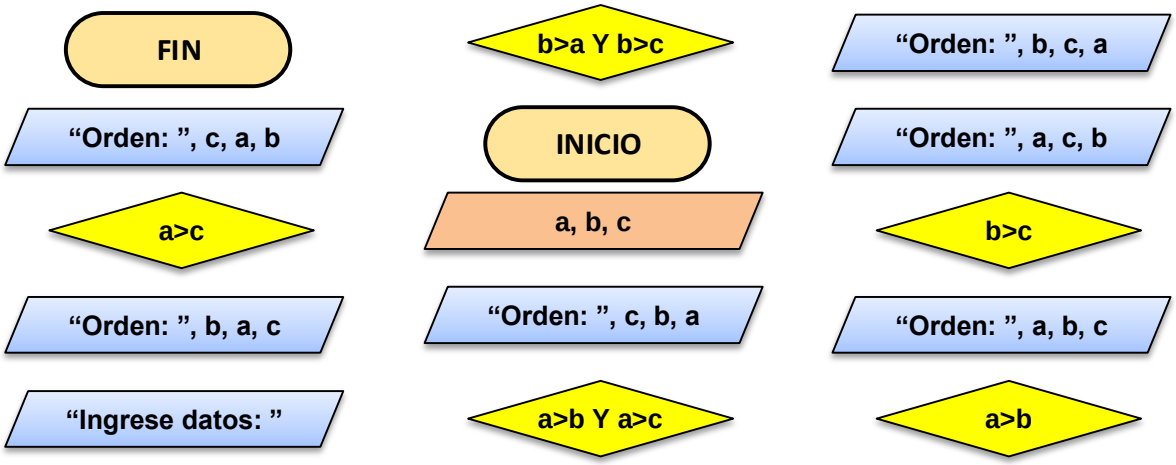
Trabajo Práctico Nº o

Tema: Fundamentos de Programación. Temas de Repaso

Apellido y Nombre: ..... Fecha:...../...../.....

EJERCICIOS

1) Los siguientes bloques corresponden a un algoritmo (DIAGRAMA DE FLUJO) que analiza 3 números ingresados por el usuario y los muestra en orden creciente. Conecte los bloques del diagrama correctamente.



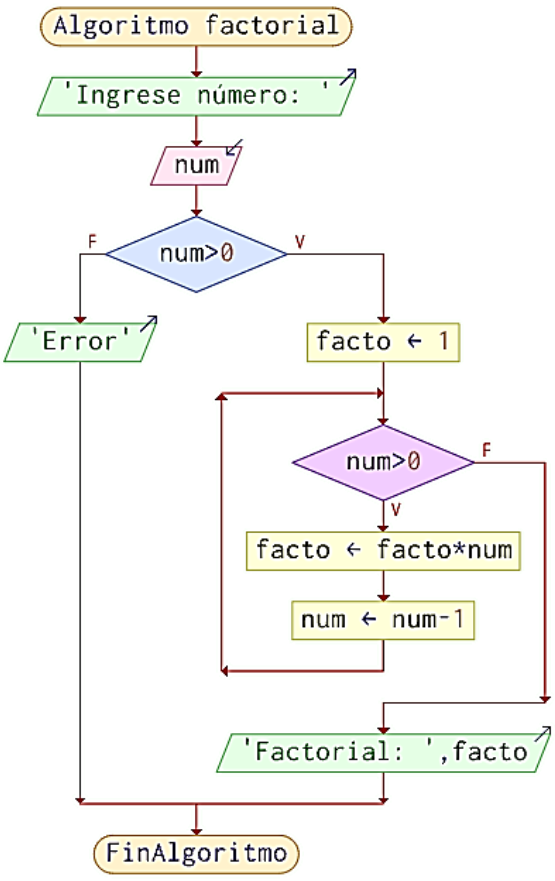
2) Diseñe un algoritmo (DIAGRAMA DE FLUJO) que calcule el producto, mediante sumas sucesivas, de 2 números enteros ingresados por el usuario. Considere que los valores de entrada pueden ser positivos, negativos o cero por lo que debe incluir los controles necesarios para realizar correctamente el cálculo. Realice la prueba de escritorio para A=7, B=-4 y A=-8, B=-5. Codifique en C/C++ el algoritmo diseñado.

3) El siguiente diagrama de flujo corresponde a un algoritmo que calcula el factorial de un número entero positivo. A partir de éste diseñe las versiones que se indican a continuación:

   a. el bucle de cálculo debe implementarse con estructuras MIENTRAS y criterio de finalización por BANDERA.

   b. el bucle de cálculo debe implementarse con estructuras REPETIR y criterio de finalización por CONTADOR.

   c. Considerando el ítem b), realice la prueba de escritorio para *num=-3*, *num=0* y *num=4*.



