**Portada**

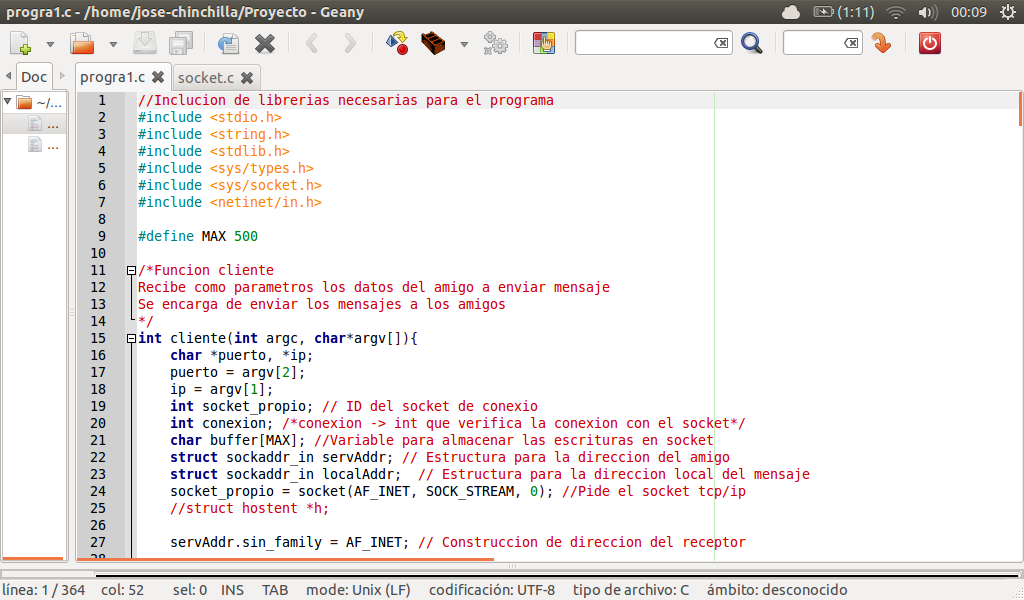
**Descripcion del problema**

Para la tarea programada #1 se debio realizar un progrma en el lenguje C, en si el problema de este programa consiste en buscar la forma de realizar un sistema de mensajeria instantanea entre dos computadoras, para realizarlo se utiliza como base la logica del sistema de mensajeria instantanea “WhatsApp”.

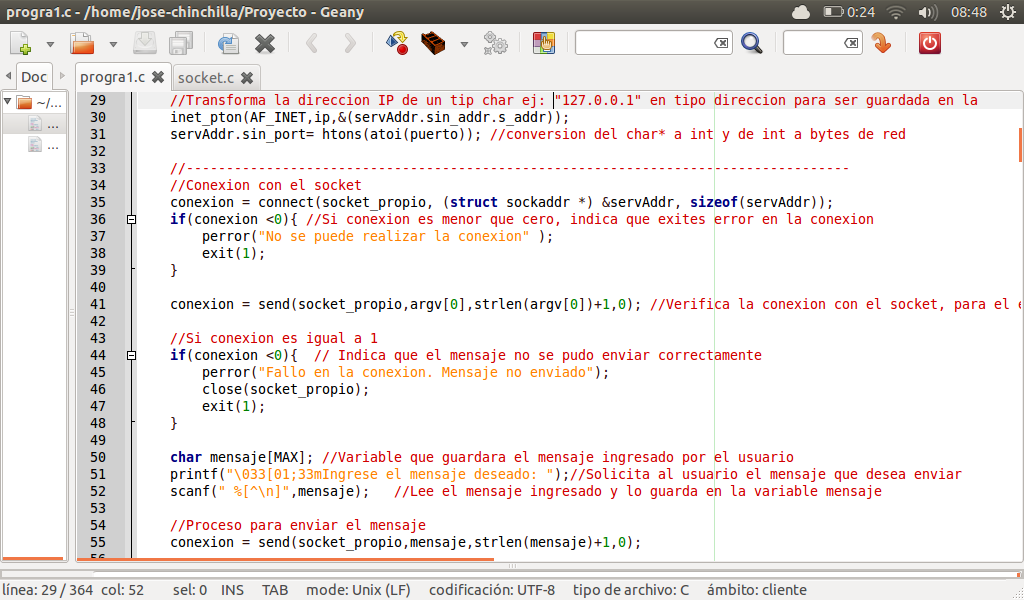
Algunos puntos importantes de esta tarea programada es que se debio aprender el manejo de sockets, el cual permite la interconexion entre las dos computadoras, otro aspesto es el manejo de la funcion fork(), la cual permite el manejo de dos procesos en el procesador a la misma vez, lo cual facilita el ingreso y salida de mensajes.

**Diseño del programa**

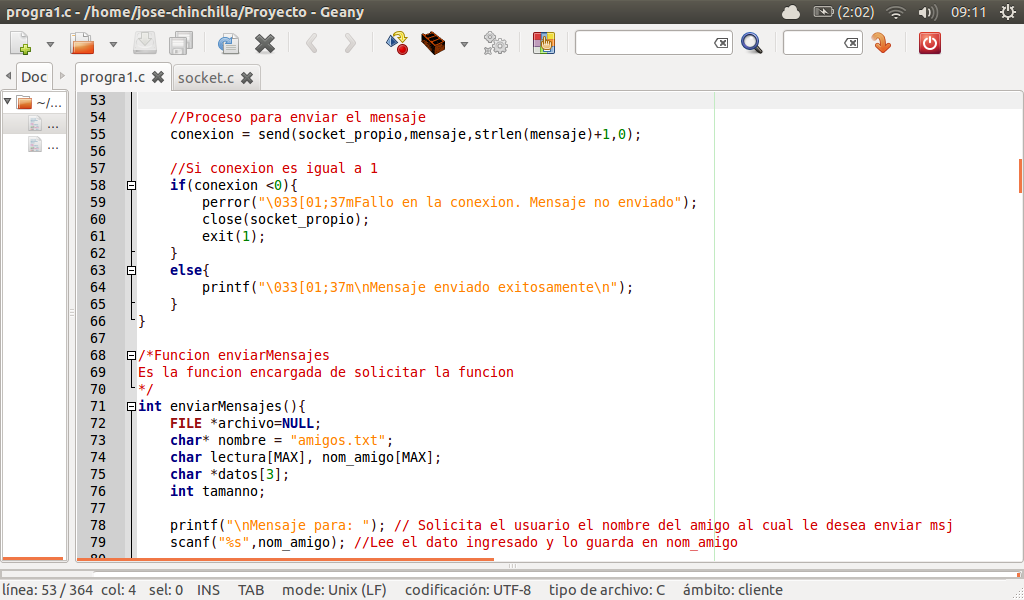
En esta parte se va a explicar el codigo utilizado para la realizaciòn del sistema de mensajeria instantanea.



