

Indicaciones específicas:

- Esta evaluación contiene 12 páginas (incluyendo esta página) con 3 preguntas. El total de puntos son 20.
- Esta prohibido el uso de celulares y audífonos
- Crea una carpeta llamada: PC2
- Luego crea una subcarpeta para cada pregunta:
 1. p1
 2. p2
 3. p3
- Una vez que termines de realizar las 3 preguntas:
 1. Elimina el directorio cMake-Buil-debug de cada pregunta
 2. Comprime la carpeta PC2 usando el winzip y obtendrás el archivo PC2.zip
- Sube el archivo **PC2.ZIP** al gradescoupe a www.gradescope.com.
- **Recuerda que solo se calificará si has enviado en el formato indicado.**

Competencias:

- Para los alumnos de la carrera de Ciencia de la Computación

Aplicar conocimientos de computación apropiados para la solución de problemas definidos y sus requerimientos en la disciplina del programa. (nivel 2)

Diseñar, implementar y evaluar soluciones a problemas complejos de computación.(nivel 2)

Crear, seleccionar, adaptar y aplicar técnicas, recursos y herramientas modernas para la práctica de la computación y comprende sus limitaciones. (nivel 2)
- Para los alumnos de las carreras de Ingeniería

Aplicar conocimientos de ingeniería en la solución de problemas complejos de ingeniería (nivel 2).

Diseñar soluciones relacionados a problemas complejos de ingeniería (nivel 2)

Crear, seleccionar y utilizar técnicas, habilidades, recursos y herramientas modernas de la ingeniería y las tecnologías de la información, incluyendo la predicción y el modelamiento, con la comprensión de sus limitaciones (nivel 2)

- Para los alumnos de Administración y Negocios Digitales

Analizar información verbal y/o lógica proveniente de distintas fuentes, encontrando relaciones y presentándola de manera clara y concisa (nivel 2)

Analizar y evaluar el comportamiento del consumidor y el desarrollo de estrategias comerciales (nivel 2)

Trabajar de manera efectiva con equipos multidisciplinarios y diversos en género, nacionalidad, edad, etc. (nivel 2)

Calificación:

Tabla de puntos (sólo para uso del professor)

Question	Points	Score
1	7	
2	6	
3	7	
Total:	20	

1. (7 points) **Evalua matrices dinámicas**

Realice un programa que permita leer el orden de una matriz cuyos valores sean mayores a 2, y el programa genere dos matrices dinámicas con valores aleatorios entre 1 y 50, las imprima y luego forme una tercera matriz también dinámica con la mezcla de los datos de las dos matrices anteriores. La tercera matriz tendrá en la primera fila los datos de la matriz 1 y a continuación y en la misma fila los datos de la matriz 2 y así sucesivamente.

El programa debe:

- Verificar que orden de la matriz. El valor de filas y columnas debe ser mayor a 2.
- Imprimir la matriz 1 y matriz 2
- Debe generar la tercera matriz, **para asignar el puntaje a esta pregunta es necesario que se genere la tercera matriz y no solo se imprima.**
- El programa debe hacer un buen uso de la memoria.

Algunos ejemplos de diálogo de este programa serían:

Listing 1: Ejemplo 1

```

Filas [mayor a 2] : 2
Columnas [mayor a 2 : 1
Filas [mayor a 2] : 4
Columnas [mayor a 2 : 6

Matriz 1

    34    37    28    16    44    36
    37    43    50    22    13    28
    41    10    14    27    41    27
    23    37    12    19    18    30

Matriz 2

    33    31    13    24    18    36
    30     3    23     9    20    18
    44     7    12    43    30    24
    22    20    35    38    49    25

Matriz Nueva - Union
    34    37    28    16    44    36    33    31    13    24
           18    36
    37    43    50    22    13    28    30     3    23     9
           20    18
    41    10    14    27    41    27    44     7    12    43
           30    24

```

23	37	12	19	18	30	22	20	35	38
	49	25							

Listing 2: Ejemplo 2

```

Filas [mayor a 2] : 5
Columnas [mayor a 2 : 4

Matriz 1

    34    37    28    16
    44    36    37    43
    50    22    13    28
    41    10    14    27
    41    27    23    37

Matriz 2

    12    19    18    30
    33    31    13    24
    18    36    30     3
    23     9    20    18
    44     7    12    43

Matriz Nueva - Union
    34    37    28    16    12    19    18    30
    44    36    37    43    33    31    13    24
    50    22    13    28    18    36    30     3
    41    10    14    27    23     9    20    18
    41    27    23    37    44     7    12    43

