



**UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA**

*La Universidad Católica de Loja*

**UNIVERSIDAD TÉCNICA PARTICULAR  
DE LOJA**

**ESCUELA:**

**ELECTRÓNICA Y TELECOMUNICACIONES**

**MATERIA:**

**METODOLOGIA DE LA PROGRAMACIÓN**

**NOMBRES Y APELLIDOS:**

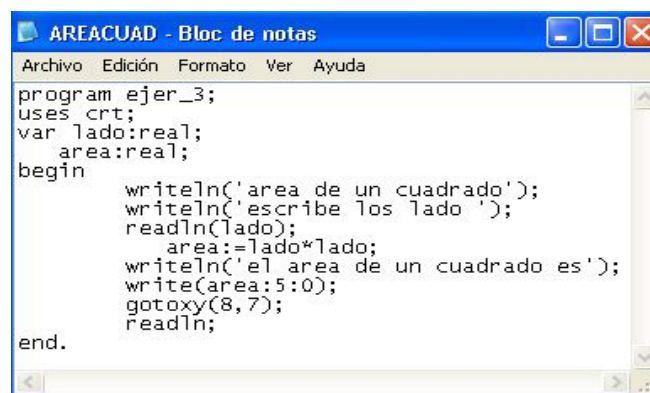
**JUAN CARLOS PEÑA SALAZAR.**

**PERIODO - LECTIVO:**

**SEPTIEMBRE 2007-FEBRERO 2008**

**C**ompilador. Para poder hablar acerca del compilador primeramente debemos tener un concepto claro de que es compilador y de todos los elementos que lo forman, sin embargo es fundamental tener una reseña histórica acerca del compilador a continuación se presenta una breve reseña histórica. Los compiladores nacen a medida de que las computadoras fueron evolucionando, ya que se tenía la necesidad de un programa el cual tenga la función de ejecutar cada una de las instrucciones y de expresar las distintas acciones a realizar de una manera más sencilla posible para el hombre con el computador y a medida que iban evolucionaban las investigaciones en el año de 1952 aparece el primer compilador realizado por Grace Hopper para el lenguaje de programación A-0 y este compilador sirve de base para la aparición del primer compilador auto contenido, es decir, capaz de compilar su propio código fuente, el cual fue creado por tres personas: Lisp, Hart y Levin en 1962. Desde 1970 se ha convertido en una práctica común escribir el compilador en un mismo lenguaje que este compila, aunque Pascal y C han sido alternativas muy usadas.

Así podemos decir que compilador es un programa informático que tiene como función principal el traducir un código fuente fig.1 (el código fuente es simplemente un texto el cual es capaz de ser leído por cualquier editor de textos) el cual esta escrito en cualquier lenguaje de programación alto nivel (por ejemplo PASCAL, TURBO PASCAL, TURBO C, C, C++) el cual es entendible por el ser humano a otro lenguaje al cual se lo conoce como lenguaje de máquina el cual es obviamente entendible por la computadora.



```
program ejer_3;
uses crt;
var lado:real;
    area:real;
begin
    writeln('area de un cuadrado');
    writeln('escribe los lado ');
    readln(lado);
    area:=lado*lado;
    writeln('el area de un cuadrado es');
    write(area:5:0);
    gotoxy(8,7);
    readln;
end.
```

Fig.1 Código Fuente

Continuando con el tema, el compilador esta conformado por dos partes como son el Front End y Back End a continuación se describe cada uno de ellos:

- **Front End.\_** Esta parte del compilador se encarga de analizar el código fuente es decir verifica que este bien realizado, genera el árbol de derivación y rellena los valores de la tabla de símbolos, esta parte del compilador es independiente de cada plataforma de los distintos sistemas operativos.
- **Back End.\_** Esta parte del compilador es contraria al front end ya que se encarga de generar el código máquina a partir de los resultados de la fase de análisis, realizada por el Front End.

Estas dos partes están divididas por diferentes razones ya que Back End puede generar el código máquina y Front End puede analizar el código fuente de los diferentes lenguajes de programación.

Después de haber mencionado las dos partes de un compilador pasamos a describir la estructura de un compilador:

El compilador esta formado por cuatro módulos uno independientemente del otrFig.2

