UNIVERSIDAD SIMÓN BOLÍVAR
Departamento de Computación y
Tecnología de la Información
CI-2125 Computación I

Nombre: _____

Carnet: _____ Sección: ____

PARCIAL 3 (25pts)

Parte 1. Marque con una V o con una F cada una de las siguientes afirmaciones según sea verdadera o falsa. (3 pts)

	La función puts imprime una cadena de caracteres, excluyendo espacios en blanco, seguido por un salto de línea.
	Una cadena de caracteres se representa mediante un arreglo bidimensional de caracteres.
	En C no se pueden definir arreglos de estructuras.
	Las cadenas de caracteres se pueden pasar a las funciones por valor o por referencia.
	Se puede usar el operador de asignación (=) para copiar una variable tipo estructura en otra del mismo tipo.
	Cuando se define un apuntador a una estructura, el acceso a los miembros de la estructura a la que apunta se realiza utilizando el operador ->
	Para acceder a los miembros de una estructura se utiliza el operador dos puntos (:).
	La función gets lee una línea completa, incluyendo espacios en blanco, hasta que se encuentre un salto de línea.
	Para comparar estructuras se utiliza el operador de igualdad (= =).
	En una cadena de caracteres, el último elemento debe ser el carácter nulo de C (‘\0’).
	Una estructura es un tipo de datos que permite agrupar bajo un mismo nombre, elementos del mismo o de distinto tipo de datos, que se encuentran relacionados lógicamente.
	Una cadena de caracteres está formada por una secuencia de caracteres encerrada entre comitas simples.

Parte 2: Realice la corrida del siguiente segmento de código y rellene la tabla

1. Sea A una matriz de enteros de n x n (con n = 5). Indique cual es el contenido de la matriz luego de ejecutar el siguiente grupo de instrucciones. **(3 Pts)**

```
for(i=0;i<5;i++)
  for(j=0;j<5;j++)
     A[j][i] =(5*i)+j;
```

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
20	21	22	23	24

Parte 3: Dado dos archivos denominados texto1.txt y texto2.txt, se desea que genere un tercer archivo llamado texto3.txt que contenga la información de ambos archivos, de manera que el archivo3.txt contiene la concatenación de los archivos texto1.txt y texto2.txt. Realice el diagrama de flujo que permita dar la solución. (6.5 p ts)

PARTE 2: COMPLETACIÓN (12.5 pts)

Se tiene un archivo donde se almacena la información de los jugadores de la Liga de Béisbol Profesional de Venezuela. Para cada jugador se almacena en una línea: nombre (15 caracteres), apellido (15 caracteres), edad, estado (10 caracteres) y cantidad de jonrones anotados. Cada jugador aparece en una línea diferente. El nombre del archivo es "jugadores.txt". Se desea generar el archivo "novatos.txt" con todos los jugadores con edad menor a 25 años.

Se ha definido un tipo de dato JUGADOR para manejar la información de un jugador. Complete el código:

```
typedef struct { (0.5 pts)
    char nombre[16];    // nombre del jugador
    char apellido[16];  // apellido del jugador
    int edad;           // edad del jugador
    char estado[11];      // estado donde nació el jugador
    int jonrones;          // jonrones del jugador
} JUGADOR;
```

Complete el código de la función `escribirJugador` que escribe del archivo referenciado por el parámetro `fp` un jugador cuyo valor lo recibe en el parámetro `g`:

```
void escribirJugador(FILE *fp,JUGADOR g){ (1 pto)
    fprintf(
        fp,                  // apuntador del archivo de escritura (0.25)
        "%s\t%s\t%d\t%s\t%d\n", // formato de escritura (0.5 pts)
        g.nombre,               // nombre del jugador (0.25)
        g.apellido,             // apellido del jugador (0.25)
        g.edad,                 // edad del jugador (0.25)
        g.estado,                 // estado donde nació el jugador (0.25)
        g.jonrones                 // jonrones del jugador (0.25)
    );
}
```

Complete el código de la función `leerJugador` que lee del archivo referenciado por el parámetro `fp` un jugador el cual lo coloca en la variable referenciada por el parámetro `g`:

```
void leerJugador(FILE *fp,JUGADOR *g){    (1 pto)

    fscanf(

        fp,                 // archivo de  lectura (0.25 pts)

        "%s %s %d %s %d\n", // formato de lectura  (0.5 pts)

        g->nombre,          // nombre del jugador  (0.25 pts)

        g->apellido,        // apellido del jugador  (0.25 pts)

        &g->edad,           // edad del jugador   (0.25 pts)

        g->pais,            // pais del jugador   (0.25 pts)

        &g->goles           // goles del jugador  (0.25 pts)

    );

}
```

Complete el código de la función `copiarNovatos`. Esta función recibe dos parámetros formales llamados `fpEnt` y `fpSal`. El primero es un archivo que contiene jugadores y ha sido previamente abierto para lectura. El segundo es un archivo que ha sido previamente abierto para escritura y que contendrá aquellos jugadores que se consideran novatos por ser su `edad < 25`. Para leer el archivo `fpEnt` se usará la función `leerJugador` definida anteriormente. Para escribir en el archivo `fpSal` se usará la función `escribirJugador` que se definió previamente

```
void copiarNovatos(FILE *fpEnt,FILE *fpSal){ (1pto)

    JUGADOR j;

    // procesamiento de archivos

    while(!feof(fpEnt)){   (0.5  pto)

        leerJugador(fpEnt,&j); (0.5  pto)

        if(j.edad<25){ (0.5  pto)

            // se considera que es novato

            escribirJugador(fpSal,j); (0.5  pto)

        }

    }

}
```

Complete el código de la función `main()` de acuerdo a las especificaciones del programa y las definiciones anteriores:

```
main(){

    FILE *fpJug;

    FILE *fpNov;

    fpJug = fopen("jugadores.txt","r"); (0.5  pto)

    if (fpJug == NULL){ (0.5  pto)

        printf("ERROR: No se pudo abrir \"jugadores.txt\"\n");

    } else {

        fpNov = fopen("novatos.txt","w"); (0.5  pto)

        if (fpNov == NULL){   (0.5  pto)

            printf("ERROR: No se pudo crear \"novatos.txt\"\n");

        } else {

            copiarNovatos(fpJug,fpNov); (0.5  pto)

            fclose(fpNov); (0.25  pto)

        }

        fclose(fpJug);   (0.25  pto)

    }

    return 0;

}
```