```

Listing 3: Ejemplo 3

```

Filas [mayor a 2] : 7
Columnas [mayor a 2 : 3

Matriz 1

    34    37    28
    16    44    36
    37    43    50
    22    13    28
    41    10    14
    27    41    27
    23    37    12

```

Matriz 2					
19	18	30			
33	31	13			
24	18	36			
30	3	23			
9	20	18			
44	7	12			
43	30	24			
Matriz Nueva - Union					
34	37	28	19	18	30
16	44	36	33	31	13
37	43	50	24	18	36
22	13	28	30	3	23
41	10	14	9	20	18
27	41	27	44	7	12
23	37	12	43	30	24

Listing 4: Ejemplo 1

Filas [mayor a 2] : 3									
Columnas [mayor a 2] : 8									
Matriz 1									
34	37	28	16	44	36	37	43		
50	22	13	28	41	10	14	27		
41	27	23	37	12	19	18	30		
Matriz 2									
33	31	13	24	18	36	30	3		
23	9	20	18	44	7	12	43		
30	24	22	20	35	38	49	25		
Matriz Nueva - Union									
34	37	28	16	44	36	37	43	33	31
	13	24	18	36	30	3			
50	22	13	28	41	10	14	27	23	9
	20	18	44	7	12	43			
41	27	23	37	12	19	18	30	30	24
	22	20	35	38	49	25			

Los criterios en la rúbrica (y el puntaje respectivo) se condicionan a que la solución presentada corresponda al problema planteado

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Ejecución	El diseño del algoritmo es ordenado y claro, siguiendo buenas prácticas en programación. La ejecución es correcta (3pts)	El diseño del algoritmo es ordenado y claro. La ejecución es correcta (2pts)	El diseño tiene algunas deficiencias pero la ejecución es correcta (1pts).	El diseño es deficiente y la ejecución no es correcta (0.5pts)
Sintaxis	No existen errores sintácticos o de compilación (2pts)	Existen algunos errores sintácticos de menor relevancia, que no afectan el resultado (1.5pts).	Existen errores sintácticos en la forma de ejecución, que no afectan el resultado (1pts).	El código tiene errores de sintaxis que afectan el resultado (0.5pts)
Optimizacion	El código es óptimo y eficiente. De buen performance e interacción con el usuario (2pts)	El código es de buen performance durante la ejecución (1.5pts)	El código no está optimizado pero la ejecución no es deficiente(1pts)	El código no está optimizado y la ejecución es deficiente (0pts)

2. (6 points) **Vectores**

Realice un programa que lea el orden de una matriz cuadrada, la llene con valores aleatorios entre 1 y 50 y luego forme un vector con los datos en este orden: primero figuraran los datos de la diagonal principal y luego los datos de la diagonal secundaria.

El programa debe:

- Verificar que el tamaño de la matriz cuadrada tenga al menos 3 filas
- La matriz que cree debe ser dinámica. No olvide liberar el espacio de memoria
- El programa debe formar el vector, no basta con imprimir los datos.
- Para asignar el puntaje a esta pregunta es **necesario que utilice un una matriz dinámica y un vector**

Listing 5: Ejemplo

Filas [mayor a 2] : 4							
34	37	28	16				
44	36	37	43				
50	22	13	28				
41	10	14	27				
Vector							
34	36	13	27	16	37	22	41

Listing 6: Ejemplo

Filas [mayor a 2] : 7													
34	37	28	16	44	36	37							
43	50	22	13	28	41	10							
14	27	41	27	23	37	12							
19	18	30	33	31	13	24							
18	36	30	3	23	9	20							
18	44	7	12	43	30	24							
22	20	35	38	49	25	16							
Vector													
34	50	41	33	23	30	16	37	41	23	33	30	44	22

Listing 7: Ejemplo

Filas [mayor a 2] : 5									
34	37	28	16	44					
36	37	43	50	22					
13	28	41	10	14					
27	41	27	23	37					
12	19	18	30	33					
Vector									
34	37	41	23	33	44	50	41	41	12

Listing 8: Ejemplo

Filas [mayor a 2] : 10														
34	37	28	16	44	36	37	43	50	22					
13	28	41	10	14	27	41	27	23	37					
12	19	18	30	33	31	13	24	18	36					
30	3	23	9	20	18	44	7	12	43					
30	24	22	20	35	38	49	25	16	21					
14	27	42	31	7	24	13	21	47	32					
6	26	35	28	37	6	47	30	14	8					
25	46	33	46	15	18	35	15	44	1					
38	9	27	29	39	35	4	2	5	50					
33	11	27	19	40	13	27	37	45	40					
Vector														
34	28	18	9	35	24	47	15	5	40	22	23	24	44	38
	7	28	33	9	33									

Los criterios en la rúbrica (y el puntaje respectivo) se condicionan a que la solución presentada corresponda al problema planteado

