Clase 3

Ejercicios de Archivos de texto:

Ejercicio 1: “entender el programa”

```
// directivas del preprocesador
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <ctype.h>
#include <math.h>
#define p printf
#define s scanf
// struct, prototipos
int main()
{
  FILE *archivo;
  char ch; // System("Color 1F")
  char *nombre="texto99.txt";
  if((archivo=fopen(nombre,"w"))==NULL)
  {
    p("\n\n***El archivo %s no pudo abrirse ***\n",nombre);
  }
  else
  {
    p("\n\nIngrese caracteres hasta[Enter]");
    while((ch=getchar())!='\n')
    {
      putc(ch,archivo);
      fclose(archivo);
    }
    //system ("Pause");
  }
  return 0;
}
```

💡 ***Explicación codeGPT:*** Este código crea un archivo de texto llamado "texto99.txt", y le permite al usuario escribir caracteres en él. Primero, se importan las bibliotecas necesarias, y se definen macros para las funciones printf y scanf. Luego, se define una variable puntero de carácter llamada "nombre" que almacena el nombre del archivo que se creará. A continuación, se verifica si el archivo se puede abrir en modo escritura ("w"). Si no se puede abrir, se muestra un mensaje de error. Si se puede abrir, se muestra un mensaje al usuario indicando que debe ingresar caracteres. Luego, se ejecuta un bucle while en el que se ingresan los caracteres hasta que se presiona enter. Dentro del bucle, cada carácter ingresado se escribe en el archivo utilizando la función putc. Por último, una vez que el usuario ha presionado Enter y se han escrito los caracteres en el archivo, este se cierra y el programa finaliza.

Ejercicio 2: “entender el programa”

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <ctype.h>
#include <math.h>
#define p printf
#define s scanf
int main()
{
  FILE *archivo;
  char ch,cadena[81];
  char *nombre="texto99.txt";
  if((archivo=fopen(nombre,"w"))==NULL)
  {
    p("\n\n***El archivo %s no pudo abrirse ***\n",nombre);
  }
  else
  {
    p("\n\n Este es el contenido del archivo %s caracter a caracter \n\n",nombre);
      while((ch=getc(archivo))!=EOF)
      {
        p("%s",cadena);
      }
    p("\n\n Este es el contenido del archivo %s renglon a renglon \n\n",nombre);
    rewind(archivo);
    while(fgets(cadena,81,archivo));
    p("\n\n **Fin**\n\n");
    getch();
    fclose(archivo);
  }
  return 0;
}
```

💡 ***Explicación codeGPT:*** Este código incluye algunas bibliotecas estándar de C y define algunos macros. En la función principal, se declara una variable de tipo "archivo" y un arreglo de caracteres llamado "cadena". Se inicializa una variable de tipo puntero a un carácter "nombre" con el nombre "texto99.txt".
Luego se realiza una verificación si el archivo se pudo abrir o no usando la función "fopen()", donde el archivo se abre con el modo de escritura. Si el archivo no se puede abrir, se imprime un mensaje de error en la consola, caso contrario se ejecuta un ciclo while, el cual lee el contenido del archivo y lo imprime de forma seguida.
Después, el programa ejecuta una sentencia "rewind ()", que mueve el archivo en posición 0; a continuación, se realiza otro ciclo while con la función "fgets()" para imprimir el contenido del archivo por línea separada.
Finalmente, el archivo se cierra con la línea de código "fclose()" y se devuelve un valor de 0 para indicar que el programa se ha ejecutado correctamente.

Ejercicio 3: capturar un texto de varios renglones, lo escribimos y en el mismo leemos su contenido

*Usar **w+** para escribir y leer el archivo*

```
#include <stdio.h>
#include <string.h>
void procesarArchivo();
int main()
{
    procesarArchivo();
    return 0;
}
```