4) Realice la prueba de escritorio del siguiente módulo, determine su propósito y escriba el código C/C++ equivalente.

```
FUNCIÓN misterio (E a: ENTERO, E b: ENTERO): ENTERO
VARIABLE
   s: LÓGICO
   i,c: ENTERO
INICIO
    c←0
    s← (a<0) Y (b<0) O (a>0) Y (b>0)
    a←abs(a)
    b←abs(b)
    MIENTRAS a>b HACER
      c←c+1
      a←a-b
    FIN_HACER
    SI s=FALSO ENTONCES
      c←(-1)*c
    FIN_SI
    misterio←c
FIN
```

5) Realice la prueba de escritorio del siguiente módulo, determine su propósito y escriba el pseudocódigo equivalente.

```
void enigma(int n, int &c)
{ int m;
  c=(-1)*n;
  m=n*(n+1);
  while(n!=0)
    { c=c+m;
      m=m-2;
      n--;
    }
}
```

6) Complete el código faltante en los siguientes módulos recursivos

```
// el módulo calcula el cociente de la división entera entre 2 N° positivos.
int cociente(int a, int b)
{
  if ( ? )
    ?????
  else
    return cociente(a-b,b)+1;
}
```

```
// el módulo calcula el valor de un término n de la serie de Fibonacci
int fibo(int n)
{
  if (n==1 || n==2)
    return 1;
  else
    ?????
}
```

```
// el módulo determina si un número entero positivo es par o impar.
int impar(int n)
{
  if ( ? )
    ?????
  else
    ?????
}
```

```
// el módulo calcula la cantidad de dígitos de un número entero positivo.
int contar_digitos(int num)
{
  if ( ? )
    ?????
  else
    ?????
}
```

```
// el módulo calcula la suma de elementos de un vector de enteros.
int suma_vector(tvector m, int ocup)
{
  if ( ? )
    ?????
  else
    ?????
}
```

```
// el módulo muestra el vector desde la último posición hacia la primera.
void listar_inverso(tvector d, int ocup)
{
  if ( ? )
    ?????
  else
    ?????
}
```

```
// el módulo determina si un valor dado se encuentra o no almacenado en un vector (de caracteres).
bool buscar(tvector p, int oc, char b)
{
  if ( ? )
    ?????
  else
  { if ( ? )
      ?????
    else
      ?????
  }
}
```

```
// el módulo determina si un vector (de enteros) sólo contiene valores impares.
int suma_impares(tvector a, int ocup)
{
  if ( ? )
    ?????
  else
    { if ( ? )
        ?????
      else
        ?????
    }
}
```

7) Utilizando la estructura general de programa, diseñe un programa modular que permita manejar un vector de enteros de 30 elementos. Considere que el programa debe presentar un menú con las siguientes opciones: 1) Agregar un valor al vector (desde las últimas posiciones hacia las primeras), 2) Determinar si un dato se encuentra presente o no en el arreglo, 3) Contar los valores primos del arreglo, y 4) Salir, que presente el mensaje "Fin de programa". Cada opción debe implementarse mediante módulos especificando el paso de parámetros.

Nota: recuerde que las operaciones de inserción/eliminación sobre arreglos sólo insertan/eliminan un valor cuando son invocadas, NO realizan cargas completas o eliminaciones totales.

8) Dado un vector de caracteres de 16 elementos:
   a. Consigne la declaración de tipos y variables de la estructura.
   b. Diseñe un procedimiento/función que permita insertar valores en el vector, considerando un criterio decreciente.
   c. Diseñe un procedimiento/función que determine, aplicando búsqueda binaria, en qué **posición** del vector se encuentra un carácter especificado por el usuario. Considere que si el valor no existe debe obtenerse como resultado -1.
   d. Diseñe un procedimiento/función **recursivo** que muestre los valores del vector que sean minúsculas.

9) Dada una matriz 7x7 de enteros:
   a. Consigne la declaración de tipos y variables de la estructura.
   b. Diseñe un procedimiento/función que realice la carga aleatoria de valores de 3 cifras en la matriz.
   c. Diseñe un procedimiento/función que calcule, por cada columna, la suma de elementos de la matriz.
   d. Diseñe un procedimiento/función que muestre, según un parámetro de opción, los elementos ubicados por encima de la diagonal principal o aquellos que se encuentren por debajo de ésta.

