**INDIQUE VERDADERO O FALSO PARA CADA UNA DE LAS SIGUIENTES EXPRESIONES:**

**El uso de funciones:**

1. Evita la necesidad de repetir las mismas instrucciones de forma redundante. Verdadero
2. Hace que el programa tarde mas tiempo en su ejecución. Falso
3. Brinda mayor claridad lógica por la descomposición de un programa en partes concisas. Verdadero

**Una función debe devolver un tipo de dato:**

1. Sin Haber sido declarado previamente. Falso
2. Que halla sido declarado en la cabecera de la función. Verdadero
3. Muchos valores que se hallan declarado en la cabecera de la función. Falso

**En funciones un pasaje por copia significa:**

1. Que el parámetro formal de la función puede o no llevar el nombre de la variable en el prototipo de la función Verdadero
2. Que se copia a la función la dirección de memoria del parámetro real que se utiliza en la llamada a la función. Falso
3. Que se copia a la función el valor del parámetro real que se utiliza en la llamada a la función. Verdadero

**En funciones cuando se hace un pasaje por copia:**

1. La modificación de los valores de los parámetros formales **no impacta** en los valores de los parámetros reales con los que se llamó a la función. Verdadero
2. La modificación de los valores de los parámetros formales **impacta** en los valores de los parámetros reales con los que se llamó a la función. Falso

**Variables locales:**

1. Son aquellas cuyo ámbito se restringe al bloque de la función en la que se han declarado y cualquier otra función que la necesite. Verdadero
2. Cualquier variable que declaremos dentro de una función, es local a esa función. Esta situación permite que existan variables con el mismo nombre en diferentes funciones y que no mantengan ninguna relación entre sí. Verdadero

**Variables globales:**

1. Son cualquier variable que se declare en el prototipo de una función. Falso
2. Si declaramos una variable global al principio del programa, y puede ser utilizada en cualquier parte de la función mail(). Verdadero, si la función es MAIN no MAIL
3. El ámbito de una variable global puede afectar a todas las funciones que componen el programa. Verdadero
4. Determinar si las siguientes afirmaciones son verdaderas o falsas y justificar en ambos casos:

Los archivos pueden ser accedidos de forma secuencial, esto quiere decir que me puedo posicionar de forma directa en el lugar del archivo que desee sin pasar por las posiciones anteriores

Falso, en c++ la información sólo puede leerse y escribirse empezando desde el principio del archivo.

Solo se pueden utilizar cuatro operadores aritméticos sobre punteros y estos

son: suma, resta, multiplicación y división**.**

**Falso, no se pueden sumar, ni restar ni multiplicar, ni dividir punteros entre sí. Y en el caso de trabajar sobre enteros sólo se pueden realizar aritméticamente 2 cosas, suma y resta de un entero sobre un puntero, que suma /resta la posición de memoria del mismo.**

1. Responder las siguientes preguntas:

* Cuales son las funciones que inicializan y cierran un archivo. De que tipo es el puntero con el que trabaja este archivo.

Para abrir un archivo: FILE \*fopen(char \*nombre, char \*mode);

Para cerrar un archivo: int fclose(FILE \*flujo);

El tipo de puntero es un puntero tipo FILE.

La estructura FILE es la estructura que controla los ficheros y se encuentra en la cabecera stdio.h. Los flujos utilizan estos punteros a ficheros para realizar las operaciones E/S. La siguiente línea de código declara una variable de tipo puntero a fichero:

* De que manera en un array de caracteres se marca su final y por que debe contemplarse al dimensionar el array.  
    
  Un array marca su final con el carácter “/0” el mismo funciona como límite para poder recorrer o mostrar una cadena de caracteres
* Cuales son las dos estructuras utilizadas para recorrer una matriz ejemplifica

Podemos recorrer las matrices de dos maneras, unidimensionalmente y bidimensionalmente.

Ejemplo: ya tu sabes el mismo de toda la vida.

1. Determinar si las siguientes afirmaciones son verdadera o falsas y justificar en ambos casos.
   * Una variable es local al main solo si esta declarada fuera de este y no afecta solo a este bloque.

Falso, puede ser local al main también siendo declarada dentro de la misma.

* + En C todos los argumentos que se pasan a una función se pasan por valor.

Falso, pueden ser pasados también por referencia.

* + Para pasar la dirección de memoria donde se encuentra un valor se utiliza un operador del tipo variable \*.

Ya lo hablamos esto (ver primer teoría)

* + En vectores multidimensionales (Matriz), es necesario manipular cada uno de los subíndices en forma conjunta para poder controlar el funcionamiento del for.

Verdadero.

* + Cuando se utilizan punteros para pasar por referencia una cadena de caracteres este en realidad apunta a la primera letra y no a la palabra completa.

Verdadero

**Explicar el concepto de punteros y los operadores que se utilizan para su manejo.**

El puntero es una variable que apunta a la dirección hexadecimal de la memoria de otra variable.  
ejemplo:

int variable; //Creamos un entero

int \* apuntador = &variable; //Creamos una apuntador a la posición en memoria de "variable"

\*apuntador = 20; //Le asignamos un valor a esa posición de memoria.

**Explicar la clasificación de archivos según el tipo de contenido y la forma de acceso.**

**Apertura de archivos de texto.**

|  |  |
| --- | --- |
| Modo | Significado |
| “r” | Read, Leer. Abre un archivo para lectura (el archivo tiene que existir). |
| “w” | Write, escribir. Crea un archivo de escritura. Si el archivo ya existe borra el contenido que tiene. |
| “a” | Append Agregar. Abre un archivo para agregar datos a un archivo ya existente. |
| “r+” | Read+, Abre un archivo para lectura/escritura. El archivo tiene existir. |
| “w+” | Crea un archivo para lectura/escritura. El archivo puede no existir. Si el archivo tenía datos estos se borran en la apertura. |
| “a+” | Crea o abre un archivo para lectura/ escritura. Los datos se escriben al final del archivo. |

**Explicar para que se utiliza el prototipo de una función y dar un ejemplo.**

 El **prototipo** proporciona información sobre el tipo devuelto o producido por la función, así como sobre sus parámetros. Se utiliza para verificar los llamados a la función, ya que controla el número y el tipo de los parámetros, comprobando si son los apropiados. En nuestro ejemplo, cada llamado a la función llamada “relleno()” debe tener exactamente tres parámetros o el compilador dará un mensaje de error.