UNIVERSIDAD NACIONAL DE INGENIERÍA

Facultad de Ciencias

Escuela Profesional de Ciencia de la Computación

Examen Parcial

CC112 **13/05/2025 Tiempo: 2:30 horas** **Ciclo: 2025-I**

Normas:

1. No compartir respuestas/consultas con sus compañeros a través de chats, redes sociales u otros medios digitales.
2. No se permiten apuntes de clase.
3. Las soluciones serán enviadas a la plataforma y/o a la cuenta de correo del profesor.
4. Todo acto anti-ético será amonestado y registrado en el historial del estudiante.

Apellidos: _____ Nombres: _____

Código: _____ Sección: _____

1. [5 puntos] El siguiente ejercicio tiene 3 partes:

```
void BurbujaRecursivo(int* arr, int n) {
   if (n <= 1)
      return;
   for (int j = 1; j < n; ++j)
      if ( arr[j] > arr[0] )
         intercambiar(arr[0], arr[j]);
   BurbujaRecursivo(arr+1, n-1);
}


void main(){
   int *vec1 = nullptr, size = 10;
   int *vec2 = nullptr;

   CreateVector(vec1, size); // Crear la memoria para este puntero
   ReadArray(vec1, size, cin); // Lee los datos desde el cin
   BurbujaRecursivo(vec1, size); // Funcion definida arriba
   PrintArray(vec1, size, cout);   // Imprime los datos en el cout

   CreateVector(vec2, size); // Crear la memoria para este puntero
   ReadArray(vec2, size, cin); // Lee los datos desde el cin
   BurbujaRecursivo(vec2, size);  // Funcion definida arriba
   PrintArray(vec2, size, cout);
```

```
    Intercambiar(vec1, vec2); // Parte 2 del ejercicio
    PrintArray(vec2, size, cout);
    PrintArray(vec2, size, cout);

    DestroyVector(vec1);   // Destruye el vector y deja en nullptr vec1
    DestroyVector(vec2);   // Destruye el vector y deja en nullptr vec2
}
```

Parte 1. Escriba el código de las funciones:
CreateVector,
ReadArray,

PrintArray,
DestroyVector

Observe que las 4 funciones no devuelven valor (son void) y manejan todo a través de los parámetros.

La forma de llamar a estas funciones **no debe cambiar**.

Parte 2. Suponga que queremos intercambiar ambos vectores pero el size fuese 1000. Intercambie ambos vectores sin mover los 1000 elementos.
La función a diseñar sería llamada asi:

Intercambiar(vec1, vec2);

Está función no devuelve nada (void) y todo lo maneja a través de los parámetros

Parte 3. La función BurbujaRecursivo ejecuta el ordenamiento de un vector de forma recursiva con el método de burbuja. Sin embargo, solamente ordena de forma ascendente. Amplie/modifique más la función Burbuja para que pueda ordenar de forma ascendente y/o descendente.
Aqui no es aceptable:

- Enviarle un parámetro tipo boolean o 1, 0 o algún typedef o enum
- Usar variables globales por todas las desventajas que ya se conoce que tienen

2. [5 puntos] El determinante de una matriz cuadrada A de tamaño $n \times n$ puede calcularse recursivamente utilizando el método de expansión por cofactores, aplicando la siguiente relación:

$$\det(A) = \sum_{j=0}^{n-1} (-1)^{i+j} \,.\, A[i][j] \,.\, \det\left(M_{ij}\right)$$

donde, M_{ij} es la submatriz que se obtiene al eliminar la fila i y la columna j de la matriz original A.

Utilizando, exclusivamente punteros para el manejo de matrices, implemente un programa que solicite al usuario el tamaño n (máximo 10) de una matriz cuadrada, lea los elementos y calcule su determinante de manera que:

- Su implementación debe seleccionar automáticamente la fila con mayor cantidad de ceros para minimizar el número de llamadas recursivas.
- Si alguna fila es nula, debe concluir que el determinante es cero, sin realizar más cálculos.

Ejemplo de ejecución:
Ingrese el tamaño de la matriz (máximo 10): 3
Ingrese los elementos de la matriz:
1 0 0
2 4 5
5 7 3
Determinante = -23

3. [5 puntos] El análisis de mapas bidimensionales es una herramienta fundamental en problemas de recorridos y agrupamiento espacial. Este problema consiste en desarrollar un programa en C++ que utilice un puntero doble (int**) para recorrer y manipular un mapa bidimensional representado mediante un arreglo estático. Cada celda del mapa contiene un valor entero que indica si el terreno es transitable (0) o un obstáculo (1). El usuario deberá ingresar las dimensiones del mapa (respetando un límite máximo predefinido, como 100x100) y los valores de cada celda. Luego, el programa deberá contar cuántas regiones transitables conectadas existen (componentes conexas de ceros), considerando movimientos en las cuatro direcciones (arriba, abajo, izquierda, derecha). El programa debe trabajar utilizando un puntero doble para acceder a los datos del mapa.

   Una **región conexa de ceros** es un grupo de celdas con valor 0 que están conectadas por sus **vecinos adyacentes** (por lo general, en 4 direcciones: arriba, abajo, izquierda, derecha).

   Ejemplo de entrada:
   Ingrese el número de filas y columnas del mapa: 4 5
   Ingrese los valores del mapa (0 = camino, 1 = obstáculo):
   1 0 0 1 1
   1 0 1 1 0
   1 1 1 0 0
   0 1 1 1 1

Salida Esperada: Mapa ingresado:
1 0 0 1 1
1 0 1 1 0
1 1 1 0 0
0 1 1 1 1
Número de regiones transitables conectadas: 3

4. [5 puntos] Desarrolle un programa en C++ que reciba una cadena de caracteres, y realice las siguientes operaciones usando únicamente punteros.
    a. Eliminar todos los signos de puntuación y espacios.
    b. Invertir la cadena completa.
    c. Dividir la cadena resultante en substrings de longitud creciente comenzando en 1 (por ejemplo: la primera letra, luego las dos siguientes, luego tres, etc.) hasta que ya no se pueda continuar.

Ejemplo de ingreso:
“Hola, mundo feliz!”;

Salida esperada:
z
il
efd
odnu
maloH