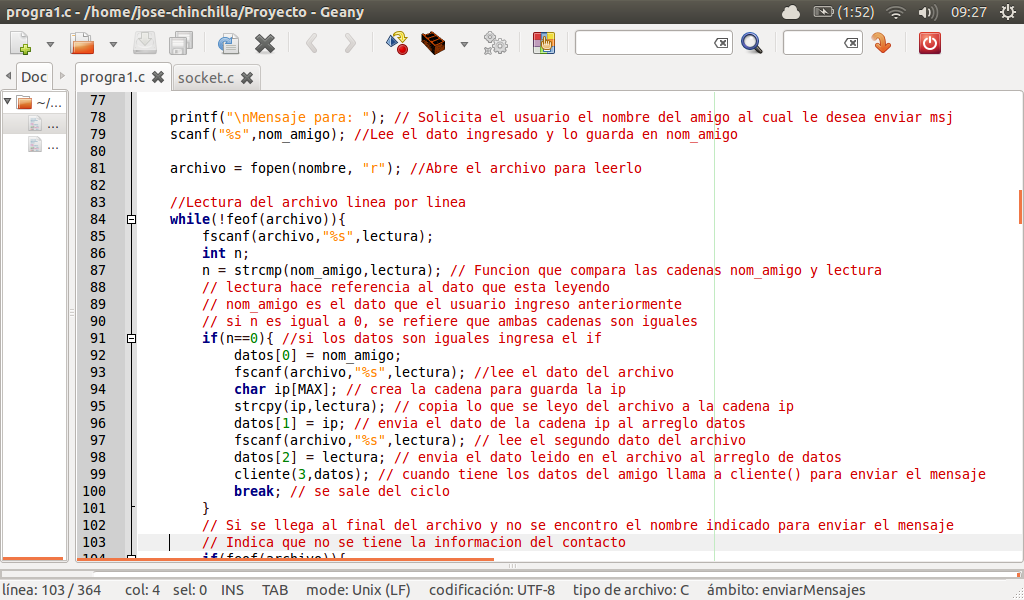
Como se muestra en la imagen anterior al principio se incluyen todas las librerias necesarias para el correcto funcionamiento de las funciones determinadas, como al incluir “sys/socket.h” esto permite el manejo y creacion de sockets en el codigo. Una vez determinada todas las librerias se crea la funcion cliente, la cual es de tipo int y recibe un argumento de tipo int y un arreglo de char\*, posteriormente se definen e inicializan una serie de valores que se van a utilizar en el resto de la funciòn. Esta funcion es la que permite la conexiòn del cliente con los sockets.



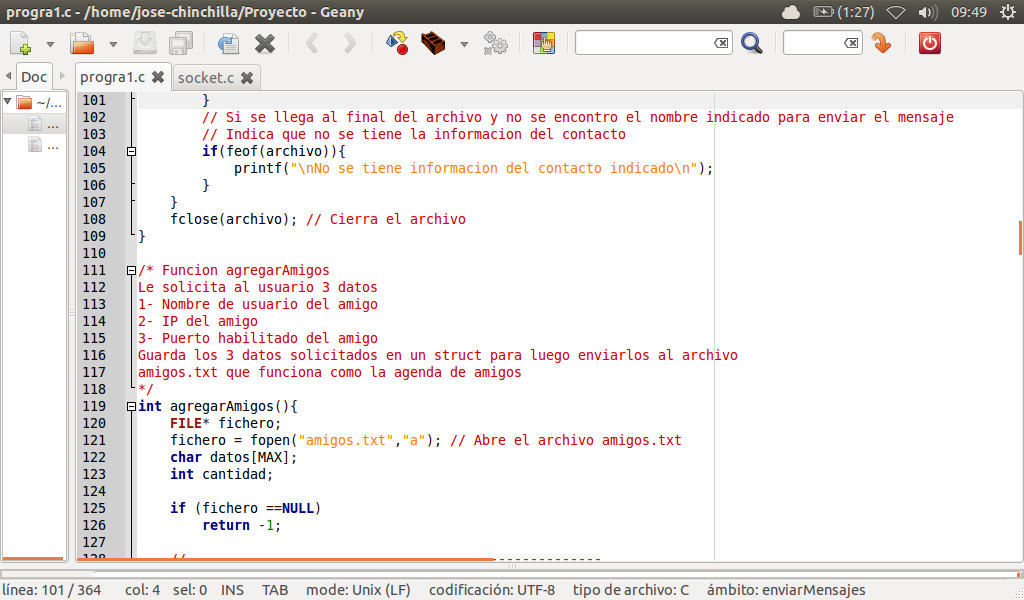
Es esta imagen se ve donde se trasforma la ip a char, esto para guardarla como parametro para posteriormente ser utilizado en los sockets. Despues comienza la conexión con el socket (todo esto a un se esta realizando en la funcion cliente), en esta parte se valida si la conexión fue exitosa, o si el mensaje fue enviado, para esto se utiliza la funcion send la cual es propia de sockets.



