Objetivo

Esta tarea tiene como finalidad repasar los temas de punteros y asignación dinámica de memoria, así como resolución de algoritmos en general.

Instrucciones

Realice los siguientes ejercicios:

1) Escriba un programa en C++ que permita recibir una cantidad ilimitada de números. Los números se almacenarán en un array creado en el heap. Luego, imprima en pantalla los valores del array - 5 por línea - junto con el cálculo del promedio de esos números. El tamaño inicial del array será de 5, y si la cantidad de números digitados sobrepasa el tamaño del array, entonces deberá crear otro array con 5 espacios de memoria adicionales y copiar los valores del array viejo al nuevo. No olvide realizar las operaciones de limpieza de la memoria correctamente.

2) Escriba un programa en C++ que asigne memoria dinámica y la inicialice para dos string cualquiera (debe tener entonces dos apuntadores, uno a cada string) e imprímalos en pantalla. Luego, intercambie los apuntadores (el apuntador A al string A debe apuntar al string B, mientras que el apuntador B debe apuntar al string A). Imprima el valor de los apuntadores después del intercambio. No olvide realizar las operaciones de limpieza de la memoria correctamente.

3) Se le proporcionan dos archivos: Mensaje.h y Source.cpp. Incluya ambos archivos en un proyecto de Visual Studio y corra el programa. Puede notar que se generan dos situaciones no deseadas:

a) Se imprime la dirección de memoria en lugar de los mensajes

b) El programa termina con una excepción de memoria

Aplique técnicas de Debugging para encontrar los problemas y realice las correcciones necesarias en el código para que funcione adecuadamente. Analice por qué es necesario crear un constructor de copia para resolver la segunda situación.

4) Realice el juego "Adivine el número" para que el usuario juegue contra la computadora:

a) Se muestra un menú con tres opciones:

1) Jugador adivina el numero

2) CPU adivina el numero

3) Salir

b) La idea del juego es la siguiente: si el usuario selecciona la opción #1, entonces el CPU generará un número entre 1 y 100 de manera pseudoaleatoria. Entonces, el jugador tiene 5 intentos para encontrar el número que el CPU seleccionó. Por cada número que el jugador ingrese, se debe mostrar una pista en pantalla (si el número es mayor o menor que el número del CPU) hasta que el jugador encuentre el número o hasta que se agoten los intentos, en cuyo caso pierde.

Por otro lado, si la opción seleccionada es la #2, entonces el jugador debe proporcionar el número entre 1 y 100 y el CPU debe encontrarlo (con las mismas condiciones descritas en la parte b).

5) Escriba un programa llamado "wc" (word count). Esta versión es una versión reducida de la utilidad de la línea de comandos Linux **wc**. En esta versión, el programa recibe como entrada un archivo de texto, y la salida del programa muestra la cantidad de líneas, palabras y letras.

Por ejemplo, asuma el siguiente texto de entrada que está en un archivo:

En ciencias de la computación una estructura de datos es una forma particular de organizar datos en una computadora para que puedan
ser utilizados de manera eficiente. Diferentes tipos de estructuras de datos son adecuados para diferentes tipos de aplicaciones
y algunos son altamente especializados para tareas específicas
Las estructuras de datos son un medio para manejar grandes cantidades de datos de manera eficiente para usos tales como grandes bases
de datos y servicios de indización de Internet. Por lo general las estructuras de datos eficientes son clave para diseñar algoritmos
eficientes. Algunos métodos formales de diseño y lenguajes de programación destacan las estructuras de datos en lugar de los algoritmos
como el factor clave de organización en el diseño de software.

La salida del programa sería:

7 122 782

Puede utilizar el archivo adjunto - texto.txt - para realizar las pruebas.