Proyecto 2

“Temperatura en una placa, estado estacionario”

Materia: Programación Básica

-Alumno: José Pablo Cuevas Cázares
-Fecha de entrega: 30/10/18
-Docente: Alma González

Descripción de la función main

El programa fue dividido en dos archivos diferentes: uno en donde declaré la función main, y otro en donde declaré una nueva función tipo void de resultados. Esto fue con el fin de cumplir los requisitos de aplicar el uso de funciones en el programa. En el archivo que contenía a la función main, declaré primero la función void de resultados, y le introduje como argumento el épsilon que daría el usuario, con el fin de no tener que escanearlo a partir de un archivo. Posterior a ello, ya dentro de la función main, se declararon variables flotantes (las temperaturas de cada lado de la placa) y épsilon1, la cual es la que da el usuario para poder tener una mejor aproximación de las temperaturas. También declaré una variable n, que representaba el número de puntos a lo largo de una línea de la placa, así como las variables “j” e “i”, las cuales se usaban en los for para asignar valores a cada punto de la placa. Después de haber declarado las variables, abrí dos archivos: uno en del cual se leían los datos para el programa (las temperaturas, n y épsilon). En este caso sí declaré épsilon para ser leída, como medio de respaldo por si no leía de manera correcta el argumento de la función resultados. Posteriormente, abrí el archivo de datos y escaneé los valores necesarios para hacer las operaciones. Una vez escaneados los datos a partir del archivo, cerré el mismo. A continuación, declaré n como n=n+2, con el fin de que tomara todas las dos columnas de la matriz. Una vez declarado n, ahora sí pude declarar por fin los arreglos de temperaturas. Es importante declarar primero el tamaño del arreglo y después el arreglo en sí, ya que si no se comete un error de segmentación. Una vez declarados los dos arreglos necesarios (uno para la temperatura en cada punto en la primera iteración, y otro que se convertiría en la temperatura “vieja”, es decir, que cuando se realizara otra iteración, se guardara el valor de la temperatura anterior en ese otro arreglo, cuyo propósito será explicado más adelante). Enseguida inicialicé todos los puntos de la placa en cero con dos ciclos for, esto con el fin de comenzar a hacer una evolución de la temperatura a partir de un caso ideal, en donde todos los puntos de la placa (a excepción de los de los lados) comienzan en cero. A continuación, declaré los valores de los puntos de cada lado, igualmente con dos for. Esto era necesario, ya que para que el programa calculara la temperatura en dos puntos tenía que comenzar con algo, puesto que no tenía mucho sentido

que calculara temperaturas cuando todas valen cero. Es por ellos que declaré los valores de la temperatura en cada lado, igualmente con el uso de la matriz, dando, por ejemplo un valor de cero a todas las "i" y un valor de n-1 a todas las jotas, y así para cada eje, con el fin de igualar todos los puntos de cada lado a una temperatura específica dada por el usuario.
Después de que a los puntos de las matrices les fueron dados valores específicos, se procedió a imprimir dichos valores en un archivo de texto, el cual fue previamente declarado en la misma función main. Después de ser imprimidos lo valores de la matriz de tamaño n*n, se cierra el archivo, y finalmente en el main se "llama" o inicializa la función resultados.

Descripción de la función resultados

La función de resultados es de tipo void, con un argumento dentro, el valor del épsilon dado por el usuario. Primero abrí dicha función con su correspondiente argumento, y dentro de ella procedí a declarar las variables necesarias para realizar las operaciones. Primero declaré una variable contador, a la que denominé como "z", cuyo objetivo era imprimir los datos en múltiples archivos, la cual inicialicé en cero. Después declaré otra variable de tipo entero, a la que denominé "epsilon3", la cual servía como un contador de operaciones, es decir, establecí una condición para que el programa se detuviera de imprimir archivos con base en el contador y "espilson3", ya que una funcionaba como un contador de archivos para imprimir, y la otra funcionaba como la cantidad de operaciones que se realizaban en dichos archivos. Después declaré "epsilon2" como un flotante, y una variable de tipo caracter a la que denominé "prenom" y finalmente declaré dos archivos, uno de lectura y otro de escritura.
Después de haber declarado las variables, abrí el archivo original de datos para leer "n" y lo cerré, ya que para las operaciones es necesario tener el tamaño de la matriz, y al igual que en el main, redefiní n como n=n+2. A continuación de haber declarado n, declaré como flotantes las dos matrices, las de temperatura en cada instante y las de temperatura vieja, de un tamaño n*n. Posteriormente, abrí el archivo en el cual guardé los valores de las matrices en la función main, es decir, los valores de las temperaturas en cada punto (que son cero, exceptuando los de las orillas, que son los que el usuario definió en el archivo datos). A continuación, una vez ya obtenidos los valores de las matrices se comenzaron a realizar las operaciones necesarias. Comencé con un do while, ya que decidí que esto sería lo mejor, puesto que las temperaturas necesitaban ser ajustadas a cada iteración con base en el épsilon, que también éste se iba ajustando en cada iteración, por lo que me pareció favorable que el programa hiciera esto cálculos mientras se cumplieran determinadas condiciones. Son precisamente estas condiciones las que permiten al programa calcular los valores de la temperatura de manera precisa, y esto también recalca la importancia de darle un valor a épsilon de acuerdo a la exactitud con la que se quiere trabajar. Entonces, se realizaron todas las operaciones para calcular la temperatura en cada punto siempre que el