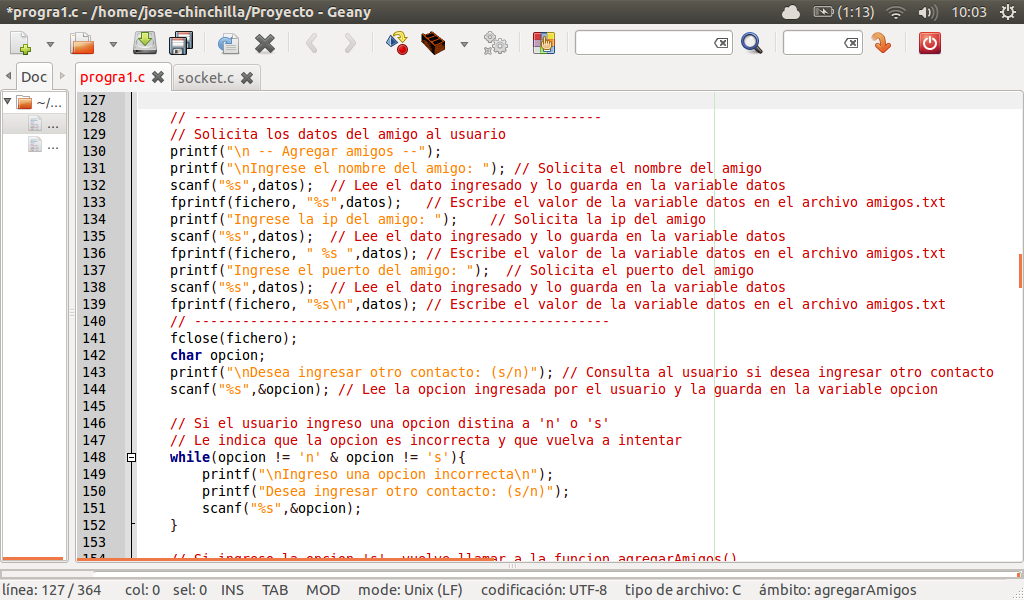
En esta parte se ve donde se termina de validar el envio de mensajes y ahi termina la funcion cliente(). Despues se comienza la funcion enviarMensajes esta funcion permite el envio de mensajes leyendo desde un archivo los contactos de los amigos del cliente. En esta imagen se ve por el momento donde se definen las variables a utilizar en esta funcion.



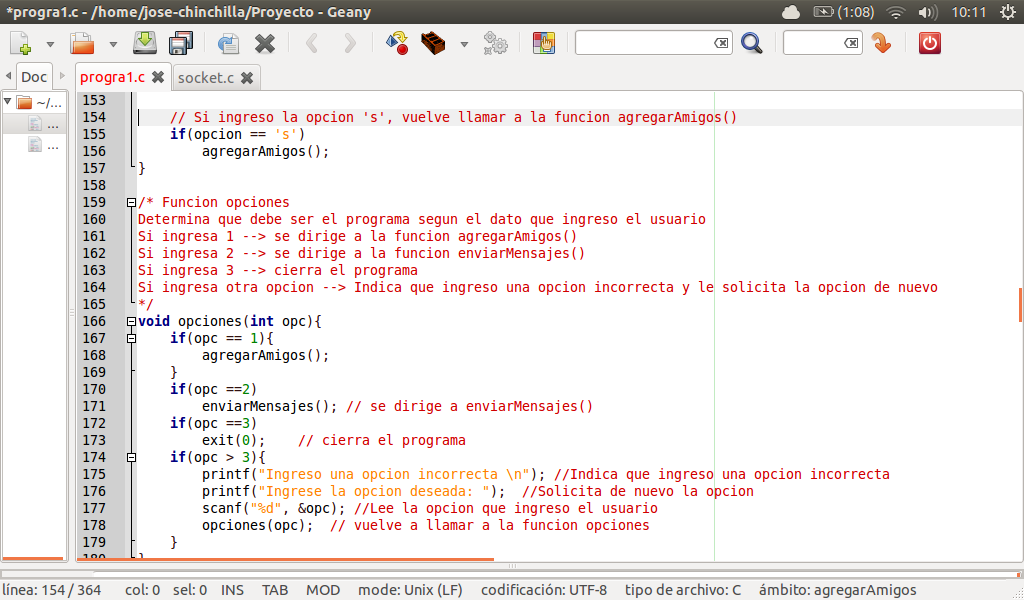
Aqui se ve donde se le pide al usuario que ingrese el nombre del amigo que quiere contactar, despues se abre el archivo de amigos, que es el que contiene los contactos, y con un while empieza a recorrer el archivo en busca del nombre anteriormente ingresado, si lo encuentra ingresa al if, ahi lee los datos, una vez con los datos llama la funcion cliente que ya fue especificada y se sale del while.



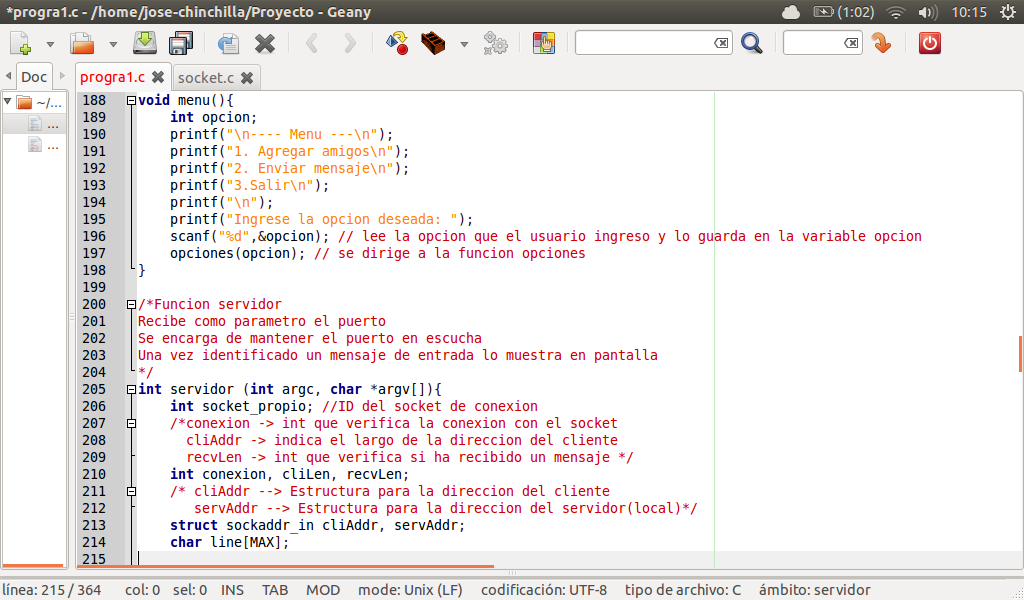
Aqui se indica que si se llego al final de la funcion enviarMensajes se imprime que no se encontro el cliente deseado, se cierra el archivo y ahi termina la funcion enviarMensajes. Despues se encuentra la funcion agregarAmigos, aquí se ve donde se abre el archivo amigos.txt que es donde se va a ingresar los datos del nuevo amigo.



