UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Programación III
Práctica Calificada 2
Pregrado
2023-1
Profesor: José A. Chávez Álvarez
Lab 1.01

Indicaciones específicas:

- Esta evaluación contiene 9 páginas (incluyendo esta página) con 4 preguntas. El total de puntos son 20.
- El tiempo límite para la evaluación es 100 minutos.
- Cada pregunta deberá ser respondida en un solo archivo con el número de la pregunta.
  - `p1.cpp`
  - `p2.cpp`
  - `p3.cpp`
  - `p4.cpp`
- Deberás subir estos archivos directamente a www.gradescope.com, uno en cada ejercicio. También puedes crear un .zip

Calificación:

Tabla de puntos (sólo para uso del professor)

Question	Points	Score
1	5	
2	5	
3	5	
4	5	
Total:	20	

1. (5 points) **Functors**

Crear el *Functor* `Transformer` el cual se encargue de realizar una transformación en una matriz de enteros. La transformación consiste en aplicar una operación matemática a cada elemento de la matriz y producir un nuevo valor. El functor debe admitir diferentes modos y especificar una `constante` para aplicar en la transformación.

El functor `Transformer` debe implementar la siguiente interfaz:

- **Constructor:** Recibe un `modo` de transformación (`SUMA`, `RESTA`, `MULTIPLICACION` y `DIVISION`) y un valor constante (`int`). La constante se utilizará en la operación de transformación.
- **Operador ():** Recibe un entero y devuelve el resultado de aplicar la transformación al entero según el modo y la constante especificados en el constructor.

En la función principal aplique a una matriz de 3×3 las siguientes transformaciones:

- `transoformer1`: Que sume el valor de 3 a cada elemento.
- `transoformer2`: Que reste el valor de 1 a cada elemento.
- `transoformer3`: Que multiplique el valor de 2 a cada elemento.
- `transoformer4`: Que divida entre 3 a cada elemento.

Implemente, además, la funcion `'modificar_matriz'` la cual aplique una determinada transformación a cada elemento de una matriz dinámica. Finalmente implemente la función `'print'` que imprima la matriz. El ejercicio debe ser validado con la siguiente función principal:

```
int main(){
    int** M = new int*[3];
    for(int i = 0; i < 3; i++)
        M[i] = new int[3];
    M[0][0] = 1; M[0][1] = 2; M[0][2] = 3;
    M[1][0] = 4; M[1][1] = 5; M[1][2] = 6;
    M[2][0] = 7; M[2][1] = 8; M[2][2] = 9;

    Transformer transformer1("SUMAR", 3);    // Paso 1
    modificar_matriz(M, transformer1);       // Paso 2
    print(M);                                // Paso 3

    // Implemente los tres pasos para cada transformacion

    for(int i = 0; i < 3; i++)
        delete[] M[i];
    delete[] M;
}
```

Salida:

```
 4   5   6
 7   8   9
10  11  12

 3   4   5
 6   7   8
 9  10  11

 6   8  10
12  14  16
18  20  22

 2   2   3
 4   4   5
 6   6   7
```

La rúbrica para esta pregunta es:

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Librería Estandar	Selección del contenedor de acuerdo con lo solicitado, uso adecuado de los iteradores, estructuras genéricas basados en contenedores. (5pts)	Selección del contenedor correcto, estructuras genéricas basados en contenedores. (4pts)	Selección del contenedor correcto, estructuras genéricas basados en contenedores, errores en el funcionamiento pasa algunas pruebas. (2pts).	No se selección ni el contenedor ni se desarrolló algoritmos y estructuras genéricas. (1pts)

2. (5 points) **Contenedores**

Implementar la funcion '`contarPares`' que cuente la cantidad de pares de elementos de un vector de enteros cuya diferencia sea igual a k. Los pares de elementos no se deben repetir. Y además, la función debe tener complejidad $\mathcal{O}(n)$. Cualquier otra solución obtendrá 0 puntos.

```
int main(){
    vector<int> vec = {1,5,3,2,4};
    int k = 2;
    int resultado = contarPares(vec, k);
    cout << "El resultado es: " << resultado << endl;
}
```