épsilon que se calculaba asociada a cada temperatura fuera mayor que el épsilon dado por el usuario, y también que "epsilon3" fuera menor a un valor arbitrario, que como ya mencioné, es el número de veces que realiza las operaciones. Dentro del do, en el do while, primero calculé la temperatura en un punto dado, con el uso de dos ciclos for, los cuales comenzaban en uno, eran menores a n-1 e iban avanzando de uno en uno. Este n-1 que aparece constantemente es necesario, ya que de ser siempre n sería el tamaño completo de la placa, y esto no es deseable, puesto que sabemos que todos los puntos de las orillas tienen un punto específico, por lo que es necesario calcular la temperatura de todos los puntos, exceptuando por supuesto los de las orillas. En este ciclo utilicé la siguiente ecuación para calcular la temperatura en un punto arbitrario:

$$T_{i,j} = \frac{T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1}}{4} \qquad (1)$$

La ecuación anterior fue usada para calcular la temperatura en un punto de la placa, con base en las temperaturas de los puntos vecinos. La siguiente ecuación fue utilizada para realizar un ajuste en dicha temperatura, y poder obtener así una mejor aproximación a la temperatura real.

$$T_{i,j}^{\text{new}} = \lambda T_{i,j}^{\text{new}} + (1 - \lambda) T_{i,j}^{\text{old}} \qquad (2)$$

En la cual lambda representa un valor arbitrario entre 1 y 2, al cual yo decidí darle un valor de 1.5. Después de ello, inicialicé la variable "epsilon2" en cero, ya que ésta iría adquiriendo un nuevo valor conforme cada iteración, y en la primera era cero.
Una vez declarado el valor inicial de "epsilon2", utilicé la siguiente ecuación para calcular "epsilon2"

$$|(\varepsilon_a)_{i,j}| = \left| \frac{T_{i,j}^{\text{new}} - T_{i,j}^{\text{old}}}{T_{i,j}^{\text{new}}} \right| 100\% \qquad (3)$$

Después creé una condición if, en la que usaba la ecuación anterior, y si el valor de esta era mayor a "epsilon2" entonces le asignaba dicho valor a "epsilon2", lo cual sucedía hasta que (en la condición del while) "epsilon2" fuera mayor a "epsilon1".

También de gran importancia, después de la condición, pero aún dentro de los ciclos for, se le asignaba el valor de la matriz de la temperatura vieja a la temperatura que acababa de ser calculada en dicha iteración, con el fin de poder calcular el parámetro "epsilon2". A continuación, se cerraron los for y comenzó la parte de impresión de los resultados. Para ello utilicé un sprintf, escribí dentro de su argumento, al igual que cuando se daban argumentos de funciones y estos se imprimían en archivos (los ejemplos hechos en clase), el nombre de la variable char previamente declarada, después una variable entera (para que le asignara un número a cada archivo impreso) y después el contador z, el cual es esta variable que acompaña a cada archivo. Enseguida creé un condicional if, en el cual establecía que si y solo si "epsilon2" era mayor que "epsilon1" y que el residuo entre z (el contador) y cinco fuera cero (para así solo imprimir los archivos múltiplos de cinco, porque si no, dependiendo del tamaño de la matriz dado por el usuario, se imprimirían una cantidad de archivos demasiado grande, no eficiente e indeseable), para después abrir un archivo, con el argumento de la variable caracter y que dicho archivo fuera de tipo writing. Si estas condiciones no se cumplían, significa que la temperatura alcanzó un balance, y como un extra decidí imprimir dichas temperaturas en un solo archivo. Una vez cerrado el condicional, pero aún dentro del archivo, inicialicé dos for para imprimir, virtualmente idénticos a los utilizados para imprimir los valores de las orillas y asignación de todos los puntos a cero (en la función main, cuando antes de realizar las operaciones). Enseguida cerré el archivo y agregué que el contador fuera ahora el mismo pero con una unidad agregada, esto con el fin de ir alterando el valor del contador de uno en uno, y lo dejé precisamente fuera del archivo para que imprimiera muchos archivos, y no solamente uno. Finalmente, al cerrar el paréntesis del "do", escribí la condición while anteriormente mencionada, la cual establecía que se haría todo lo anteriormente mencionado si y solo si el valor de "epsilon2" fuera mayor que el valor de "epsilon1" y que "epsilon3" fuera menor a un valor arbitrario (en este caso lo designé como 3000, para asegurarme de que el número de operaciones realizadas fuera el suficiente).

