* ***LENGUAJE C: CARACTERÍSTICAS.***

1. ***ES COMPILADO:*** Convierte el código fuente en ficheros objeto. Estos son enlazados con librerías necesarias dando lugar a un fichero ejecutable.
2. ***ES DE NIVEL MEDIO:*** Combina elementos de lenguaje de alto nivel con la funcionalidad del lenguaje ensamblador. Es decir que **trabaja a un nivel más cerca de la computadora.**
3. ***ES ESTRUCTURADO:*** Permite crear procedimientos en bloques dentro de otros procedimientos.
4. ***ES PORTABLE:*** Nos permite utilizar el mismo código en diferentes equipos y sistemas informáticos.
5. ***ES RELATIVAMENTE PEQUEÑO:*** Es económico en cuanto a expresiones se refiere, se puede describir en poco espacio y es fácil y rápido de entender.
6. ***ES ABUNDANTE EN CUANTO A OPERADORES Y TIPOS DE DATOS:*** Posee un operador para cada una de las posibles operaciones en código de máquina.

* ***LENGUAJE C: DESVENTAJAS.***

1. Carece de instrucciones entrada/salida.
2. Carece de instrucciones de manejo de cadena de caracteres.
3. La precedencia de los operadores dificultan la comprensión de algunas expresiones.

* ***IDENTIFICADORES:*** Nombres dados a constantes, variables y funciones. Debe siempre empezar con una letra o el carácter de subrayado.
* ***CONSTANTES:*** Valores que una vez fijados por el compilador, no cambian durante la ejecución del programa. Su valor puede ser un número, un carácter o una cadena de caracteres. Se utiliza la directiva ***#define***.
* ***TIPOS DE DATOS:***

1. ***Int:*** Almacenan **números enteros sin decimales.**
2. ***Char:*** Almacenan **caracteres**.
3. ***Float:*** Almacenan **números decimales.**
4. ***Double:*** Almacenan **números reales.**

* ***FUNCIONES GENERALES:***

1. ***Scanf():*** Se utiliza para que el usuario pueda cargar por teclado un dato que va a ser almacenado dentro de una variable.
2. ***Printf():*** Se utiliza para imprimir por pantalla tanto un texto como el contenido de una variable.
3. ***Getch(); Getche(); Getc(); Getchar(); Gets():*** Se utilizan para que el Usuario ingrese un caracter por teclado.

* ***OPERADORES:***

1. ***LÓGICOS:*** Permiten hacer comparaciones entre dos o más variables. Estos son:

* **&&:** Conjunción Y.
* **||:** Disyunción Ó.
* **!:** Distinto de…

1. ***ARITMÉTICOS:*** Permiten realizar operaciones tales como suma, resta, división y multiplicación. Estos son:

* **+ :** Suma
* **- :** Resta
* **/ :** División
* **\* :** Multiplicación
* **% :** Resto de una división entera
* **++ :** Incremento en 1
* **-- :** Decremento en 1

1. ***ASIGNACIÓN:*** Permiten darle (asignar) un valor a una variable. Estos son:

* **= :** Asignación Simple
* **\*= :** Multiplicación + Asignación
* **/= :** División + Asignación
* **%= :** Mod + Asignación
* **+= :** Suma + Asignación
* **-= :** Resta + Asignación

1. ***RELACIÓN:*** Permiten establecer una relación entre dos variables. Estos son:

* < **:** Menor
* > **:** Mayor
* <= **:** Menor o igual
* >= **:** Mayor o igual
* == **:** Igualdad
* != **:** Distinto de…
* ***BUCLES:*** Se utilizan al querer repetir una tarea una determinada cantidad de veces.
* ***TIPOS DE BUCLES:***

1. ***FOR:*** Se declara un valor inicial, una condición y un contador, y se ejecuta el programa hasta que la condición no se cumpla.
2. ***WHILE:*** Se declara una condición, y el programa se ejecuta hasta que ésta ya no se cumpla.
3. ***DO-WHILE:*** Se ejecuta, al menos, una vez. Realiza las sentencias y luego evalúa la condición. Se ejecuta hasta que ésta ya no se cumpla.

* ***SENTENCIAS CONDICIONALES:***

1. ***IF:*** Si se cumple su condición, se ejecuta la sentencia.
2. ***IF-ELSE:*** Si no se cumple la condición del IF, sigue con las instrucciones del ELSE.
3. ***SWITCH:*** Analiza una condición y según su valor, ejecuta una de las opciones (cases).

* ***SENTENCIAS DE SALTOS:***

1. ***BREAK:*** Lleva la ejecución del programa a salir del bucle de manera forzada.
2. ***CONTINUE:*** No finaliza el bucle, pero suspende el presente ciclo.
3. ***RETURN:*** Sale de la función en la que se encuentra y devuelve el control a la instancia o rutina que la llamó, **opcionalmente** con un valor de retorno.