```
void procesarArchivo()
{
    FILE *f;
    char cadena [80];
    f=fopen("texto.txt","w+");
    if (f!=NULL)
    {
        // Escritura del archivo
        printf("indicar fin con ctrl z\n");
        printf("Escriba un texto: ");
        while (gets(cadena)!=NULL)
        {
            fputs(cadena, f);
            fputc('\n',f);
            printf("Escriba un texto: ");
        }
        // Lectura del archivo
        rewind(f);
        while (fgets(cadena,80,f))
        {
            printf("%s",cadena);
        }
        fclose(f);
    }
    else
    {
        printf("no se pudo abrir el archivo");
    }
}
```

💡 ***Explicacion codeGPT:***

El código es un programa en C que procesa un archivo de texto llamado "texto.txt". El programa comienza abriendo el archivo en modo de escritura y lectura (w+). Luego, el programa solicita al usuario que escriba una cadena de texto y la guarda en "texto.txt". El programa repite esta solicitud de texto hasta que el usuario ingresa "ctrl+z" para indicar el final de la entrada.
A continuación, el programa lee el contenido del archivo "texto.txt" usando un ciclo while y la función fgets() que lee líneas completas desde un archivo. Cada línea leída es impresa por pantalla con printf(). Al finalizar la lectura, el programa cierra el archivo.
En caso de que el archivo no se pueda abrir correctamente, se muestra un mensaje de error indicando que no se pudo abrir el archivo.

La funcion **rewind(f)** se utiliza para mover el puntero de archivo a la posición inicial. Más específicamente, se mueve el indicador de posición de archivo al principio del archivo (byte 0) y borra cualquier indicación de error o fin de archivo. Esto es útil cuando se necesita leer el contenido completo del archivo una vez más desde el principio.

Ejercicio 4: capturar un texto de varios renglones en un archivo y copiarlo en un segundo archivo. Para ello utilizar funciones.

*void mostrar (FILE *);*

*void cargar (FILE *);*

*void grabar(FILE *,FILE *);*

```
#include <stdio.h>
#include <string.h>
void cargar (FILE *); // cargar el texto en un primer archivo
void mostrar (FILE *); // mostrar
void grabar(FILE *,FILE *); // grabar el texto del archivo 1 en el archivo 2
int main()
{
```

```
    FILE f,f2;
    cargar(&f);
    printf("texto del archivo original: \n");
    mostrar(&f);
    grabar(&f,&f2);
    printf("texto del archivo copia: \n");
    mostrar(&f2);
    return 0;
}

void cargar(FILE *f)
{
    char cadena [80];
    if ((f=fopen("texto.txt","w"))!=NULL)
    {
        printf("indicar fin con ctrl z\n");
        printf("Escriba un texto: ");
        while (gets(cadena)!=NULL)
        {
            fputs(cadena, f);
            fputc('\n',f);
            printf("Escriba un texto: ");
        }
        fclose(f);
    }
    else
    {
        printf("no se pudo abrir el archivo");
    }
}

void mostrar (FILE *f)
{
    char cadena [80];
    if ((f=fopen("texto.txt","r"))!=NULL||(f=fopen("texto2.txt","r"))!=NULL)
    {
        while (fgets(cadena,80,f))
        {
            printf("%s",cadena);
        }
        fclose(f);
    }
    else
    {
        printf("no se pudo abrir el archivo");
    }
}

void grabar(FILE *f,FILE *f2)
{
    char cadena[80];
    if ((f=fopen("texto.txt","r"))!=NULL&&(f2=fopen("texto2.txt","w"))!=NULL)
    {

        while (fgets(cadena,80,f))
        {
            fputs(cadena, f2);
        }
        fclose(f);fclose(f2);
    }
    else
    {
        printf("no se pudo abrir el archivo");
    }

}
```

Ejercicio 5: capturar un texto de varios renglones en un archivo y determinar la cantidad de palabras por renglon.

```
#include <stdio.h>
#define n 80
void procesarArchivo(); //cargar texto y mostrarlo por pantalla
void contarPalabras();
int main()
{
    procesarArchivo();
    contarPalabras();
    return 0;
}

void procesarArchivo()
{
    FILE *f;
    char cadena [n];
    if ((f=fopen("texto.txt","w+"))!=NULL)
    {
        // crear y escribir en archivo
        printf("indicar fin con ctrl z\n");
        printf("Escriba un texto: ");
        while (gets(cadena)!=NULL)
        {
            fputs(cadena, f);
            fputc('\n',f);
            printf("Escriba un texto: ");
        }
        rewind(f);
        // mostrar texto del archivo
        printf("\ntexto en el archivo: \n");
        while (fgets(cadena,n,f))
        {
            printf("%s",cadena);
        }
        fclose(f);
    }
```

```
    else
    {
        printf("no se pudo abrir el archivo");
    }
}

void contarPalabras()
{
    FILE *f;
    char cadena [n];
    int cnt = 0;

    if ((f=fopen("texto.txt","r"))!=NULL)
    {
        while (fscanf(f, "%s", cadena) == 1)
        {
            cnt++;
        }
        fclose(f);
        printf("\nEl archivo tiene %d palabras.\n", cnt);
    }
    else
    {
        printf("no se pudo abrir el archivo");
    }
}
```

Ejercicio 6: Idem ejercicio 5, pero tiene que dereminar la palabra más larga por renglón y poder mostrarlo.

