Icono

Descripción generada automáticamenteUn dibujo de una persona

Descripción generada automáticamente con confianza bajaINSTITUTO POLITÉCNICO NACIONALESCUELA SUPERIOR DE CÓMPUTO

**PRACTICA 7**

“FUNCIONES & PROCEDIMIENTOS”

ALUMNO:

LOPEZ PEREZ ALBERTO ANDREI

PROFESOR:

TECLA PARRA ROBERTO

GRUPO:

5CM4

MATERIA:

COMPILADORES

## Introducción

En esta práctica se implementaron los conceptos base de compiladores y la implementación de YACC, donde se da continuidad a la práctica número 6, la cual fue una "Calculadora de Vectores con Ciclo For". Lo que nos atañe realizar en esta práctica es:

Agregar funciones y procedimientos

Donde, nuevamente ayudándonos de nuestra *máquina virtual de pila*, podemos insertar fácilmente estas nuevas funcionalidades. Lo primero que hay que entender es que para poder realizar funciones y procedimientos tenemos que ayudarnos de un elemento más en nuestra máquina el cual es la *pila de llamadas a función*, el cual nos permite anidar funciones y, sobre todo, regresar al flujo normal de nuestro programa, por lo que la implementación de esto es indispensable.

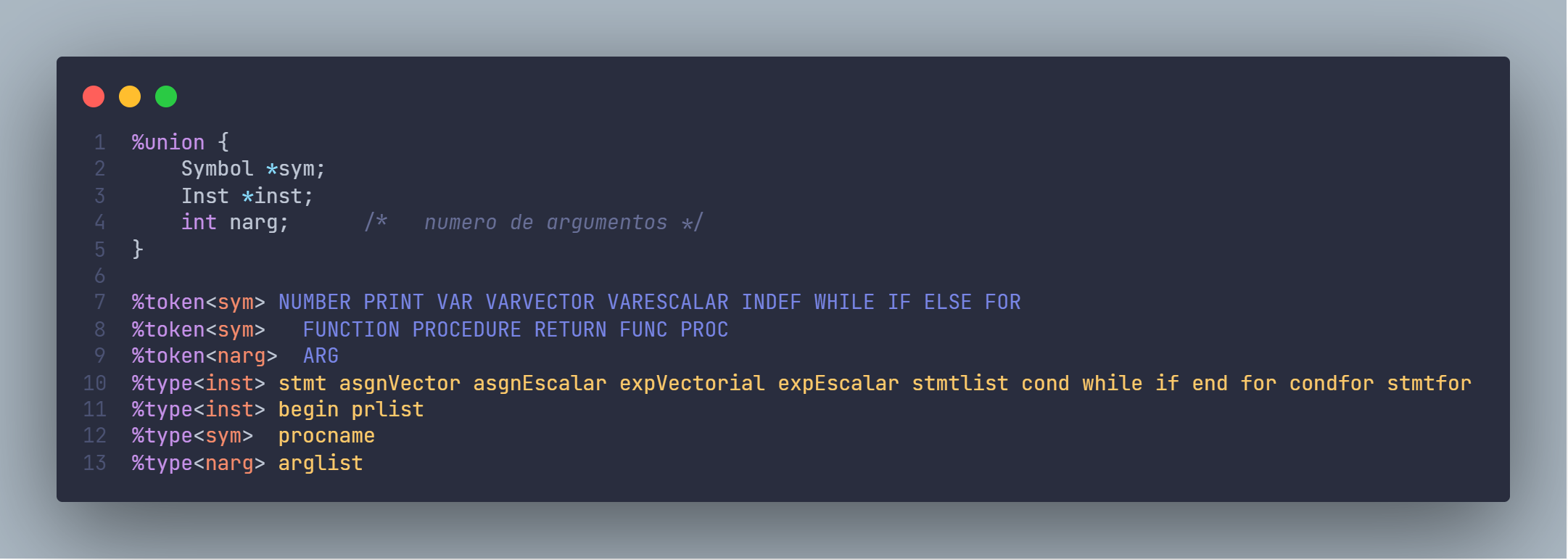
Ya la implementación de lo demás es un poco más trivial ya que las funciones se guardan en la misma RAM por lo que solo es cuestión de hacer las funciones para poder realizar llamadas para a partir de esa llamada obtener la dirección de memoria donde se encuentra alojada la primera instrucción de nuestra función y con ayuda de la pila regresar a nuestro contador en el flujo normal del programa.

Hablando de **funciones** y **procedimientos,** en nuestro contexto, tomaremos como una **función** a aquella que realiza un funcionamiento en su cuerpo y retorna un valor en específico, mientras que los **procedimientos** los tomaremos como simples instrucciones que se ejecutan sin regresar nada (algo así como funciones void).

Para poder realizar lo anterior se nos proporciona el código del **HOC6.y** visto en clase donde se implementa precisamente las **funciones** y **procedimientos**en una calculadora normal*,* por lo que con ayuda de ese nos apoyamos para construir nuestra propia implementación acomodada al contexto de la "Calculadora de Vectores".

## Desarrollo

Lo primero que realizamos fue agregar nuevos terminales y no terminales para las funciones y procedimientos en nuestras definiciones de YACC, además agregamos un nuevo tipo para los elementos de la pila de YACC en la unión, el cual nos va a servir para generar los argumentos y variables locales que se vayan a utilizar en las funciones y procedimientos.



Vista de las Definiciones de YACC donde se agregaron nuevos terminales y no terminales, además de un nuevo tipo en la unión.

Luego, como se puede ver en la siguiente imagen, se añadieron todas las producciones necesarias para generar las definiciones de las funciones y procedimientos, sus argumentos, algunas instrucciones que pueden ir en el cuerpo de las funciones (como el return) y las llamadas a función (ya sea una función o un procedimiento).