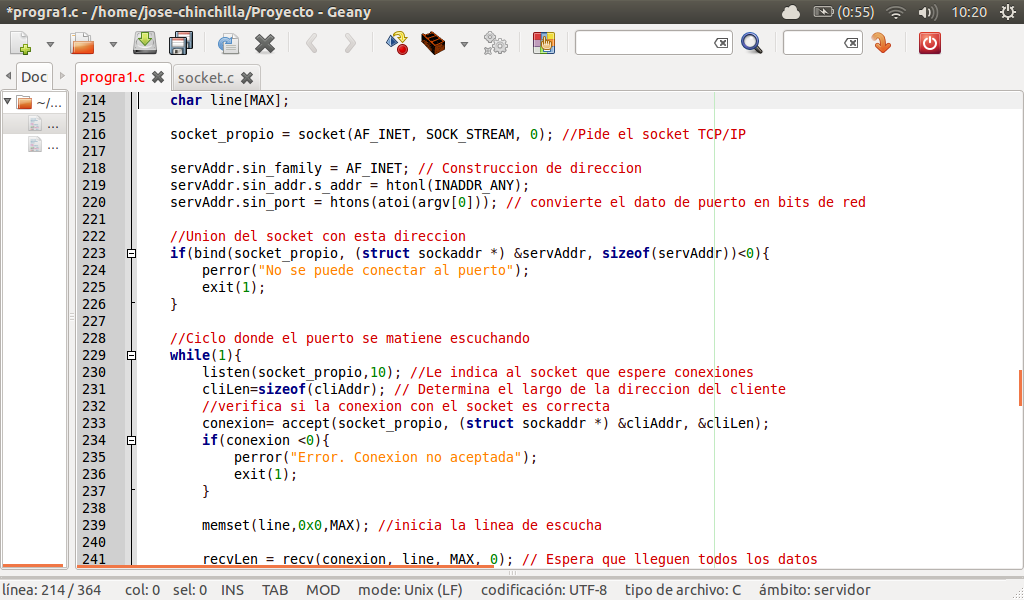
Esto es la continuacion de la funcion agregarAmigos, en esta parte lo que se hace es solicitar al cliente los datos necesarios del amigo, estos serian nombre, Ip y el puerto que utiliza su amigo. Despues viene la opciòn de si quiere volver a ingresar otro amigo, si se responde que si vuelve a comenzar la funcion agregarAmigos.



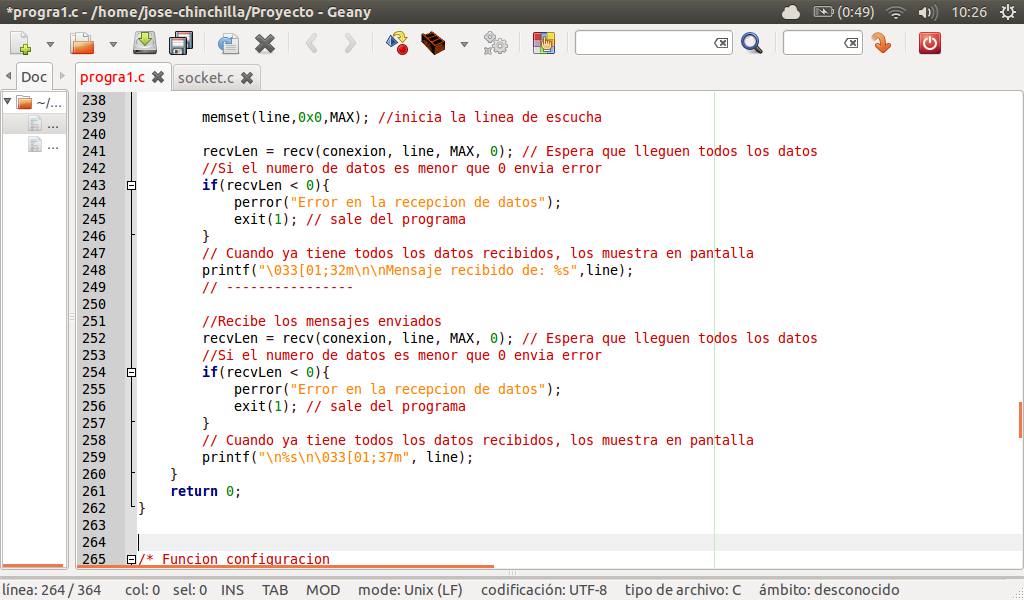
Aqui se ve donde comienza la funcion opciones, esta recibe un int, el cual es el que se va utilizar para llamar la opcion deseada por el usuario, ya sea agregar amigo, enviar mensajo o cerrar el programa.

[](http://c.learncodethehardway.org/book/ex2.html)

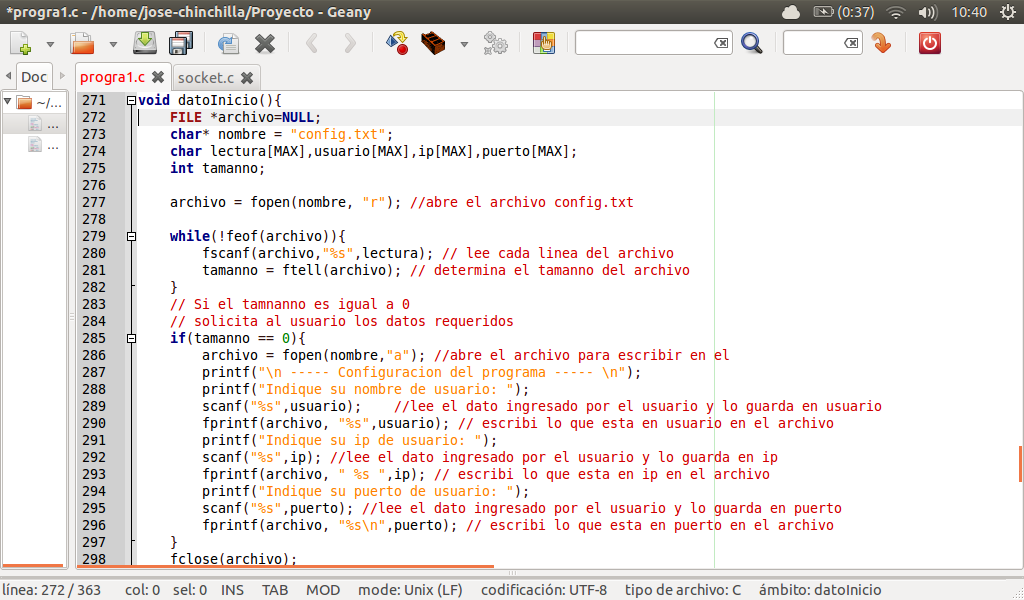
Aqui se muestra la funcion menu, esta lo que hace es imprimir en pantalla las funciones que se pueden hacer y despues solicita una opcion, la cual al ser ingresada se llama a la funcion opciones con este dato para realizar la funcion deseada. Posteriormente se comienza la funcion servidor, esta funcion lo que hace es mantener el puesto de escucha abuerto y cuando llega un mensaje lo muestra en pantalla.