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Ejecución	El diseño del algoritmo es ordenado y claro, siguiendo buenas prácticas en programación. La ejecución es correcta (3pts)	El diseño del algoritmo es ordenado y claro. La ejecución es correcta (2pts)	El diseño tiene algunas deficiencias pero la ejecución es correcta (1pts).	El diseño es deficiente y la ejecución no es correcta (0.5pts)
Sintaxis	No existen errores sintácticos o de compilación (2pts)	Existen algunos errores sintácticos de menor relevancia, que no afectan el resultado (1.5pts).	Existen errores sintácticos en la forma de ejecución, que no afectan el resultado (1pts).	El código tiene errores de sintaxis que afectan el resultado (0.5pts)
Optimizacion	El código es óptimo y eficiente. De buen performance e interacción con el usuario (2pts)	El código es de buen performance durante la ejecución (1.5pts)	El código no está optimizado pero la ejecución no es deficiente(1pts)	El código no está optimizado y la ejecución es deficiente (0pts)

3. (7 points) **POO**

Desarrolle un programa orientado a objetos, que permita leer como dato un número entero y el programa a través de los servicios que ofrece la clase CArreglo, permita:

- Generar n números al azar cuyos valores estan entre 1 y 50 y los almacene en un arreglo dinámico de números enteros.
- Imprima el arreglo
- Halle e imprima la suma
- Halle e imprima el número mayor
- Halle e imprima el número menor
- Cuente el número de números primos que se encuentran en el arreglo. Si se ha generado números repetidos en el arreglo original, estos números entran en el conteo
- Haga un buen uso de la memoria
- **Para asignar el puntaje a esta pregunta es necesario que haya realizado un programa orientado a objetos. Es decir utilice clases y objetos.**

Listing 9: Ejemplo 4

```
Numero de elementos : 6

Elementos del arreglo
  34   37   28   16   44   36

La suma de todos los elementos es: 195
El mayor elementos es: 44
El menor elemento es : 16
En el arreglo hay 1 numeros primos
```

Listing 10: Ejemplo 4

```
Numero de elementos : 23

Elementos del arreglo
  34   37   28   16   44   36   37   43   50   22   13   28
    41   10   14   27   41   27   23   37   12   19
    18

La suma de todos los elementos es: 657
El mayor elementos es: 50
El menor elemento es : 10
En el arreglo hay 9 numeros primos
```

Listing 11: Ejemplo 4

```
Numero de elementos : 30

Elementos del arreglo
    34    37    28    16    44    36    37    43    50    22    13    28
        41    10    14    27    41    27    23    37    12    19
        18    30    33    31    13    24    18    36

La suma de todos los elementos es: 842
El mayor elementos es: 50
El menor elemento es : 10
En el arreglo hay 11 numeros primos
```

A continuación se muestra parte del código que Ud. debe completar, de tal manera que el programa realice lo que se pide.

Listing 12: Ejemplo

```
#include "CArreglo.h"
#include <ctime>
#include <cstdlib>

using namespace std;

int main()
{ int n;

    srand(time(nullptr));
    cout << "Numero de elementos: ";
    cin >> n;
    CArreglo a(n);
    a.imprimir();
    cout << "\n\n";
    cout << "La suma de todos los elementos es: " << a.suma()
        << "\n";
    cout << "El mayor elementos es: " << a.mayor() << "\n";
    cout << "El menor elemento es: " << a.menor() << "\n";
    cout << "En el arreglo hay " << a.contarPrimos() << "
        numeros primos\n";
    return 0;
}
```

Los criterios en la rúbrica (y el puntaje respectivo) se condicionan a que la solución presentada corresponda al problema planteado

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Ejecución	El diseño del algoritmo es ordenado y claro, siguiendo buenas prácticas en programación. La ejecución es correcta (3pts)	El diseño del algoritmo es ordenado y claro. La ejecución es correcta (2pts)	El diseño tiene algunas deficiencias pero la ejecución es correcta (1pts).	El diseño es deficiente y la ejecución no es correcta (0.5pts)
Sintaxis	No existen errores sintácticos o de compilación (2pts)	Existen algunos errores sintácticos de menor relevancia, que no afectan el resultado (1.5pts).	Existen errores sintácticos en la forma de ejecución, que no afectan el resultado (1pts).	El código tiene errores de sintaxis que afectan el resultado (0.5pts)
Optimizacion	El código es óptimo y eficiente. De buen performance e interacción con el usuario (2pts)	El código es de buen performance durante la ejecución (1.5pts)	El código no está optimizado pero la ejecución no es deficiente(1pts)	El código no está optimizado y la ejecución es deficiente (0pts)