```
#include <stdio.h>
#include <string.h>
#define n 80
void procesarArchivo(); //cargar texto y mostrarlo por pantalla
void palMasLarga();
int main()
{
    procesarArchivo();
    printf("\n");
    palMasLarga();
    return 0;
}

void procesarArchivo()
{
    FILE *f;
    char cadena [n];
    if ((f=fopen("texto.txt","w+"))!=NULL)
    {
        // crear y escribir en archivo
        printf("indicar fin con ctrl z\n");
        printf("Escriba un texto: ");
        while (gets(cadena)!=NULL)
        {
            fputs(cadena, f);
            fputc('\n',f);
            printf("Escriba un texto: ");
        }
        rewind(f);
        // mostrar texto del archivo
        printf("\ntexto en el archivo: \n");
        while (fgets(cadena,n,f))
        {
            printf("%s",cadena);
        }
        fclose(f);
    }
    else
    {
        printf("no se pudo abrir el archivo");
    }
}

void palMasLarga() {
    FILE *f;
    char cadena[n];
    int linea = 1;

    if ((f = fopen("texto.txt", "r")) != NULL) {
        while (fgets(cadena, n, f)) {
            char *palabra = strtok(cadena, " ");
            int longitud_maxima = 0, numero_palabra = 0;
            char palabra_mas_larga[n];
            while (palabra != NULL) {
                numero_palabra += 1;
                int longitud_actual = strlen(palabra);
                if (longitud_actual > longitud_maxima) {
                    longitud_maxima = longitud_actual;
                    strcpy(palabra_mas_larga, palabra);
                }
                palabra = strtok(NULL, " ");
            }
            printf("En la linea %d, la palabra mas larga es: %s\n", linea, palabra_mas_larga);
            linea += 1;
        }
        fclose(f);
    }
    else {
        printf("no se pudo abrir el archivo\n");
    }
}
```

Ejercicio 7: Analizar si los siguientes textos son polindromos en un programa en c

le avisara sara si va el

yo hago yoga hoy

somos o no somos

se es o no se es

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

#define n 100

int sonPalindromos(char *cadena);

int main()
{
    FILE *f;
    int i=1;
    if ((f=fopen("palindromos.txt","rt"))!=NULL)
    {
        char cadena[n];
        while ((fgets(cadena,n,f))!=NULL)
        {
            if ((sonPalindromos(cadena))==1)
            {
                printf("el texto %d es palindromo\n",i);
            }
            else
            {
                printf("el texto %d no es palindromo\n",i);
            }
            i++;
        }
        fclose(f);
    }
    else
    {
        printf("no se pudo abrir el archivo");
    }
    return 0;
}

int sonPalindromos(char *cadena) {
    int i, j;
    int longitud = strlen(cadena) - 1; // no contar el carácter nulo '\0' al final
    for (i = 0, j = longitud; i < j; i++, j--) {
        // ignorar espacios y puntuación
        while (i < longitud && !isalnum(cadena[i])) {
            i++;
        }
        while (j > 0 && !isalnum(cadena[j])) {
            j--;
        }
        // comparar caracteres
        if (cadena[i] != cadena[j]) {
            return 0; // no es palíndromo
        }
    }
    return 1; // es palíndromo
}
```

💡 ***Explicacion codeGPT:***

sonPalindromos es una función que recibe una cadena de caracteres y devuelve un entero (0 o 1) dependiendo de si la cadena es un palíndromo o no. Para determinar si es palíndromo, la función utiliza dos variables enteras que se mueven desde el principio y el final de la cadena hacia su centro. La función también ignora los espacios y puntuación en los extremos de la cadena mientras se comparan los caracteres. Si se encuentra una diferencia entre los caracteres que se comparan, la función devuelve 0 y se detiene, indicando que la cadena no es un palíndromo. Si la función llega al final de la cadena y todas las comparaciones son iguales, devuelve 1, indicando que la cadena es un palíndromo.

La función **"isalnum"** es una función que devuelve true si su argumento es una letra o un número, y devuelve false en cualquier otro caso.

Función sonPalindromos alternativa sin usar la función isalnum()

```
int sonPalindromos(char *cadena) {
    int i, j;
    int longitud = strlen(cadena) - 1; // no contar el carácter nulo '\0' al final
    for (i = 0, j = longitud; i < j; i++, j--) {
        // ignorar espacios y puntuación
        while (i < longitud && !(cadena[i] >= 'a' && cadena[i] <= 'z') && !(cadena[i] >= 'A' && cadena[i] <= 'Z') && !(cadena[i] >= '0' && cadena[i] <= '9')) {
            i++;
        }
        while (j > 0 && !(cadena[j] >= 'a' && cadena[j] <= 'z') && !(cadena[j] >= 'A' && cadena[j] <= 'Z') && !(cadena[j] >= '0' && cadena[j] <= '9')) {
            j--;
        }
        // comparar caracteres
        if (cadena[i] != cadena[j]) {
            return 0; // no es palíndromo
```

```
        }
    }
    return 1; // es palíndromo
}
```