**Lenguajes de Programación**

**Tarea 2**

**Maestro:** M. A. Ismael Gutiérrez Pimentel

**Hora:** V6 – **Día:** L,M,V – **Grupo:** 010

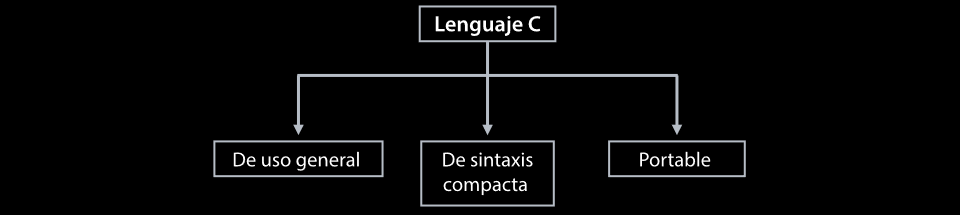
**Salón:** 4201

|  |  |  |
| --- | --- | --- |
| **NOMBRE** | **MATRÍCULA** | **Carrera** |
| Carlos Arturo Nava Matamoros | 1889318 | ITS |

*Ciudad Universitaria, San Nicolás de los Garza*

***DEFINICIÓN DE LENGUAJES DE PROGRAMACIÓN***

En términos generales, un lenguaje de programación es una herramienta que permite desarrollar software o programas para computadora. Los lenguajes de programación son empleados para diseñar e implementar programas encargados de definir y administrar el comportamiento de los dispositivos físicos y lógicos de una computadora. Lo anterior se logra mediante la creación e implementación de algoritmos de precisión que se utilizan como una forma de comunicación humana con la computadora.  
  
A grandes rasgos, un lenguaje de programación se conforma de una serie de símbolos y reglas de sintaxis y semántica que definen la estructura principal del lenguaje y le dan un significado a sus elementos y expresiones.  
  
Programación es el proceso de análisis, diseño, implementación, prueba y depuración de un algoritmo, a partir de un lenguaje que compila y genera un código fuente ejecutado en la computadora.  
  
La función principal de los lenguajes de programación es escribir programas que permiten la comunicación usuario-máquina. Unos programas especiales (compiladores o intérpretes) convierten las instrucciones escritas en código fuente, en instrucciones escritas en lenguaje máquina (0 y 1).



*Características del lenguaje C*

***CLASIFICACIÓN DE LOS LENGUAJES DE ALTO NIVEL***

* **Lenguajes de programación imperativos:** entre ellos tenemos el Cobol, Pascal, C y Ada.
* **Lenguajes de programación declarativos:** el Lisp y el Prolog.
* **Lenguajes de programación orientados a objetos:** el Smalltalk y el C++.
* **Lenguajes de programación orientados al problema:** son aquellos lenguajes específicos para gestión.
* **Lenguajes de programación naturales:** son los nuevos lenguajes que pretender aproximar el diseño y la construcción de programas al lenguaje de las personas.

***EJEMPLOS DE LENGUAJES***

*INTERPRETADOS*

Un **lenguaje de programación interpretado** es aquel que el código fuente se ejecuta directamente, instrucción a instrucción.

* Java
* C#
* Visual Basic .NET
* JavaScript
* VBScript
* MATLAB
* GML
* Perl
* Python
* Lisp
* Ruby
* PHP
* ASP

*COMPILADOS*

Un lenguaje compilado es aquel cuyo código fuente, escrito en un lenguaje de alto nivel, es traducido por un compilador a un archivo ejecutable entendible para la máquina en determinada plataforma.

* *C*
* *C++*
* *Objective-C*
* *C#*
* *Fortran*
* *Pascal*
* *Haskell*
* *Visual Basic*

*ORIENTADOS A OBJETOS*

Los lenguajes de programación orientados a objetos tratan a los programas como conjuntos de objetos que se ayudan entre ellos para realizar acciones. Entendiendo como objeto al entidades que contienen datos. Permitiendo que los programas sean más fáciles de escribir, mantener y reutilizar.

* C++
* Objective C.
* Java.
* Smalltalk.
* Eiffel.
* Lexico (en castellano)
* Ruby.
* Python.

*FUNCIONES DE LOS LENGUAJES INTERPRETADOS Y COMPILADOS*

* Los lenguajes compilados muy raro van a poder generar programas multiplataforma.
* Los lenguajes compilados son menos flexibles que los lenguajes interpretados.
* Los lenguajes compilados generar un archivo ejecutable del que ya no tenemos que preocuparnos, simplemente ejecutar.
* Cuando arreglamos un error en un lenguaje compilado tendremos que volver a compilar el programa para que el ejecutable se vuelva a generar con los cambios.
* Los lenguajes interpretados son normalmente multiplataforma, cada S.O. tiene su intérprete del lenguaje.
* Cada que vamos a ejecutar un programa que es de un lenguaje interpretado tendremos que interpretar el archivo, el programa compilado ya está listo para funcionar por eso es más rápido.

*FUNCIONES DE LOS LENGUAJES DE PROGRAMACIÓN ORIENTADOS A OBJETOS*

* **Reusabilidad**. Cuando hemos diseñado adecuadamente las clases, se pueden usar en distintas partes del programa y en numerosos proyectos.
* **Mantenibilidad**. Debido a las sencillez para abstraer el problema, los programas orientados a objetos son más sencillos de leer y comprender, pues nos permiten ocultar detalles de implementación dejando visibles sólo aquellos detalles más relevantes.
* **Modificabilidad**. La facilidad de añadir, suprimir o modificar nuevos objetos nos permite hacer modificaciones de una forma muy sencilla.
* **Fiabilidad**. Al dividir el problema en partes más pequeñas podemos probarlas de manera independiente y aislar mucho más fácilmente los posibles errores que puedan surgir.

***BIBLIOGRAFÍA***

[***https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/1023/mod\_resource/content/1/contenido/index.html***](https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/1023/mod_resource/content/1/contenido/index.html)

[***http://www.larevistainformatica.com/clasificacion-de-los-lenguajes-de-programacion.html***](http://www.larevistainformatica.com/clasificacion-de-los-lenguajes-de-programacion.html)

[***https://lenguajesdeprogramacion.net/diccionario/que-es-un-lenguaje-interpretado/***](https://lenguajesdeprogramacion.net/diccionario/que-es-un-lenguaje-interpretado/)

[***https://otroespacioblog.wordpress.com/2012/09/02/lenguajes-de-programacion-compilados-vs-interpretados/***](https://otroespacioblog.wordpress.com/2012/09/02/lenguajes-de-programacion-compilados-vs-interpretados/)

[***https://ed.team/blog/lenguajes-compilados-vs-lenguajes-interpretados***](https://ed.team/blog/lenguajes-compilados-vs-lenguajes-interpretados)