* ***PILAS:*** Estructura de datos en la que el acceso a sus elementos es de tipo **LIFO** (Last In, First Out). Permite almacenar y recuperar datos.
* ***OPERACIONES CON PILAS:***
* ***CREAR:*** Se crea la pila.
* ***INICIALIZAR:*** Se deja la pila vacía, lista para ser usada.
* ***APILAR:*** Se añade un elemento a la pila.
* ***DESAPILAR:*** Se elimina el elemento superior de la pila.
* ***LEER:*** Agrega al tope de la pila un elemento ingresado por teclado.
* ***TOPE:*** Devuelve el valor del elemento que está en la cima de la pila. **NO LO ELIMINA.**
* ***PilaVacia:*** Devuelve **CIERTO** si la pila está vacía, o **FALSO** de lo contrario.
* ***MOSTRAR:*** Muestra por pantalla el contenido de la pila.
* ***FUNCIONES:*** Se utilizan para descomponer al programa general en subprogramas. Conjunto de instrucciones que realizan una tarea específica. En general, toman valores de entrada (**parámetros**) y proporcionan un valor de retorno.
* ***TIPOS DE PARÁMETROS:*** Variables que reciben valores de entrada.

1. ***FORMALES:*** Están puestos en la definición de la función. Se decoran dentro de la lista de parámetros con un **nombre** y su **tipo de dato**. Al declararlos no contienen datos.
2. ***ACTUALES:*** Se usan en la invocación de la función. Son variables del programa invocante y se ubican reemplazando a los parámetros formales.
3. ***PASAJE DE PARÁMETROS:*** Existen dos formas: Por **valor** (o copia), y por **referencia.** Por copia, si el dato se modifica en la función, no cambia en su parámetro actual. Por referencia, si el dato se modifica en la función, **también** se modifica en su parámetro actual.

* ***TIPOS DE VARIABLES:***
* ***LOCALES:*** Son aquellos que se declaran dentro de una función. **Nacen y mueren solamente en esa función.**
* ***GLOBALES:*** Son aquellas que se declaran **fuera de toda función.** Son accesibles desde cualquier parte del programa. **No deben usarse.**
* ***PUNTEROS:*** Son **direcciones de memoria** asociadas a un identificador y a un tipo de dato que almacena **otra dirección de memoria** de un dato del mismo tipo que del puntero.
* ***OPERADORES DE PUNTEROS:***

1. **& :** Se utiliza para **ASIGNAR** valores a datos de tipo puntero.
2. **\* :** Se utiliza para **ACCEDER** a los contenidos a los que apunta el puntero.
3. **= :** Se utiliza para **ASIGNAR** a un puntero, una dirección de memoria concreta, la dirección de una variable o el contenido de otro puntero.

* ***ARREGLOS:*** Conjuntos de variables del **mismo tipo** que se diferencian entre sí por sus posiciones y se encuentran ordenadas a través de un sub índice.

***NOTA:*** En la **DECLARACIÓN** de un arreglo, el número entre corchetes es el **NÚMERO DE ELEMENTOS** que tiene ese arreglo. Al **USAR** el arreglo, el número entre corchetes es el **SUB ÍNDICE.**

* ***ARREGLOS PARALELOS:*** Son aquellos que están relacionados por un **MISMO ÍNDICE**. O sea, las celdas ubicadas en la misma posición para distintos arreglos. Sus datos tienen un vínculo entre sí.
* ***ARREGLOS DINÁMICOS:*** Son aquellos cuyo tamaño es definido en tiempo de ejecución.
* ***ORDENAMIENTO POR SELECCIÓN:***

Para ordenar mediante el método de selección, lo que se hace es, tomar y considerar al primer elemento (de izquierda a derecha) como el **menor de todos**, y luego ir **recorriendo todo el arreglo en busca de un elemento menor**.

En caso de encontrarlo, reemplazará al que ya es considerado como el menor y los cambiará de lugar, para luego repetir el proceso. Si de lo contrario no encuentra algún elemento menor que el que ya es considerado como el menor, pero que es el más próximo que le sigue, lo pondrá al lado de ese primer elemento, para luego repetir el proceso.

De esta forma se va ordenando el arreglo por selección. Porque selecciona de a menores haciendo las respectivas comparaciones.

* ***ORDENAMIENTO POR INSERCIÓN:***

Para ordenar mediante método de inserción, lo que se hace es, tomar la parte izquierda del arreglo y considerarlo como la **parte ordenada**.

A diferencia del método de ordenación por selección, el método por inserción **no recorre todo el arreglo**, sino que a medida que lo va recorriendo, va insertando cada elemento en el lugar que le corresponde (previamente generando un hueco) mediante comparaciones entre los distintos elementos del arreglo.

* ***MÁS CONCEPTOS:***

1. ***BUFFER:*** Porción de memoria en una computadora (o instrumento digital) reservada para el **almacenamiento temporal** de información digital, mientras que está esperando a ser procesada.
2. ***LENGUAJE TIPADO:*** Es tipado cuando, por ejemplo, en una variable entera **NO SE PUEDE** guardar un valor de **DIFERENTE TIPO**, por ejemplo, un decimal.
3. ***ALGORITMO:*** Serie de pasos ordenados que se utilizan para resolver un problema.
4. ***ALGORITMO: CARCTERÍSTICAS:***

* ***PRECISO:*** Debe tener **instrucciones claras.**
* ***DEFINIDO:*** Si se sigue más de una vez, **siempre nos conduce al mismo resultado.**
* ***FINITO:*** Cuenta con un determinado número de pasos con un **inicio** y un **fin.**