Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI-2611 - Algoritmos y Estructuras I

Ejercicios: Arreglos y Subprogramas en GCL [1]
C. Chang, Versión 1.2

Ejercicio 0: Iteración en arreglos e invariantes

Se tienen dos arreglos de números enteros, B y C ambos de dimensión $[0..N)$, ambos ordenados de forma creciente. Se desea llenar el arreglo r de dimensión [0 .. 2N), con los valores de B y C y que al hacerlo r también esté ordenado de forma creciente.

Por ejemplo: para B= [-2, -2, 1, 2, 3, 3] y C = [-1, 0, 1, 2, 100, 200], se tiene $N = 6$ y r=[-2, -2, -1, 0, 1, 1, 2, 2, 3, 3, 100, 200]

- **Escriba un programa correcto en GCL** que ordene B y C de forma creciente en el arreglo r.
- Anote la precondición y postcondición del programa, así como la cota y el invariante de cada ciclo. No realice demostraciones.

Ejercicio 1: Iteración en arreglos

Se tiene un arreglo b de N elementos de tipo entero, ordenado de forma creciente (no estrictamente). Implemente una función que sea una **versión iterativa del algoritmo de búsqueda binaria**, que cuente la cantidad de veces que un elemento aparece en el arreglo.
Por ejemplo para b = [-8, 2, 4, 5, 5, 5, 19] , la función retorna 3 al buscar el número 5; retorna 1 al buscar el número 2 y retorna 0 al buscar el número 1. La función debe realizar la búsqueda binaria del elemento y luego contar el número de ocurrencias del elemento en el arreglo ordenado.

- Indique la precondición y la postcondición de la función, así como la cota de los ciclos.
- No realice las demás anotaciones. No realice demostraciones.

Ejercicio 2: Corrección de la asignación en arreglos

Para cada una de las siguientes triples de Hoare, **demuestre formalmente** que es correcta o explique por qué es incorrecta, suponiendo que los elementos de los arreglos están bien definidos e inicializados.

- **i**. $\{c[0] = 2 \wedge c[1] = 1\} \quad c[c[0]] := c[0] - c[1] \quad \{c[2] \leq c[1] \leq c[0]\}$
- **ii** . $\{b[1] = 5\} \quad b[b[1]] := 3 \quad ; \quad b[1] := b[5] - b[1] \quad \{b[1] < 0\}$
- **iii**. $\{b[0] = 0 \wedge (\forall i \,|\, 0 \leq i \leq 5 : c[i] = i)\} \quad b[\, c[3] - 1] := 0 \quad \{b[0] + b[1].b[2] = 0\}$
- **iv** . $\{b[1] = 2\} \quad b[b[1]] := 3 \quad ; \quad b[1] := b[2] - b[1] \quad \{b[1] > 0\}$

Ejercicio 3 : Iteración en matrices. Matriz Pre-mágica

Dada una matriz A de números enteros, de $N \times M$, determine si la matriz es "pre-mágica". Una matriz es "pre-mágica"si tiene al menos una fila y una columna tales que la suma de sus elementos de el mismo resultado.

Por ejemplo, la matriz A es "pre-mágica"porque la fila 0 suma 15 y la columna 2 también.

$$A = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 11 & 10 & 9 & 8 & 7 & 6 \\ 12 & 13 & -3 & 15 & 16 & 17 \\ 23 & 22 & 0 & 20 & 19 & 18 \\ 24 & 0 & 7 & 27 & -1 & 4 \end{bmatrix}$$

- Utilice análisis descendente para estructurar su código.
- Indique la precondición y postcondición de sus funciones o procedimientos, así como el invariante y la cota de las iteraciones.
- No realice las demás anotaciones. No realice demostraciones.

Ejercicio 4: Iteración en matrices. Filas Capicúa

Se tiene una matriz A de números enteros en el rango [0,9] de dimensiones $M \times N$. Se desea contar cuántas filas de la matriz son un número palíndromo o capicúa, esto es, que los valores de las casillas de la fila se lean igual de izquierda a derecha que de derecha a izquierda (ver figura 1, matriz izquierda). Adicionalmente, se desea sobreescribir cada casilla de las filas que no son números palíndromos por la suma de todos los valores de dicha fila, módulo 10 (ver figura 1, matriz derecha).