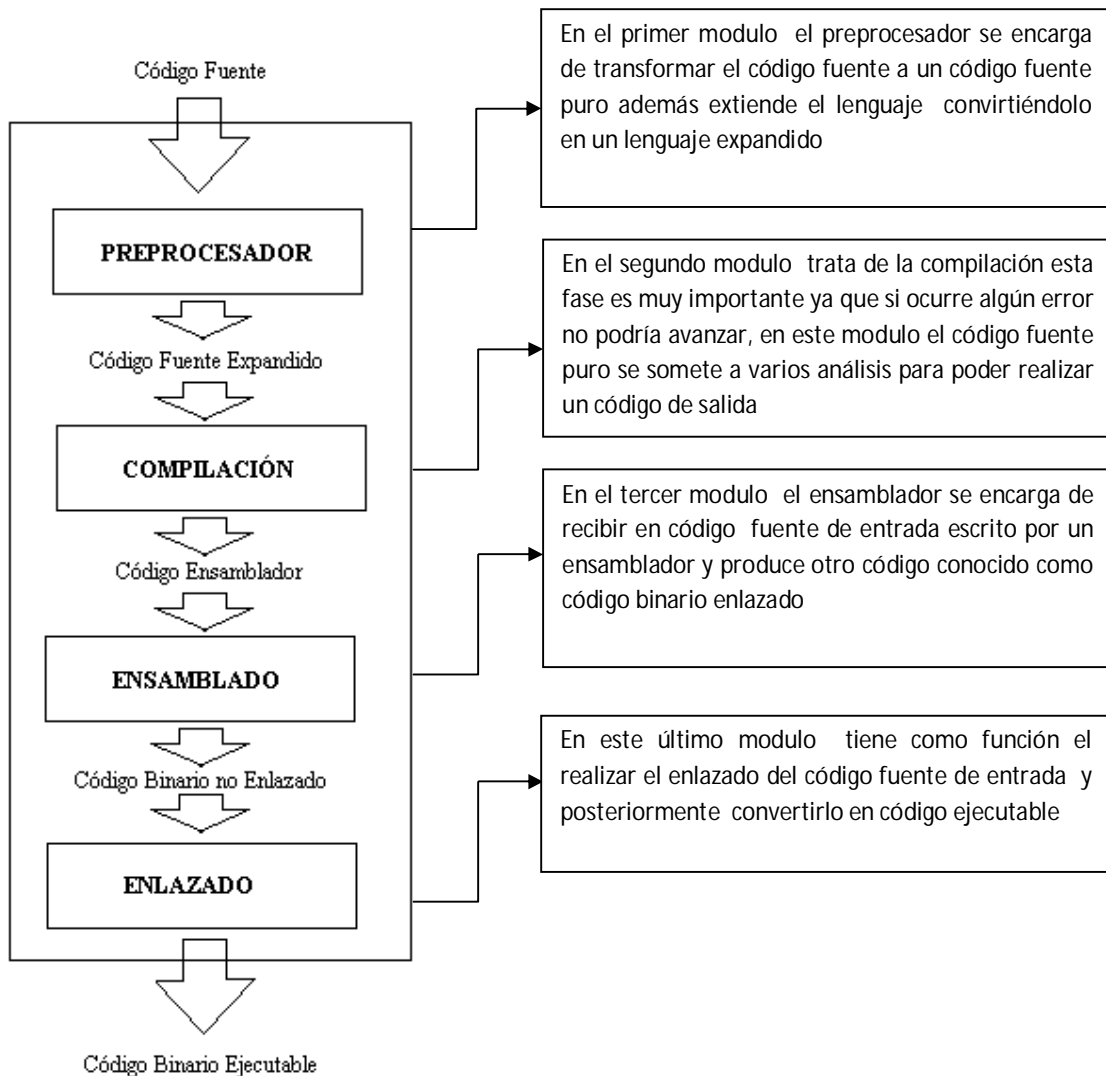


Fig.2 estructura del Compilador

Continuando con el trabajo pasamos a ver los distintos tipos de compiladores entre los cuales tenemos los siguientes:

Esta taxonomía de los tipos de compiladores no es excluyente, por lo que puede haber compiladores que se adscriban a varias categorías:

- **Compiladores cruzados:** Son aquellos que generan código para un sistema distinto del que están funcionando.
- **Compiladores optimizadores:** Estos compiladores tienen como función el cambiar el código para mejorar su eficiencia, pero manteniendo la funcionalidad del programa original.
- **Compiladores de una sola pasada:** Estos compiladores generan el código máquina a partir de una única lectura del código fuente.
- **Compiladores de varias pasadas:** Estos compiladores se diferencian de los otros compiladores ya que necesitan leer el código fuente varias veces antes de poder producir el código máquina.

**I**nterpretador. \_El interpretador es un programa el cual tiene una función muy importante ya que se encarga de analizar y ejecutar un programa que este desarrollado en cualquier lenguaje programación de alto nivel. Los interpretadores tienen ciertas diferencias con los compiladores ya que los interpretadores solo tienen la función de traducir el programas y no guardar cambios pero en cambio los compiladores tienen la función de traducir del lenguaje de alto nivel a lenguaje máquina, otras diferencias entre los interpretadores y los compiladores es que los interpretadores son más lentos que los compiladores, debido a una simple razón ya que los interpretadores tiene cumplir ciertas necesidades las cuales son que deben traducir las sentencias de cada programa al momento que se lo ejecuta pero a pesar de tener esta desventaja presenta ciertas ventajas una de ellas es que los interpretador son flexibles como entornos de programación y depuración y otra ventaja es que permite al programa interpretado no depender de la máquina donde se esta ejecutando la interpretación.

Comparando la actuación tanto la del compilador y el interpretador con la de un ser humano, un compilador equivale a un traductor profesional que, a partir de un texto, prepara otro independiente traducido a otra lengua, mientras que un intérprete corresponde al intérprete humano, que traduce de viva voz las palabras que oye, sin dejar constancia por escrito.

**T**raductor. Los traductores tienen como finalidad la de reducir o traducir los diferentes tipos de programas desarrollados en los diferentes tipos de lenguaje de programación de alto nivel a lenguaje máquina. Existen diferentes tipos de traductores para cada tipo de lenguaje de programación. En cada lenguaje de programación siempre existe un compromiso entre su poder de expresión y su dificultad de traducción. El poder de expresión en un lenguaje de programación se mide por las facilidades que ofrece para expresar algunas órdenes, mientras el lenguaje se aparece cada vez más al usado comúnmente por los humanos el poder de expresión del lenguaje dentro de la programación será mayor sin embargo de un lenguaje mayor será la dificultad para traducirlo al lenguaje de máquina.