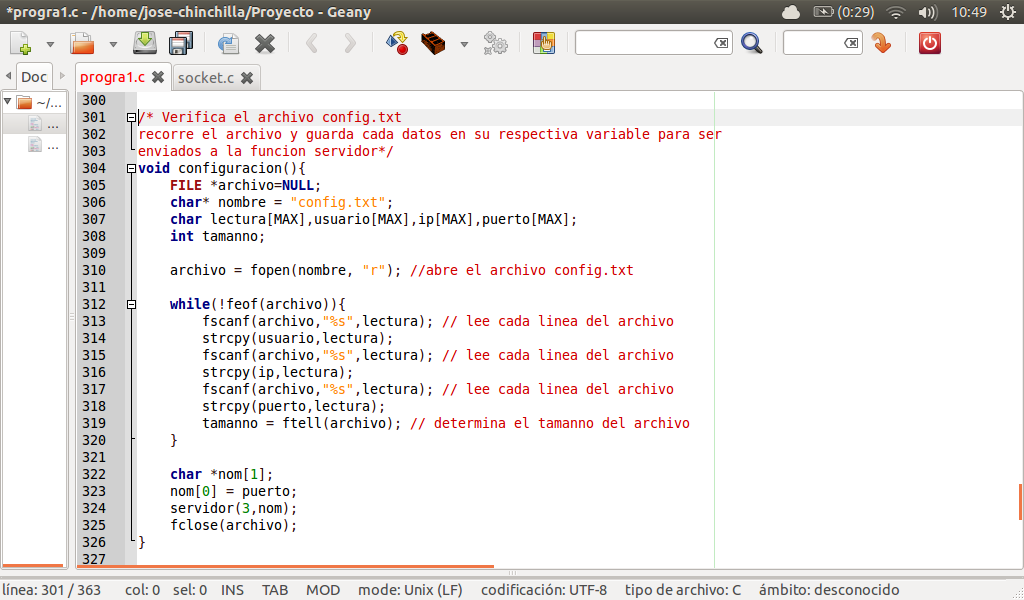
Aqui lo que se ve es donde se define el socket propio del usuario, agregandole al direccion del servidor la direccion ip y el puerto, despues estas dos se unen con la funcion bind(). Luego se crea un ciclo que es el que permite tener el puerto escuchando por la llegada de mensajes.



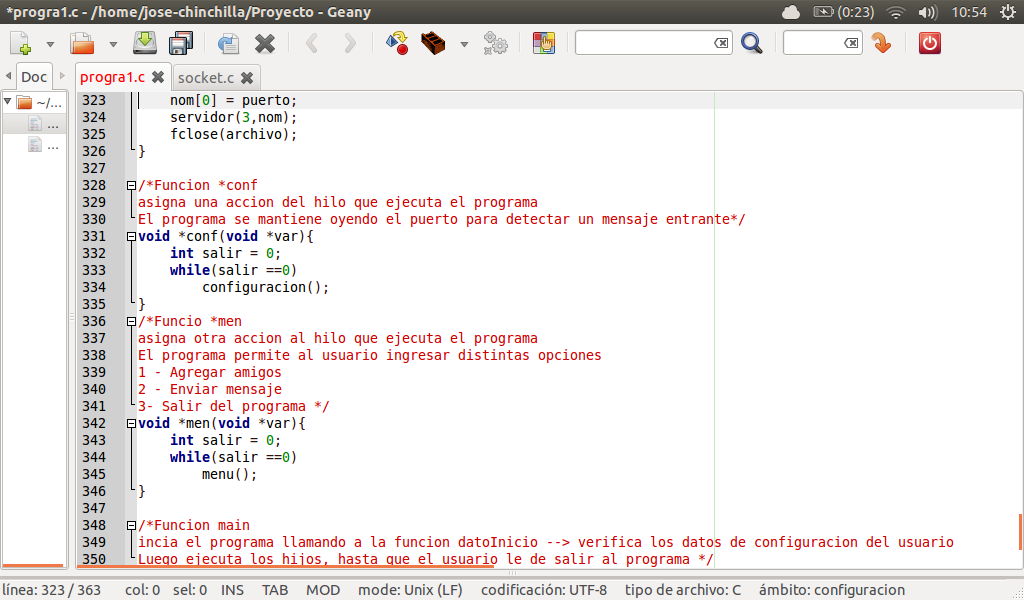
Aquí se muestra como se inicia la linea de escucha, con recv() espera la llegada de mensajes entrantes, si vienen todos los datos imprime el mensaje nuevo, sino tira error en la recepcion de datos.