Escriba un programa correcto que resuelva el problema descrito. Utilice análisis descendente para proponer e implementar **al menos dos procedimientos o funciones** que ayuden a estructurar el programa principal. Indique los parámetros de dichos subprogramas y su tipo. Indique las precondición del programa principal y de los subprogramas. No indique invariantes, cotas ni postcondiciones. No demuestre correctitud.

$$A = \begin{bmatrix} 3 & 8 & 2 & 8 & 3 \\ 3 & 2 & 4 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 2 & 3 & 2 & 3 \end{bmatrix} \qquad A = \begin{bmatrix} 3 & 8 & 2 & 8 & 3 \\ 9 & 9 & 9 & 9 & 9 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Figura 1: Izquerda: Ejemplo de una matriz A de entrada. Las filas 0, 2, 3 forman números palíndromos. La cantidad de filas palíndromos de la matriz A es 3. Derecha: La matriz A al finalizar el programa.

Ejercicio 5: Iteración en matrices. Recorridos

Dada una matriz A de números enteros, de dimensiones $[0..M) \times [0..N)$, **se desea numerar cada casilla** comenzando en $A[M-1][0]$ y comenzando con el número 0 ($A[M-1][0] := 0$), **siguiendo estrictamente** el recorrido que se muestra en la figura 1 (tomando como ejemplo una matriz 5×6).

$$A = \begin{bmatrix} 4 & 5 & 14 & 15 & 24 & 25 \\ 3 & 6 & 13 & 16 & 23 & 26 \\ 2 & 7 & 12 & 17 & 22 & 27 \\ 1 & 8 & 11 & 18 & 21 & 28 \\ 0 & 9 & 10 & 19 & 20 & 29 \end{bmatrix}$$

Figura 2: Ejemplo de recorrido deseado y asignaciones de la matriz

Previamente un equipo de desarrolladores se planteó el problema de realizar el recorrido de la figura 3:

$$B = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 11 & 10 & 9 & 8 & 7 & 6 \\ 12 & 13 & 14 & 15 & 16 & 17 \\ 23 & 22 & 21 & 20 & 19 & 18 \\ 24 & 25 & 26 & 27 & 28 & 29 \end{bmatrix}$$

Figura 3: Ejemplo de recorrido y asignaciones de una matriz, para un problema anterior.

Para el problema conocido (figura 3) propusieron la siguiente postcondición del programa:

$\{Q : (\forall i, j \,|\, 0 \leq i < M \wedge 0 \leq j < N : (par.i \Rightarrow B[i][j] = N * i + j) \wedge (impar.i \Rightarrow B[i][j] = N * i + (N - j - 1)))\}$

Note que en general las postcondiciones solo reflejan el resultado del programa, no la manera en que el resultado es alcanzado.

- **Escriba un programa correcto en GCL** que recorra y numere las casillas de **la matriz** A siguiendo estrictamente el recorrido que se muestra en el ejemplo de la **figura 2**.
- Proponga el invariante para el ciclo interno de su programa. Para ello, se sugiere adaptar la postcondición del problema de la figura 3. No indique las demás aserciones. No realice demostraciones.

Ejercicio 6: Recorrido en Arreglos y Matrices, Procedimientos y Funciones.

Nota: Tome un tiempo para entender el enunciado. No hay restricciones sobre el uso de instrucciones, operaciones, variables, o tipos de datos de GCL.

Parte a. Procedimiento:

Programe correctamente el **procedimiento**:

move_in_The_Matrix(in-out i, j: int , in N, M: int, in TM: array [0..N)× [0.. M) of int, in p: bool)

tal que, posicionados en la fila i y en la columna j (con $N, M > 0$; $0 \leq i < N$ y $0 \leq j < M$) de la matriz TM, se determine una nueva posición en la matriz, de la siguiente manera:

Si p es true, cuando $TM[i-1][j] \leq TM[i][j]$ nos moveremos una fila hacia arriba en la matriz. En caso que $TM[i-1][j] > TM[i][j]$, nos moveremos una fila hacia abajo.

Si p es false, cuando $TM[i][j-1] \leq TM[i][j]$ nos moveremos una columna hacia la izquierda en la matriz. En caso que $TM[i][j-1] > TM[i][j]$ nos moveremos una columna a la derecha.

La matriz es circular, esto es, si $i = 0$, se considera que $i - 1 = N - 1$ y si $i = N - 1$, entonces $i + 1 = 0$. De forma similar, si $j = 0$, se considera que $j - 1 = M - 1$ y si $j = M - 1$, entonces $j + 1 = 0$. De esta manera, no es posible desplazarse a posiciones inexistentes de la matriz. Claramente, actualizar correctamente los valores de i y j es parte de lo que se debe programar en el procedimiento. En resumen, el procedimiento actulizará el valor de i o de j, dependiendo del valor de p y de ciertos valores de TM. Así, por ejemplo, dada la matriz:

$$Matrix = \begin{bmatrix} 1 & 2 & 2 & 3 & 4 & 5 \\ 9 & 10 & 9 & 18 & 7 & 21 \\ 12 & 13 & -3 & 15 & 36 & 17 \\ 23 & 22 & 0 & 20 & 19 & 18 \\ -24 & 0 & 7 & 27 & -1 & 4 \end{bmatrix}$$

Si $i = 2$ y $j = 3$, la llamada move_in_The_Matrix(i , j , 5, 6, $Matrix$, true), asigna a i el valor 3 (nos movemos una fila hacia abajo en la matriz), pues $Matrix[1][3] > Matrix[2][3]$ (18>15).