Nuevas producciones en la sección de Reglas de YACC

Hacemos hincapié en que todas las producciones están colocadas de forma lógica a como funcionan las funciones y procedimientos. Un ejemplo es que para llamar una función su producción está definida en **expVectorial** ya que esta regresa un valor vectorial, mientras que un procedimiento su producción se encuentra como un simple **stmt**.

ℹ Es importante mencionar que, por cuestiones de simpleza, las funciones regresan únicamente valores vectoriales. Igualmente los argumentos solo pueden llegar a ser vectoriales.

Todas las acciones gramaticales de todas las producciones obedecen al principio de ir generando código en la RAM e insertándolo, guardando ciertos valores importantes en algo que nosotros conocemos como **marco de función**, el cual posee varios datos importantes como lo que es la dirección de la primera instrucción de la función, a donde tiene que regresar cuando se termine de ejecutar, su número de argumentos, etc. Todo lo anterior se liga a un símbolo el cual por supuesto contiene el nombre de la función guardándose así en la tabla de símbolos.

Todo lo anterior es implementado en **code.c** donde además de definir lo que es un marco de función define algunas nuevas variables que ayudan a que la ejecución sea la adecuada.



Cambios en la Máquina Virtual para soportar la pila de funciones y preparando el lugar de los marcos de función.

Algunas de las funciones más importantes para realizar las funciones son las siguientes:



Funciones importantes para crear y ejecutar funciones

Las cuales obedecen el funcionamiento de las producciones y funciones apoyándose de la RAM para funcionar.

ℹ Es importante mencionar que las keywords para los nuevos terminales y la exportación de las nuevas funciones es necesaria llevarla acabo en sus respectivos archivos **init.c** y **symbol.h**

Como nota final, es importante mencionar que también se modificó el analizador léxico y el main de modo que permitiera el uso de argumentos y de las propias funciones y procedimientos.

### Compilación y Ejecución

Para poder compilar y ejecutar:

Compilación de YACC: "yacc -d calcVectores.y"

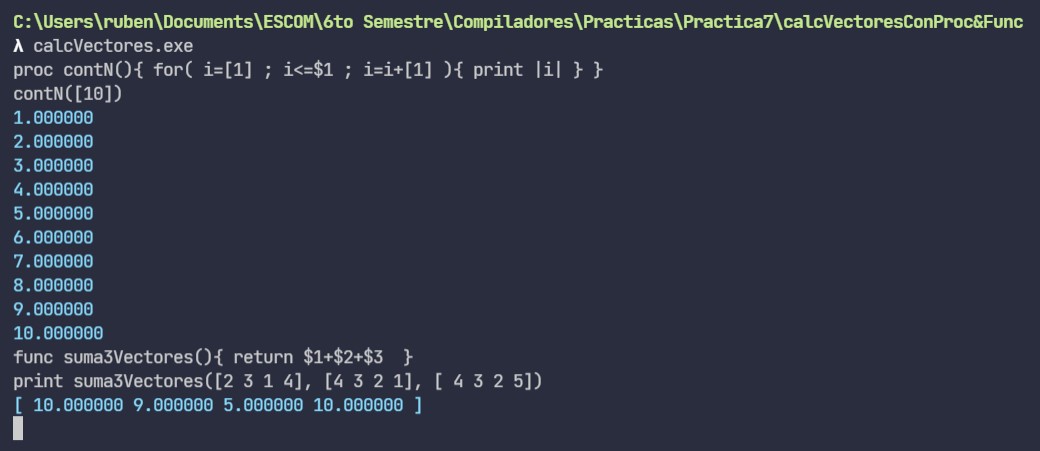
Compilación en programa C (WINDOWS): "gcc y.tab.c vector.c code.c symbol.c init.c -o calcVectores.exe -lm"

Compilación en programa C (LINUX): "gcc y.tab.c vector.c code.c symbol.c init.c -o calcVectores.out -lm"

Ejecución (WINDOWS): "calcVectores.exe"

Ejecución (LINUX): "./calcVectores.out"

### Programa en Ejecución



Ejemplo de la ejecución donde se crea un procedimiento para contar del 1 hasta la magnitud del vector mandado como argumento y donde se crea una función que suma 3 vectores pasados como argumentos.

## Conclusión

Esta práctica es la conclusión de una serie de calculadores que versión a versión iban incrementando en sus funcionalidades. Como bien hemos dicho en anteriores prácticas, el crecimiento de las calculadoras se vio verdaderamente incrementado a partir de la creación de la *máquina virtual de pila* ya que nos abrió las posibilidades de generar sentencias más complejas en funcionamiento y de algún modo, en estos momentos nos estamos acercando a una implementación realmente cercana a lo que es cualquier lenguaje de programación. Sin duda alguna, el añadir funciones y procedimientos mejorar potencialmente las ideas a implementar con la calculadora de vectores y en este caso, facilita al usuario el manejo de las instrucciones, por lo que podemos ver que la implementación de las 6 prácticas realmente se acomodaron como un guante a la "Calculadora de Vectores". Considero que la implementación de esto último no varía mucho en complejidad con respecto a por lo menos las 3 anteriores, pero repito nuevamente que todo depende en la comprensión total de los conceptos de la *máquina virtual de pila,* de YACC, compiladores, y de todos los conceptos que, con el paso de los años, hayamos adquirido al manejar computadoras y lenguajes de programación.

## Video Explicación & Ejecución

htt

ps://

[y](https://youtu.be/WxEXuTWVTdI)

outu.be/WxEXuTWVTd

[I](https://youtu.be/WxEXuTWVTdI)