Análisis de resultados

En la primera iteración, se obtuvo la siguiente imagen:

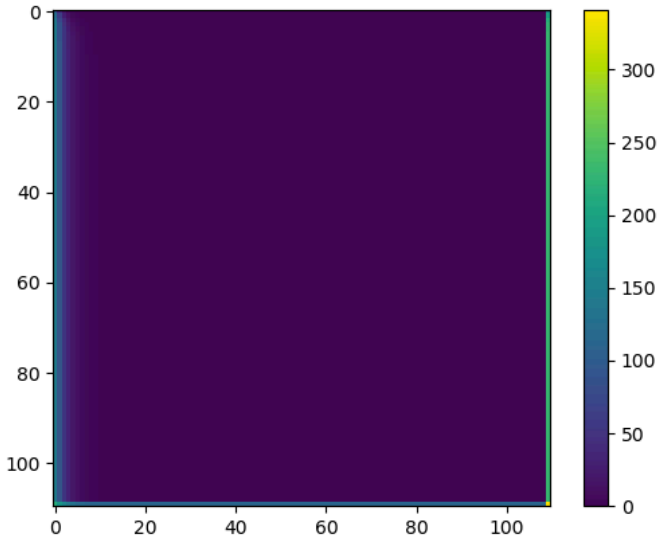

Figura1. En la imagen ser observan las temperaturas en cada punto cuando se ha realizado solamente una iteración

En la primera imagen se observa como la temperatura a lo largo de la placa, que evidentemente es uniforme, puesto que no se ha realizado mas que una iteración.
A continuación, se muestra gráfica de temperatura cuando se han realizado diez iteraciones:

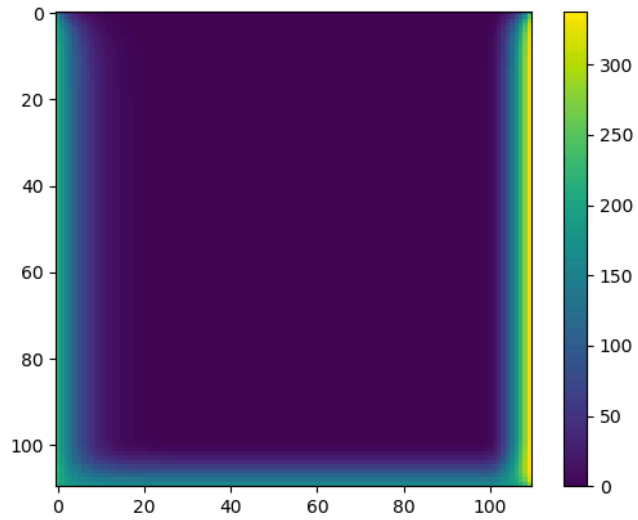

Figura2. En la imagen ser observan las temperaturas en cada punto cuando de la placa durante la décima iteración.

Como se puede observar, solamente después de la décima iteración y se notan cambios en cuanto a la uniformidad térmica de la placa, y como era de esperarse (por la manera como se distribuye el calor), los puntos más cercanos al lado más caliente se calientan más rápido y adquieren una mayor temperatura. Finalmente, se tiene la gráfica en la cuadrigentésima quinta iteración:

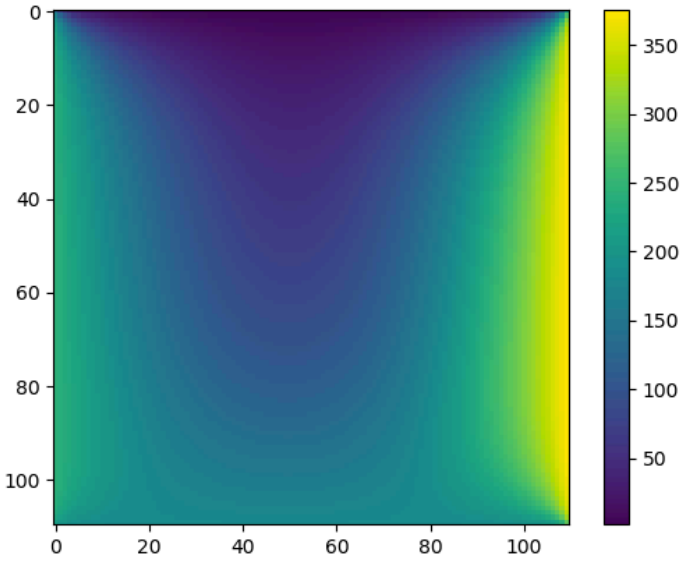

Figura3. En la imagen ser observan las temperaturas en cada punto cuando de la placa durante la cuadrigentésima quinta y última iteración.

Finalmente, se puede observar una distribución uniforme de las temperaturas, lo cual es lo esperado, ya que la placa, después de un cierto periodo de tiempo, tenderá a homogenizarse, al menos en las cercanías de ciertas regiones. Esto se ejemplifica al analizar las temperaturas cercanas a la máxima, las cuales se asemejan, así como al analizar las temperaturas cercanas a la mínima, que tienen aún un color morado. Esto se debe también al tamaño de la matriz, puesto que, entre menos puntos tenga la matriz más fácilmente fluirá el calor a través de ella, y entre más, lógicamente tardará más y se tendrán que realizar más iteraciones parar lograr una completa homogenización de la placa