbología que ayuda al usuario tener una referencia de que hace el diagrama.
- Tiene un inicio y un fin
- Todas sus lineas están conectadas
- Tiene comentarios
- Va de arriba hacia abajo
- No puede llegar más de una linea al símbolo

5.- Simbología en pseudocódigo y codificación

Diagrama	Pseudocódigo	codificación
	Escribir	printf
	Inicio / fin	{ }
	Leer	scanf
	Si No	if else
	//comen.	// comentario
	Hacer mientras	Do While
	Mientras	while

6.- Tipos de funciones con parámetros y tipos de datos de retorno (4 tip

1. Función con parámetro y tipo de dato de retorno:

```
int NombreFunción (int a, int b)
```

2. Función con parámetro y sin datos de retorno:

```
NombreFunción (int a, int b)
```

3. Función sin parámetro pero con dato de retorno.

```
Int NombreFunción ( )
```

4. Función sin parámetros y datos de retorno: NombreFunción ( )

7.- Codificación de estructuras iterativas y condicionales

```
if (condición) {
}
```

```
for (var, condición, incremento){
}
```

```
do {
  // Acción
} while (condición);
```

```
if (condición) {
} else {
}
```

```
while (condición) {
}
```

```
switch (varOpc) {
case n:
  //
break;
}
```

8.- Etapas del análisis

El análisis es crucial, pues este se encarga de conocer y analizar la necesidad de el usuario.

Se trabajan dos puntos importantes:

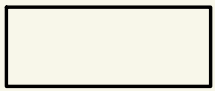
Conjunto de entrada / datos de entrada

Todo aquel dato que se necesita para que el programa trabaje.
Estos datos son provenientes de la necesidad de el usuario.

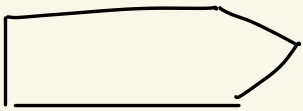
Conjunto de salida / datos de salida

Son los datos que queremos que resulten al ejecutar el programa; estos también provienen de la necesidad del usuario.

9.- Figuras utilizadas en diagrama de flujo

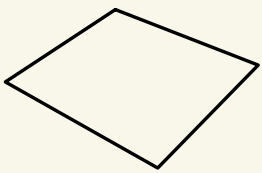


Acciones



Llamado a función.

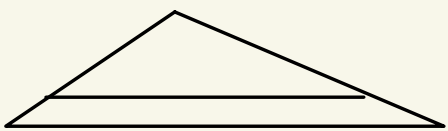
Conexión de hoja



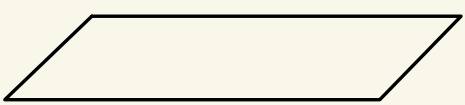
condicional

(de aquí derivan ciclos / iteracciones)

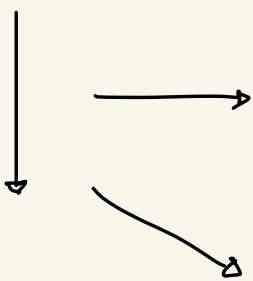
Conexión de páginas



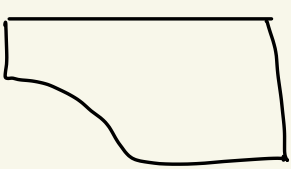
condicional múltiple



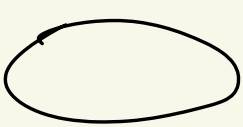
Leer



Dirección de Diagrama.



Escribir



Inicio / fin

10.- Uso de break y tipos de datos booleanos

Break

El uso del break usual, es para detener algún ciclo; al detener un ciclo con break el programa sigue con el flujo del código una vez ya detenido la acción con el break.

Se usa mucho en la condición múltiple.

```
switch (opc) {

case 1 :
  // Acción
break;
case 2 :
  // Acción
break;
```

Booleanos

Los booleanos son tipos de datos con dos tipos de "valores" indican un estado ya sea falso o verdadero.

boolean { falso, verdadero }