

## Practica 2.- “Separación de Campos”

### 2.1.1 Sintaxis de las líneas de código en lenguaje ensamblador.

*OBJETIVO DEL SUBTEMA. Implementar un programa para separar las partes de una línea en lenguaje ensamblador previo análisis de la misma.*

#### **Introducción.**

El primer paso para la elaboración de un ensamblador es el reconocimiento de las partes que constituyen una línea de instrucción, las partes de la línea se determinan como Etiqueta, Instrucción, Operandos y Comentarios.

La línea de instrucción se fracciona en cuatro partes importantes:

1. Etiqueta
2. Instrucción
3. Operandos
4. Comentarios

**ETIQUETA:** Es una cadena alfanumérica que inicia con el carácter ‘\_’ o con un carácter alfabético y se debe escribir al inicio de la línea sin ningún espacio anterior.

**INSTRUCCIÓN:** Cadena de caracteres usualmente alfabéticos que son parte del conjunto de instrucciones del microcontrolador o microprocesador, o también pueden ser directivas del ensamblador. Serán las palabras reservadas que no se pueden usar como etiquetas, y estas siempre se escribirán en una segunda columna, es decir mínimo debe existir un espacio antes de la cadena, además las instrucciones se encuentran definidas dentro de la tabla de códigos de operación del micro.

**OPERANDOS:** Son los datos con los que trabajaran las instrucciones, estos pueden ser números, en distintas bases numéricas (decimal, octal, hexadecimal o binario; también podrían ser caracteres como códigos ASCII), registros internos del micro o incluso etiquetas, se encuentran en la tercera columna. No todas las instrucciones tienen operandos, existirán líneas en donde no exista operando, de igual forma habrá instrucciones que requieran de mas de un operando en cuyo caso estos se separan con ‘,’.

**COMENTARIOS:** Se pueden escribir en cualquier parte de la línea, se identifican por que inician con el carácter ‘;’.

Ejemplo: Si el usuario del sistema introduce la línea de instrucción determinada como:

INICIO LDAA #50,Y; Etiqueta <inicio>

El sistema deberá ser capaz de separar esta línea de la siguiente manera:

Etiqueta: INICIO  
Instrucción: LDAA  
Operando 1: #50  
Operando 2: Y

## **Desarrollo.**

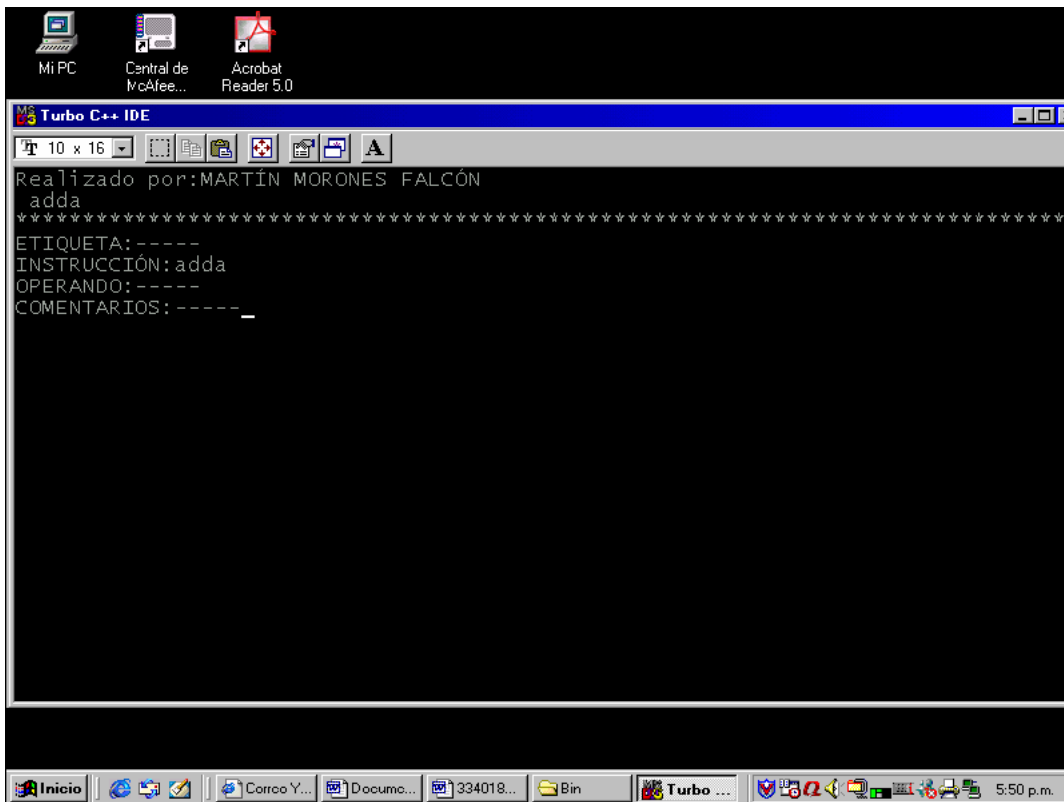
La practica desarrollada en el laboratorio de clase y también como trabajo para realizar fuera de laboratorio, implico la lectura de una cadena de caracteres, como primer paso, para después verificar la existencia de espacios en blanco generados por la barra espaciadora o por tabuladores, para así saber que caracteres pertenecen a cada una de las partes en que dividimos la Línea de Instrucción.