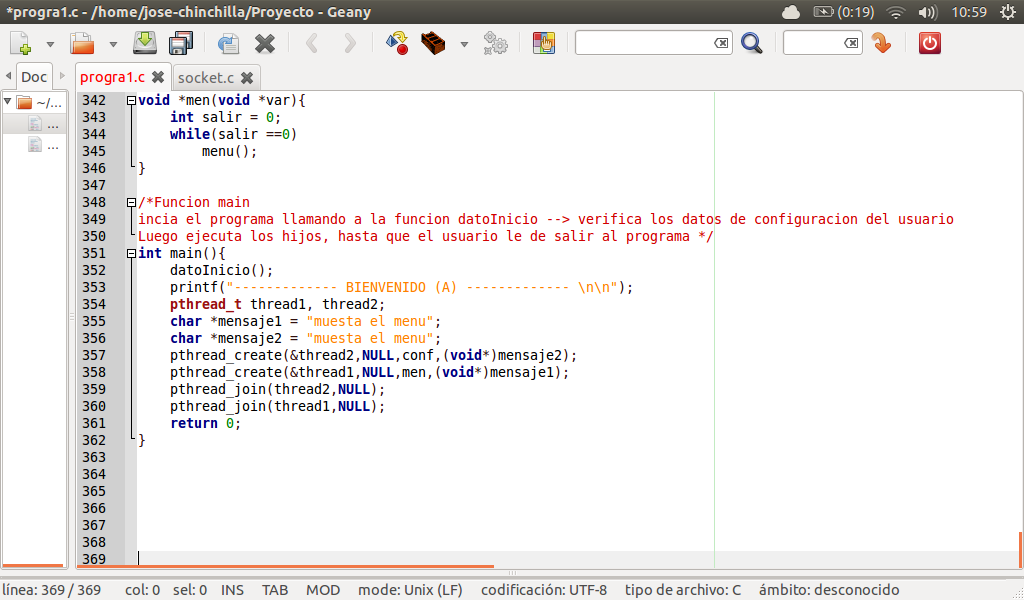
Esta funcion datoInicio() lo que hace es verificar si el archivo config.txt , con un while revisa si el archivo esta varcio de ser asi le solicita al usuaro, su nombre de usuario, la direccion Ip y el puerto a utilizar, y estos datos se ingresan en el archivo.



Aqui se muestra la funcion configuracion, esta lo que hace es con un while volver a leer el archivo config.txt y asigna los datos a variables establecidas, para despuès pasarlas como parametros a la funcionservidor().

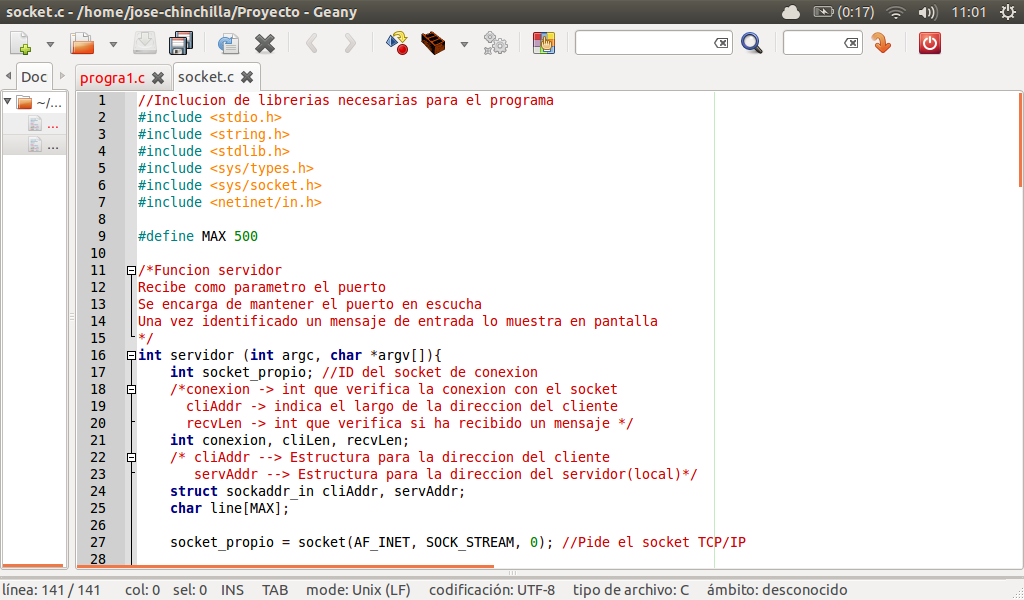


Es esta imagen se muestran dos funciones \*conf() esta asigna una acciòn al hilo de ejecutar el programa, lo hace llamando a la funcion configuraciòn. La otra funcion que se muestra es \*men esta asigna otra acciòn lo que hace que se realize diferentes opciones, esto se hace llamando la funcion menu().

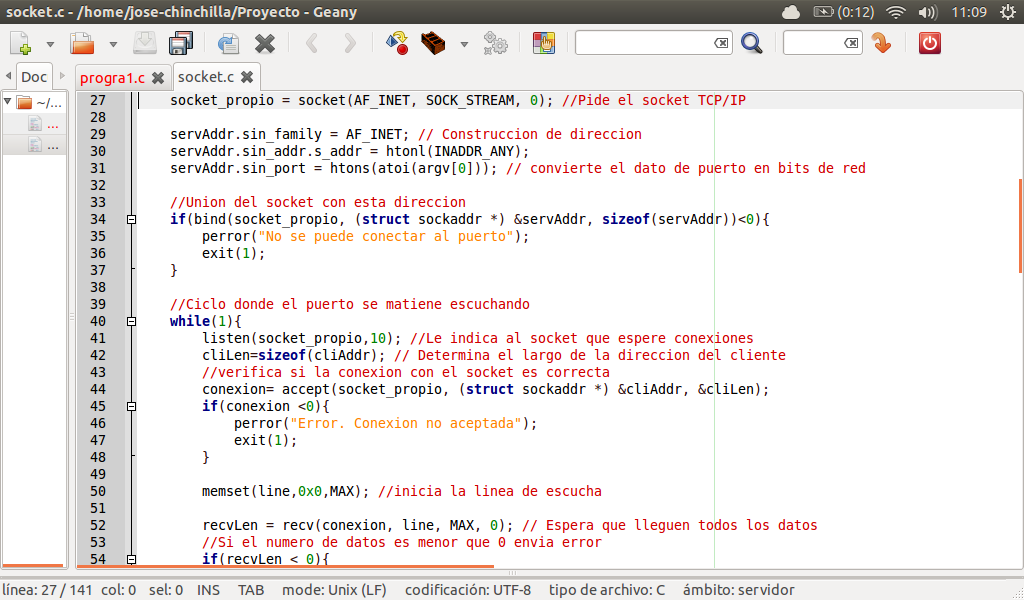


En esta imagen se muestra el main lo que permite el inicio del programas.