Si $i = 2$ y $j = 3$, la llamada move_in_The_Matrix(i , j , 5, 6, $Matrix$, false), asigna a j el valor 2 (nos movemos una columna a la izquierda), pues $Matrix[2][2] \leq Matrix[2][3]$ (-3 ≤15).

Si $i = 0$ y $j = 0$, la llamada move_in_The_Matrix(i , j , 5, 6, $Matrix$, true), asigna a i el valor 4 (nos movemos una fila hacia arriba en la matriz), pues $Matrix[4][0] \leq Matrix[0][0]$ ($0 - 1 = 4$ pues la matriz es circular, y $-24 \leq 1$).

Si $i = 2$ y $j = 5$, la llamada move_in_The_Matrix(i , j , 5, 6, $Matrix$, false), asigna a j el valor 0 (nos movemos una columna a la derecha, 5+1=0, la matriz es circular), pues $Matrix[2][4] > Matrix[2][5]$ (36 >17).

Tenga en cuenta que en cada llamada al procedimiento se modifica el valor de una sola variable (i o j) , dependiendo del valor del parámetro p. Al programar el procedimiento, no indique las aserciones.

Parte b. Programa Principal:

Nota: puede escribir el programa principal aunque no haya programado el procedimiento.

Se tiene una matriz $Matrix$ de números enteros, de F filas y C columnas, y un arreglo $pill$ de E elementos booleanos, con $F, C > 0$ y $E \geq 0$.

Escriba un programa correcto que comenzando en la posición $Matrix[0][0]$, llame E veces al procedimiento move_in_The_Matrix, con $pill[h]$ como su parámetro booleano (p), en la iteración h.

Por ejemplo, si $pill = [true, false, false]$ y la matriz $Matrix$ es la dada anteriormente, el programa principal hará 3 llamadas a move_in_The_Matrix(), la primera con i=0 y j=0, y parámetro booleano true, y las dos restantes con los valores actualizados de i y j, y el parámetro booleano false.

Terminado el ciclo, el programa revisa si algún elemento en la fila i o en la columna j es igual a 1, asignando a la variable booleana is_the_one verdadero si se encontró el valor 1, y asignándole el valor false si no se encontró. Puede suponer que existen una o varias **funciones** que realizan esta búsqueda; no se requiere programarlas, solo debe dar la definición de dichas funciones y hacer las llamadas en el programa principal.

Por ejemplo, si al terminar el ciclo se tiene que i=4 y j=0, entonces $is\_the\_one := true$ pues en la columna 0 hay un elemento cuyo valor es 1.

Ejercicio 7: Corrección de llamadas a procedimientos. Euclides

Dado el siguiente procedimiento:

proc euclides(in b, c :int ; out q, r: int)
{P: $b \geq 0 \wedge c > 0$ }
{Q: $b = q * c + r \ \wedge 0 \leq r < c$ }

Tenga en cuenta que los operadores **div** y **mod** se definen en base a esta descomposición de b:
b div $c = q$, y b mod $c = r$.

Si la siguiente tripleta de Hoare es correcta, demuéstrela formalmente. En caso de ser incorrecta, explique por qué:

$\{ \ q = 5 \wedge 0 \leq r < 3 \ \}$ euclides($q + r$, $q - r$, b, c) $\{ \ (b = 1 \vee b = 2) \wedge \ 0 \leq c < 3 \ \}$

Ejercicio 8: Corrección de llamadas a procedimientos. Productos Notables

Dado el siguiente procedimiento:

proc producto_notable_limitado(in a, b :int ; out z: int)
{P: $a \geq 0 \land b > a$ }
{Q: z $= a^2 + 2ab + b^2$ }

Si la siguiente tripleta de Hoare es correcta, demuéstrela formalmente. En caso de ser incorrecta, explique por qué:

$$\{\ 0 < b < 5 \land z = -1 \land c = 1\ \}\ \text{producto\_notable\_limitado}(b + z, b + c, b)\ \{\ 2^2 \leq b \leq 2^6\ \}$$