En lo particular el algoritmo creado se basa en la implementación de la función `< isspace >` de la librería `ctype.h`, la cual verifica los espacios en blanco, no importando si son tabuladores o una gran cantidad de espacios.

Otra de las especificaciones dadas para el desarrollo de esta practica fue el aislamiento de los operandos, que en la línea de instrucción se determinaban por comas que separaban a estos elementos, para el buen funcionamiento de esta sección del sistema se guardo la cadena de caracteres determinada como operandos para después condicionar la existencia de comas dando así la pauta de los diferentes operandos existentes. Los comentarios se distinguen por estar después de un punto y coma, lo cual facilita la búsqueda de estos, solo se verifico donde se encontraba un punto y como para después guardar en una cadena los caracteres que le precedían.

La practica consiste en separar una cadena en tokens a partir de encontrar un espacio, un tabulador ó una coma en el caso de los operandos.

En esta práctica para tener menos dificultad en la separación de token's se recomienda el uso de la función del lenguaje c `strtok` que ayuda mucho, pero no por si sola esta función ayuda tenemos que validar algunos otros aspectos como que pudiera solo aparecer una sola instrucción dentro de la línea y al sepárala aparezca de esta forma:

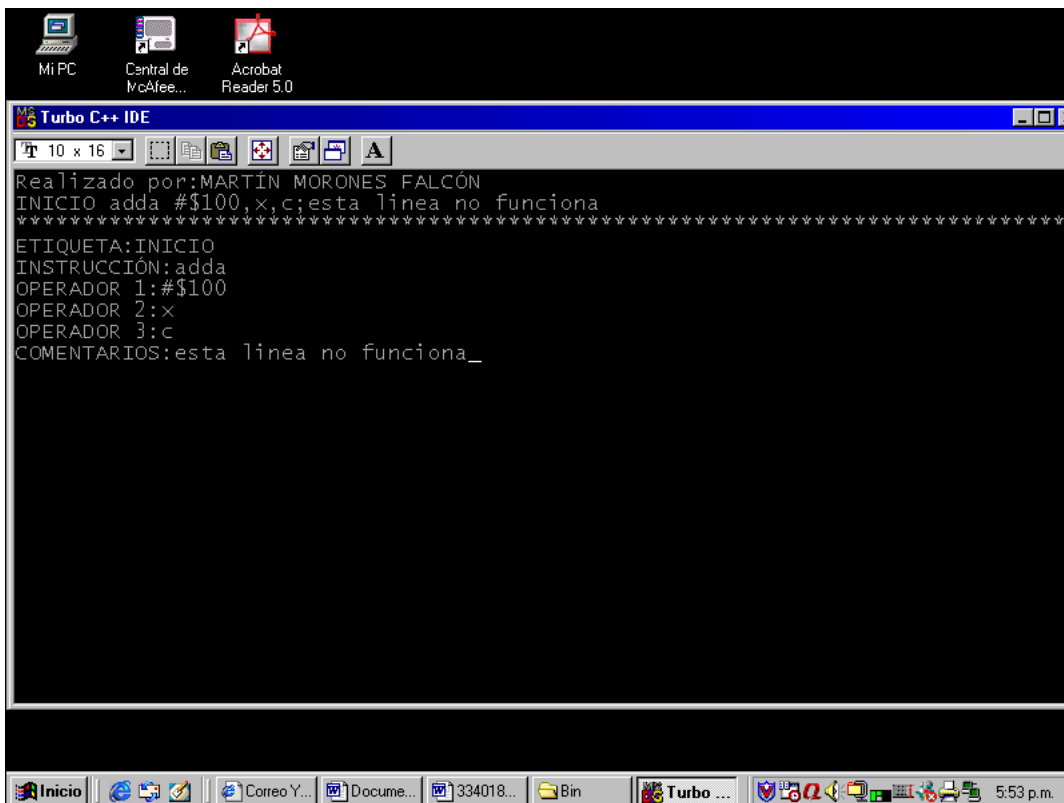


The screenshot shows the Turbo C++ IDE window. The title bar is 'Turbo C++ IDE'. The menu bar includes File, Edit, Compile, Run, and Help. The toolbar has icons for opening, saving, printing, and running. The main text area contains the following assembly code:

```
Realizado por:MARTÍN MORONES FALCÓN
adda
*****
ETIQUETA:-----
INSTRUCCIÓN:adda
OPERANDO:-----
COMENTARIOS:-----_
```

The taskbar at the bottom shows the 'Inicio' button and several open applications: Correo Y..., Docume..., 334018..., Bin, Turbo ..., and a system tray with various icons and the time 5:50 p.m.

Otra validación es separar todos lo operador por medio de una coma y que aparezca de esta forma:



The screenshot shows the Turbo C++ IDE window. The title bar is 'Turbo C++ IDE'. The menu bar includes File, Edit, Compile, Run, and Help. The toolbar has icons for opening, saving, printing, and running. The main text area contains the following assembly code:

```
Realizado por:MARTÍN MORONES FALCÓN
INICIO adda #$100,x,c;esta linea no funciona
*****
ETIQUETA:INICIO
INSTRUCCIÓN:adda
OPERADOR 1:#$100
OPERADOR 2:x
OPERADOR 3:c
COMENTARIOS:esta linea no funciona_
```

The taskbar at the bottom shows the 'Inicio' button and several open applications: Correo Y..., Docume..., 334018..., Bin, Turbo ..., and a system tray with various icons and the time 5:53 p.m.

En la pantalla anterior se ve la estructura que tenia que llevar una línea y como debe de aparecer ya separada.