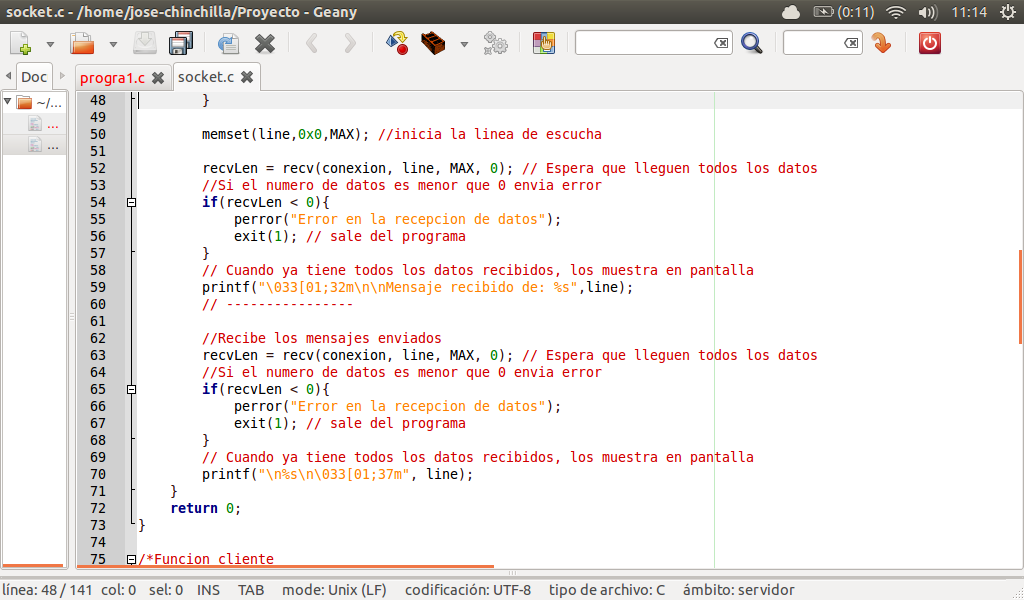
Acontinuacion se mostrara el codigo del archivo socket.c



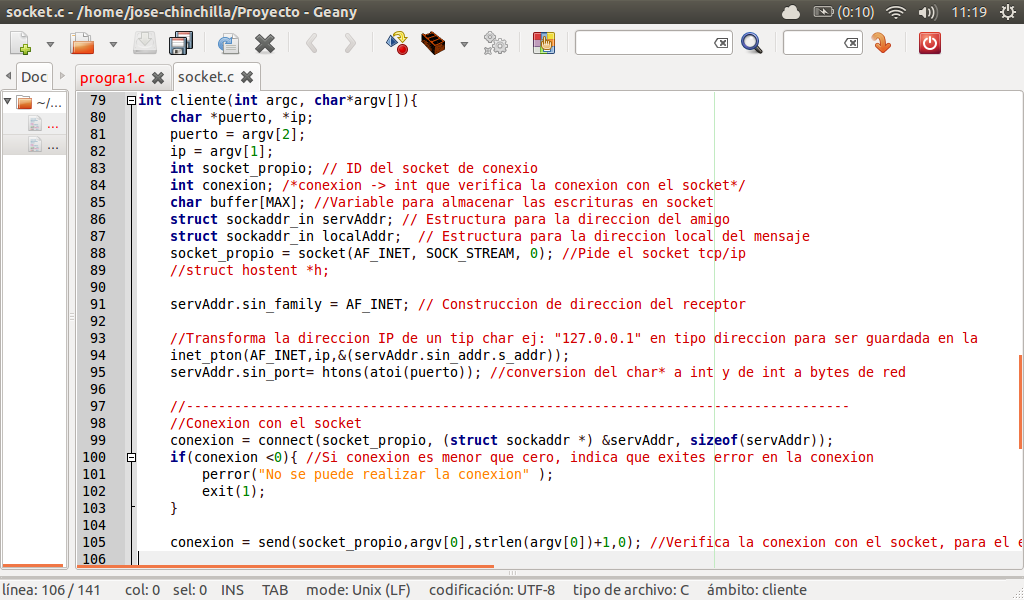
Al igual que en el archivo progra1.c se incluyen las librerias necesarias para le programa. Despuès se inicia la funcion servidor(), esta funcion recibe el puerto y esta funcion es la encargada de mantener el puerto en escucha .



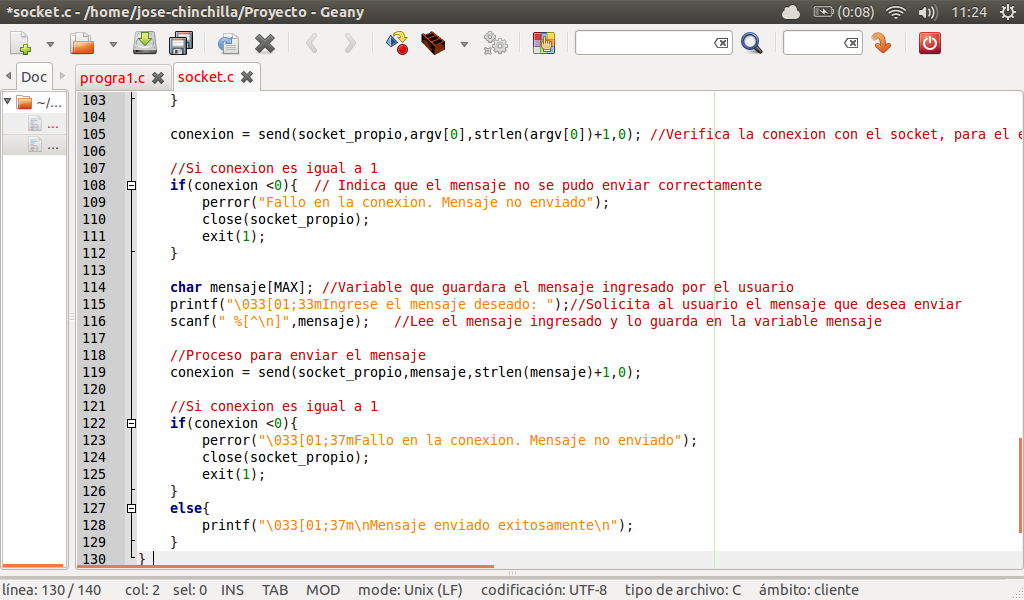
Aqui se muestra donde se construye la direccion ip, combinando la ip y el puerto en la funcion bind() y con el ciclo se mantiene el puerto de escucha en escucha si la conexión con el socket es correcta.



