1. **(0.9 puntos)** Para cada inciso indique si es verdadero o falso.

   a) Cuando se pasa como parámetro un arreglo se pasa su dirección por referencia.
   b) El operador * (de indirección) se puede utilizar acceder al contenido de memoria de un arreglo.
   c) Un arreglo puede manipularse como un puntero, pero un puntero no puede manipularse como un arreglo.
   d) Los arreglos se pueden copiar con una asignación siempre y cuando tengan el mismo tipo.
   e) Es posible asignar una variable *float* a una variable *int* sin inconvenientes.
   f) Se puede asignar una variable de tipo *float* a una variable *double* sin inconvenientes

2. **(0.9 puntos)** Para cada inciso indique si es verdadero o falso.

   a) Dos variables definidas como estructuras solo pueden compararse solo si tienen el mismo tipo.
   b) El operador -> puede utilizarse cuando se tiene un puntero a cualquier tipo de dato.
   c) *typedef* puede utilizarse para definir tanto un tipo estructura como un tipo predefinido.
   d) No es posible determinar la posición de memoria de un campo de una variable de tipo *struct*.
   e) Dos variables de tipo *struct* no pueden asignarse, aunque tengan el mismo tipo.
   f) Un arreglo de estructuras se pasa por referencia a una función de manera automática.

3. **(0.8 puntos)** Indique qué imprime el programa. Justifique.

```c
void singleton(char *s);
int main(){
    char texto[] = "Hola\0 ?";
    singleton(texto);
    singleton("Desde");
    singleton("Casa");
    return 0;
}
```

```c
void singleton(char * s) {
    static char * p1 = NULL;
    if (p1==NULL){
        p1 = s;
        printf("Nada\n");
    }
    else
        printf("%s\n", p1);
}
```

4. **(0.8 puntos)** Implemente las siguientes estructuras y funciones:
   a. Utilizando *typedef* defina una estructura que represente la inscripción de un alumno en una materia. Los datos de una inscripción son: *cod_materia*(entero de 1 a 50), *cod_alumno* (entero).
   b. Utilizando *typedef* defina los tipos necesarios para implementar el tipo *lista_inscrip* que utilice la estructura definida en a) para definir una lista de inscripciones.
   c. Defina la función *informar_inscripciones* que reciba una lista de inscripciones y la recorra para informar para cada materia la cantidad de alumnos inscriptos.

5. **(0.9 puntos)** Defina dos macros: una que calcule el valor absoluto de 2 números y otra que utilice la primera para calcular la distancia de Manhattan para 2 puntos del plano (P_x, P_y) y (Q_x , Q_y) con la fórmula: $|P_x - Q_x|+|P_y - Q_y|$

6. **(0.9 puntos)** Marque con verdadero o falso las siguientes afirmaciones:
   a) Para duplicar un archivo de texto es mejor utilizar las funciones *fgets/fputs* que *fread/fwrite*.
   b) La función *feof* retorna falso cuando se alcanza el último byte de un archivo.
   c) La función *fprintf* puede utilizar tanto para escribir en un archivo como para escribir en pantalla.
   d) La función *ftell* con parámetro *SEEK_END* permite posicionarse al final de un archivo.
   e) El tamaño de un archivo binario de *struct{int hora; int min;}* es menor que su equivalente de texto.
   f) *fgets(stdout)* equivale a *gets()*.

7. **(1.6 puntos)**
   a) Defina el tipo de dato Pirámide que permita representar de manera dinámica y eficiente una pirámide de enteros con *n* elementos en su base, n-1 en el siguiente nivel, y así sucesivamente hasta llegar al último nivel con 1 elemento.
   b) Dados los siguientes prototipos, implementar una función que permita reservar y liberar una pirámide de n elementos de base.

```c
Piramide crear_piramide(int n);   void liberar_piramide(Piramide p, int n);
```

8. **(2.0 puntos)** Desarrolle una biblioteca que implemente una función para transformar un archivo binario en uno de texto. Indique claramente el nombre del archivo donde se encuentra el prototipo de la función, la declaración de tipos y la correspondiente implementación de la función.

El archivo binario u origen contiene información sobre la valoración de series de televisión: nombre (máximo 30 caracteres), visualizaciones mensuales (entero), valoración de los usuarios (real de 0 a 10). El archivo de texto o destino debe tener una línea por cada registro del archivo binario y los campos deben estar separados por '|'. La valoración tener un dígito decimal.

La función de conversión recibe dos *strings*, el primero con el nombre de un archivo binario y el segundo con el nombre de un archivo de texto y retorna un valor numérico con el resultado: 0➔ conversión exitosa, 1➔fallo al abrir archivo binario, 2➔ fallo al crear el archivo de texto.

9. **(1.2 puntos)** Escriba un programa que reciba el nombre de un archivo binario y el nombre de un archivo de texto como argumentos a la función *main* y realice la conversión utilizando la función de la biblioteca desarrollada en el punto anterior. En caso de que el programa no reciba la cantidad de argumentos correcta debe imprimir un mensaje de error. También debe imprimir el resultado de la conversión.