Salida:

```
El resultado es: 3  // 5-3,1-3,2-4
```

La rúbrica para esta pregunta es:

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Librería Estandar	Selección del contenedor de acuerdo con lo solicitado, uso adecuado de los iteradores, estructuras genéricas basados en contenedores. (5pts)	Selección del contenedor correcto, estructuras genéricas basados en contenedores. (4pts)	Selección del contenedor correcto, estructuras genéricas basados en contenedores, errores en el funcionamiento pasa algunas pruebas. (2pts).	No se selección ni el contenedor ni se desarrolló algoritmos y estructuras genéricas. (1pts)

3. (5 points) **Pregunta 3**

El siguiente algoritmo calcula el segundo elemento mínimo de una secuencia de números A. Es decir, si ordenamos la secuencia de menor a mayor, sin considerar las repeticiones, esta función calcularía el segundo elemento de la secuencia ordenada.

```
1.  ENCONTRAR-SEGUNDO-MINIMO(A)
2.  min1 = ∞
3.  min2 = ∞
4.  for  i = 0, ..., size(A) − 1
5.    if  A[i] < min1
6.      min2 = min1
7.      min1 = A[i]
8.    else if  A[i] < min2  and  A[i] ≠ min1
9.      min2 = A[i]
10. return  min2
```

Para este problema:

- Describa el Invariante de Bucle.
- Utilice el Invariante de Bucle para demostrar el algoritmo:
  - Inicialización: ¿El I.B. se cumple en la primera iteración?¿Como?
  - Mantenimiento: Asumiendo que el I.B. se cumple al comenzar la iteración i, ¿Este se mantiene al terminar la iteración?¿Como?
  - Terminación: Al terminar el bucle, utilice el I.B. para demostrar el algoritmo.
- Envíe su solución con el nombre de la pregunta, el formato puede ser `PDF` o imagen. No es necesario implementar el algoritmo en C++, pero puede hacerlo para verificarlo.

La rúbrica para esta pregunta es:

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Complejidad Algorítmica	Buen nivel de abstracción, el problema logro realizar con la complejidad algorítmica solicitado, funciona correctamente y sin errores. (`5pts`)	Buen nivel de abstracción, el problema logro realizar lo solicitado sin lograr alcanzar la complejidad algorítmica solicitado, funciona correctamente y sin errores. (`4pts`)	Programa no funciona adecuadamente, bajo nivel de abstracción, más de 3 errores, nivel de complejidad algorítmica incorrecta. (`2pts`)	Se intento pero no se logró que funcione lo solicitado. (`1pts`)

4. (5 points) **Pregunta 4**

La función "`findMax`" encuentra el máximo elemento en un vector de números enteros. Utilizando la librería `thread`, implemente otra función llamada "`findMaxParallel`" que realice el mismo trabajo pero con 4 hilos.

```
int findMax(const vector<int>& numbers) {
    return *max_element(numbers.begin(), numbers.end());
}

// ========================================================
// Implementar aqui la funcion 'findMaxParallel'


// ========================================================
int main() {
    vector<int> numbers = {9, 12, 4, 7, 1, 18, 3, 5, 10};

    int Max1 = findMax(numbers);
    cout << "Elemento␣Maximo:␣" << Max1 << endl;

    int Max2 = findMaxParallel(numbers);
    cout << "Elemento␣Maximo␣(Paralelo):␣" << Max2 << endl;
    return 0;
}
```

Salida:

```
Elemento Maximo: 18
Elemento Maximo (Paralelo): 18
```

La rúbrica para esta pregunta es:

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Programación Concurrente	Buen nivel de abstracción, el problema se desarrolla utilizando la cantidad de hilos solicitados, se controla adecuadamente los race condition, funciona correctamente y sin errores. (`5pts`)	Buen nivel de abstracción, el problema no se utiliza la cantidad de hilos solicitados, no se controla los race condition adecuadamente, funciona correctamente y sin errores. (`4pts`)	Programa no funciona, bajo nivel de abstracción, más de 3 errores visibles , no se usa los hilos adecuadamente ni un control de race condition. (`2pts`)	Contiene errores que no hace que funcione el programa. (`1pts`)