10) Dado un arreglo multidimensional (5x3x4) de enteros y un vector del mismo tipo e igual cantidad de elementos:
   a. Consigne la declaración de tipos y variables de la estructura.
   b. Diseñe un procedimiento/función que realice la carga aleatoria de valores en el arreglo multidimensional. Considerando que los valores aleatorios deben pertenecer al intervalo [500, 999].
   c. Diseñe un procedimiento/función que copie los valores múltiplos de 3 del arreglo multidimensional al vector. Considere que la identificación de múltiplos de 3 debe realizarse mediante un módulo que implemente el método de descomposición de dígitos.
   d. Diseñe un procedimiento/función que determine los valores máximo y mínimo del arreglo.
   e. Calcule el **rango** del arreglo multidimensional (indique la fórmula aplicada y el cálculo realizado).

11) Considerando el siguiente vector:

13	28	11	31	9	23	16

a. Indique, gráficamente y paso a paso, cómo se realiza la ordenación **decreciente** del vector aplicando el *método de ordenación Burbuja*.

b. Sabiendo que, en cada recorrido, el algoritmo de ordenación por *Selección* compara una posición (pivote) del arreglo con las restantes, realizando intercambios cuando sea necesario; modifique este algoritmo para que en cada recorrido sólo se realice un intercambio. Considere criterio de ordenación creciente.

12) El propietario de la tienda online *AllBooks* desea registrar información acerca de los 4600 libros disponibles en el inventario de la tienda. Por cada libro debe guardarse: código de libro, título, género, autor, isbn, formato (impreso, digital), cantidad de páginas, datos editoriales (año de edición, editorial), unidades disponibles. Para ello, se solicita:

a. Consigne la declaración de tipos y variables de la estructura que represente la situación planteada.

b. Diseñe los procedimientos/funciones que **listen** los libros (título, autor e isbn) cuyo género y año de edición correspondan a lo solicitado por el usuario. Indique cuántos libros se encontraron.

c. Diseñe los procedimientos/funciones que **muestren** el libro (título, género y editorial) con más páginas.

13) Dado el siguiente módulo, ¿cuál es la definición de datos que le corresponde?

```
void listar_empleados(tvector empleados, int ocup, tcad depto)
{int i, cont=0;
  for(i=0;i<=ocup;i++)
   if (strcmp(empleados[i].depto,depto)==0)
    { cout << "Legajo: " << empleados[i].legajo << endl;
      cout << "Apellido: " << empleados[i].apellido << endl;
      cout << "Nombre: " << empleados[i].nombre << endl;
      cout << "Cargo: " << empleados[i].cargo.nombre << endl;
      cout << "Sueldo básico: " << empleados[i].cargo.basico << endl;
      cout << "Fecha de ingreso: " << empleados[i].ingreso.dia;
      cout << "/" << empleados[i].ingreso.mes;
      cout << "/" << empleados[i].ingreso.anio << endl;
      cont++;
    }
   cout << "Empleados de " << depto << ": " << cont << endl;
}
```

14) El gerente de una clínica desea registrar información acerca de 2000 pacientes y 80 obras sociales que operan con la institución. Por cada paciente se debe guardar: id de paciente, apellido, nombre, DNI, fecha de nacimiento (día, mes, año), domicilio (número, calle, barrio) y carnet (número de carnet, plan y código de obra social). Y por cada obra social se debe guardar: código de obra social, nombre de obra social, sitio web y cantidad de prestadores. Para ello, se solicita:

a. Consigne la declaración de tipos y variables de la estructura que represente la situación planteada.

b. Diseñe los procedimientos/funciones que **listen**, por cada obra social, los pacientes (apellido, nombre, número de carnet y plan) correspondientes. Indique cuántos pacientes se listaron por obra social.

c. Diseñe los procedimientos/funciones que **muestren** las obras sociales (nombre de obra social y sitio web) que no tengan pacientes registrados en la clínica.

15) El administrador del sitio de videos *MyTube* desea registrar información acerca de sus 1400 usuarios y los 6000 videos publicados. Por cada usuario se debe guardar: id de usuario, nickname, apellido, nombre, contacto (email, teléfono) y cantidad de seguidores. Y por cada video se debe guardar: id de video, título, descripción, duración (horas, minutos, segundos), fecha de publicación (día, mes, año), cantidad de vistas e id de usuario. Para ello, se solicita:

a. Consigne la declaración de tipos y variables de la estructura que represente la situación planteada.

b. Diseñe los procedimientos/funciones que **listen**, para cada usuario, los videos (título, descripción y duración) correspondientes. Indique la cantidad de videos publicados por cada usuario.

c. Diseñe los procedimientos/funciones que **muestre** el video (título y duración) con la mayor cantidad de vistas de cada usuario.