Informe Trabajo Practico N1

Algoritmos y Programación II

(75.41)

Segundo Cuatrimestre 2016

Alejo Iván Acevedo

Corrector: Joaquín Torré Zaffaroni

**Dificultades a afrontar:**

Para la realización del TP1 se ha solicitado implementar 3 utilidades las cuales son: calculadora polaca inversa (dc), more y paste. Como así también implementar 3 funciones del archivo strutil.h: split, join y free\_strv.

Cada una de las implementaciones tiene sus dificultades las cuales pasare a enumerar a continuación.

* Calculadora polaca inversa:

El funcionamiento de la misma es bastante intrincada ya que el mismo es poco intuitivo.

Por ejemplo: 4 2 3 + \* es 4\*(3+2) es decir se hace la operación del primer operador entre los últimos dos números, se guarda y se hace la operación del siguiente operador entre el resultado y el siguiente numero. Esto se hace sucesivamente dependiendo de la cantidad de operadores que haya.

* More:

Esta implementación tiene las dificultades de leer archivos, validar que exista el archivo y que el numero pasado por parámetro sea entero positivo.

* Paste:

En esta implementación también se tienen que leer archivos, validar que sean exactamente dos parámetros lo que se reciben y se puede solo almacenar una sola linea en memoria.

* Strutil.h:

Por ultimo en esta implementacion no se pueden usar TDA y se debe tener mucho cuidado al manejar las cadenas ya que olvidando poner el carácter ‘\0’ que informa el fin de cadena podemos generar un overflow del programa.

**Soluciones y algoritmo:**

* Para afrontar las dificultades de la implementación de la calculadora polaca inversa se ha decidido en principio utilizar un TDA cola para almacenar los operadores ingresados por entrada estándar y un TDA pila para almacenar los números ingresados por entrada estándar. Para recibir la entrada estándar se ha utilizado la función getline() de la librería stdio.h. Por ultimo el funcionamiento es el siguiente: el programa recibe la cadena ingresada por entrada estándar, luego apila todos los números y luego encola todo los signos, una vez que finalizo con la cadena libera su memoria, finalmente desapila dos numero y desencola un signo, realiza la operación y apila el resultado, hace esta operación hasta que no encuentre mas signos y envía el resultado por salida estándar.
* En la implementación de la función more se reciben el nombre del archivo y la cantidad de lineas que el usuario quiere leer del mismo. Se constata que este numero sea positivo y que el archivo exista y se usa un bucle for para que se impriman por salida estándar tantas lineas como fueron pedidas, luego se ingresa a un bucle while que constata que el archivo no halla llegado a su fin y se imprimen tantas lineas como el vez que el usuario apreté enter.
* Para la implementación de la función paste, se reciben dos nombres de archivos, se constatan que existan y se imprimen tantos números de lineas como los mismos compartan, es decir, si el primero tiene 3 lineas y el segundo 5, se imprimen solo 3 lineas de cada uno. Las lineas del primer archivos se encuentran a la izquierda de un tab y las del segundo a la derecha del mismo. Estas lineas son impresas por salida estándar.
* Cuando se implemento las funciones del strutil se uso siempre memoria dinámica. Para la función free\_strv, se utiliza un bucle while que recorre el arreglo hasta encontrar un NULL, cada vez que recorre este arreglo libera el espacio de memoria que utiliza cada cadena y cuando sale del mismo libera el total de la memoria del arreglo. En la función split se utiliza el bucle for para recorrer la totalidad de la cadena pasada por parámetro buscando el separador también pasado por parámetro, cada vez que encuentra el mismo guarda la comienza a guardar la cadena siguiente en una nueva componente del array, finalmente se devuelve el array con cada nueva cadena. Finalmente en la función join, se recorre el array pasado por parámetro al igual que en la función antes mencionada free\_strv, y se va concatenando cada una de las componentes del mismo siendo separado por un carácter que pasa el usuario por parámetro.

**Orden de cada función:**

* Calculadora polaca inversa: El orden de esta función es O(n), ya que se recorre solo una vez la cadena ingresada por entrada estándar.
* Paste: Esta función tiene un orden O(n), ya que imprime por salida una cantidad de veces igual a la cantidad de lineas del archivo mas corto.
* Paste: En esta función el orden depende del usuario ya que no tiene una cantidad de lineas fijas del archivo para recorrerlas.
* free\_strv: El orden de esta funcion es O(n) ya que recorre el array pasado por parametro, solo un vez.
* Split: Esta funcion tiene un orden O(n), ya que recorre el total de la cadena pasada por parametro.
* Join: Esta funcion al igual que free\_strv recorre el array pasado por parametro por lo que su orden es O(n)