TRABAJO PRÁCTICO 1

75.41 Algoritmos y Programación II

Cátedra Rosa Wachenchauzer

Cuatrimestre II, 2016

Sinisi Fernando - Padrón: 99139

Corrector – Miguel Alfaro

Dificultades

Luego de leer el enunciado del trabajo práctico y analizar lo que se pide realizar, llegue a los siguiente problemas:

Leer archivos pasados por linea de comandos y validar tanto que la cantidad se la correcta y que también puedan abrirse y leer.

Encontrar una buena función que leyera linea por linea los archivos.

Entender como funciona la calculadora con notación polaca inversa y hallar la mejor manera de implementarla.

Implementar una buena lógica para el manejo de cadena.

Realizar la biblioteca strutil sin utilizar TDA.

Resolución

Paste:

Para esta aplicación al recibir en la función principal, por linea de comando los parámetros, los valido viendo su cantidad, y si corresponden a archivos que se puedan abrir en modo lectura.
En caso que no pase esa prueba se devuelve un código de error y muestra un mensaje por salida estándar al usuario.
Si no hubiera fallado, mediante un ciclo indefinido, leo de a una linea a la vez de cada archivo mediante la función getline que me hace una copia de la linea y la guarda en memoria dinámica. Esta función lo bueno que tiene es que me devuelve el largo de la cadena y la copia de la misma en memoria dinámica (después de usarla al final de todo debo liberarla).
Luego mediante salida estándar solo se imprime la linea leída mediante getline del primer archivo, se coloca un carácter de tabulación, se imprime la linea correspondiente del segundo archivo y se vuelve a realizar el mismo ciclo.
Si llegara a fallar en la lectura de alguna linea y/o llegara al final algún archivo terminaría la aplicación.

Si tenían igual cantidad de lineas termina correctamente y sino algún archivo llego al final antes y muestra mensaje al usuario por salida estándar.

More:

En esta aplicación utilice básicamente la misma lógica y funciones que en el paste ya que realiza la validación de los parámetros de la linea de comandos y utiliza getline para leer la linea correspondiente.
En caso que sean validos y se pudiera leer el archivo realiza un ciclo definido es decir que se repita la cantidad de veces que el usuario mando por parámetro y dentro del mismo se realizaba la impresión por pantalla de la linea leída mediante getline.
Si el archivo tiene menos lineas que las requeridas por el usuario, se muestran todas y se le dice que termino el archivo.
En caso contrario el programa entra en un ciclo indefinido que solo se corta si el usuario lo decide ingresando el carácter asociado a la terminación, o si sigue se llega al final del archivo.
Esto se debe que el usuario dentro del ciclo puede elegir ingresando enter, leer otra linea o ingresando * para salir.

Tanto more como paste son de tiempo constante ya que dependen de la cantidad de lineas del archivo y en el peor caso el getline tendría que leer todas las lineas del archivo.

Dc:

En esta aplicación aumento la dificultad y cambio la lógica ya que no se reciben parámetros por linea de comandos sino por entrada estándar.
En este caso decidí implementar el algoritmo usando las estructura auxiliar pila además de las siguientes funciones:

Función split (definida en la biblioteca strutil.h) que me devuelve un arreglo dinámico donde cada posición corresponde a una subcadena de una cadena que fueron separadas mediante un separador definido.

Función strtod que me devuelve un numero del tipo double si alguna parte de la cadena eran dígitos y me devuelve un puntero a donde dejo de convertirse(si pudo convertirla completa me devuelve un carácter vacío).

Función strcmp: Para realizar la comparación de cadenas con operaciones a realizar o para validaciones.

Función getline: Para leer la linea que me dice la operación a realizar.

La pila era la mejor opción para este caso ya que la calculadora con notación polaca inversa realiza las ultimas dos operación y de ahí para atrás como si estuviera apilando los resultados de las ultimas operaciones y la pila funciona justamente haciendo que solo se pueda acceder a lo ultimo que ingreso en ella. Y el strutil lo utilice para facilitar el manejo de cadenas ya que las operaciones son una cadena de largo n con mezcla de espacios caracteres, números , etc.

El algoritmo consta de leer la cadena con la operación mediante getline, aplicar la función split para separar la cadena mediante el separador espacio así me separa los números de las operaciones. Luego mediante un ciclo indefinido es decir hasta el final del arreglo de subcadenas se fija, si es un numero se lo apila y si es operación se desapilan los últimos 2 números apilados, se realiza la operación indicada y se apila el resultado, volviendo a comenzar el ciclo con la próxima posición del arreglo.
Si la cadena tuviera caracteres inválidos o la operación ingresada no corresponde con la notación polaca inversa o otros casos en que falle, se libera toda la memoria dinámica y se devuelve un código de error y muestra un mensaje de operación invalida.
Si la operación era valida se muestra el resultado y termina en caso que por entrada estándar no hubiera otra operación, sino volvería a leer la otra cadena.

El orden de el dc es cuadrático ya que depende del largo de la operación y de la cantidad de operaciones a realizar es decir, si tengo n operaciones a realizar y cada una tiene largo n es decir muchos números y caracteres seria de orden $o(n^2)$.

Función Split (strutil):

Para realizar esta funcion primero cree un ciclo que recorra la cadena y aumente un contador que representa la cantidad de separadores que aparecen en la cadena. Con esto ya se cuanta memoria tengo que pedir ya que

el arreglo tendrá la cantidad de separadores mas la posición reservada para el NULL.
Luego ya con la cantidad contada y el malloc realizado recorro de nuevo la cadena y guardo los caracteres hasta que aparece un separador, en es momento uso la función strdup que me hace una copia de la cadena hasta ahí recorrida y la guarda en memoria dinámica. Y esa dirección la coloco en la posición correspondiente del arreglo dinámico que cree.
Luego continuo avanzando hasta los próximos separadores repitiendo la operación y al finalizar se devuelve el arreglo dinámico.
Si llegara a fallar en pedir memoria, libera la ya pedida y devuelve null.
EL orden es lineal ya que depende solo de el largo de la cadena a aplicarle el split es decir es o(n).

Función Join (strutil):

Es la operación inversa del split así que se espera que tarde también tiempo constante y así es ya que depende del largo de las subcadenas del arreglo y al juntarlas todas va a depender del largo de la cadena devuelta entonces es 0(n).
En este algoritmo al igual que el anterior primero recorro el arreglo para saber cuantas subcadenas hay, y luego recorro cada subcadena hasta llegar al final de la misma y la sumo o concateno a la anterior en el nuevo arreglo que pedí con la memoria suficiente para toda la cadena total.
Al terminar me quedo la cadena armada y solo debo agregarle el carácter de fin de linea y devolverla.

Función free_strv (strutil):

Esta funcion libera la memoria de un arreglo de cadena y basicamente es un ciclo que recorre todas las posiciones del arreglo haciendoles free y luego al terminar libera la memoria del arreglo.

Es de tiempo constante ya que depende de la cantidad de cadenas que contenga el arreglo y para liberarla toda necesito recorrerlo todo.