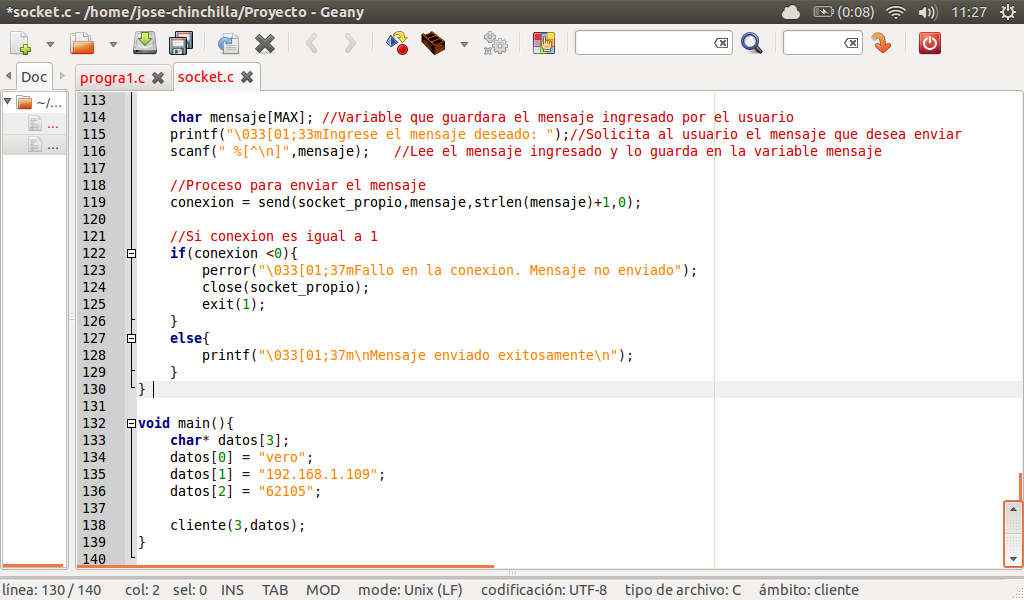
Aquí se inicia el puerto de escucha, lo que hace es ver si hay un mensaje de entrada, si ciene con todos los datos se immprime en pantalla sino tira error.



Aqui se muestra la funcion cliente la cual permite el envio de mensajes, se comienza con la construccion de la direccion del cliente receptor y la conexión con el socket



Aqui se muestra la validacion para enviar mensajes si se puede conectar con el socket lo envia con la funcion send(), sino muestra mensaje de error.

Por ultimo se muestra el main de los sockets.

**Librerías utilizadas**

**stdio.h:** Librería que nos permite definir tres tipos de variables, variables macros y diversas funciones para la realización de entradas y salidas datos de nuestro programa. Con esta librería se pueden declarar variables como:

* size\_t: Variable que nos brinda el mismo resultado que la palabra reservada sizeof
* printf: Función que nos permite realizar impresiones en consola
* scanf: Función que permite leer entradas de texto al programa principalmente de la entrada estándar de C (teclado)

**string.h:** Librería que permite definir macros, funciones para realizar la manipulación de un conjunto de caracteres. Con esta librería se pueden utilizar funciones como:

* strcmp: Función que permite realizar la comparación entre dos cadenas de caracteres
* strlen: Función que da como resultado el tamaño de una cadena de caracteres

**stdlib.h:** Librería que permite definir funciones generales, como las siguientes:

* atof: Función que convierte un arreglo de caracteres (char\*) a un punto flotante (float)
* atoi: Función que convierte un arreglo de caracteres (char\*) a un entero (int)
* abs: Función que realiza el valor absoluto a un valor numérico
* malloc: [Función que devuelve un puntero a un espacio de memoria que se encuentra vacío](http://www.tutorialspoint.com/c_standard_library/)

**sys/types.h:** Librería que permite definir símbolos y estructuras para utilizar sockets. Se pueden definir estructuras como:

* ip\_addr\_t: Función para la dirección IP
* u\_long: Tipo de dato long sin signo
* in\_addr\_t: Función para direcciones de internet

**sys/socket.h:** Librería que contiene la definición y estructura de un socket.

* Struct sockaddr\_in

**netinet/in.h:** Librería que contiene las definiciones de toda la familia de protocolos de Internet.

**Lecciones aprendidas**

Josue

Con la realización de esta tarea programada aprendí como se le puede dar un gran uso a las conexiones de internet, como mediante una simple dirección IP que todas las computadoras y redes poseen se puede dar el traslado de caracteres y de hasta archivos de una computadora a otra mediante una conexión TCD-IP.

Por otro lado se hizo una gran investigación sobre el manejo de archivos txt en el lenguaje de programación C, de la cual se aprendió que en lenguaje cuenta con una librería en la cual están previamente definidas las funciones que nos permiten y nos ayudan a abrir, leer y modificar archivos txt, como por ejemplo las funciones que utilizamos en nuestro programa como: fopen, fscanf, ftell, entre otra que no se utilizaron pero que fueron aprendidas para ser utilizadas en otras ocaciones.

Concluyendo con una tarea programada que nos llevo tiempo en realizarla por la dificultad de los sockets y mas en un lenguaje de programación levemente desconocido, pero dando como resultado un gran aprendizaje tanto en los dos segmentos en que se dividió la tarea (manejo de archivos y sockets), como en el aprendizaje del lenguaje de programación C.

Jose Daniel

Al realizar esta tarea programada puede aprender mas a fondo sobre el manejo del lenguaje C y sobre el programa git, siendo manejado ya fuera desde consola o desde la pagina web de github. Sin embargo esta tarea programada se baso màs que todo en el aprender el manejo de sockets y de la funcion fork() entonces estas fueron de las cosas que mas aprendi.   
Sin embargo, esta fue una progra que fomento el trabajo en equipo, y esta es una de las cosas que forman parte de las lecciones aprendidas para esta tarea programada.

Bibliografía

IBM (2010). Z/OS V1R11 information center (April 2010) Recuperado de <http://publib.boulder.ibm.com/infocenter/zos/v1r11/index.jsp?topic=/com.ibm.zos.r11.bpxbd00/typesh.htm>

The C Standard Library (s. f.). Recuperado de http://www.tutorialspoint.com/c\_standard\_library/