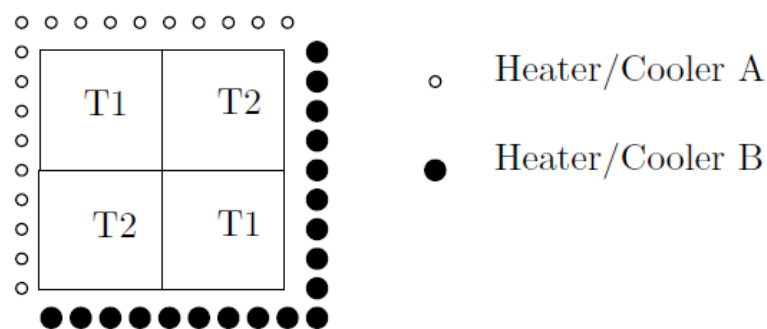


## Purpose

The purpose of this assignment is to have you work with two-dimensional arrays in the context of a simulation.

## Scenario

The scenario is a heat-flow simulation of the transfer of heat from two sides of a metal plate across the entire plate. A metal plate, as shown below, begins at a pair of fixed temperature. Heater/cooler A is placed up against the top and left (north and west) sides of the plate. Heater/cooler B is placed up against the bottom and right (south and east) sides of the plate. The plate itself starts at two initial temperatures: T1 and T2, as shown below.



For example, the plate might start at T1=10 degrees and at T2=35 degrees, the first heater/cooler might be at 80 degrees and the second heater/cooler might be at 60 degrees. As the simulation starts the sides of the plate will begin to heat up and thus take on temperatures closer to the heaters. This transfer of heat will continue across the plate until it reaches a “stable” temperature, where stable is defined, roughly, as “not changing significantly”.

## Method

To implement this simulation, we will use a finite state element approximation, that is, we will model the plate with a 2-D array. Only the plate is represented by the 2-D array; thus the heater/coolers are outside (but exactly adjacent to) the array. The plate will be represented by an (8x6) array: 8 rows and 6 columns, thus 48 elements. The temperature of all the plate elements will be initially set to the two initial plate temperatures.

To simulate the transfer of heat, you will traverse the entire array over and over. On each traversal, you will visit each element in your array and compute the new temperature in the following fashion: To determine the new temperature of a specific element, take the average of its four neighbors (north, east, south, west) from the previous cycle. Be careful, if you are updating an element on an edge of the plate, then you will have only THREE plate values and ONE "outside" value, where an "outside" value is the temperature of one of the heater/coolers. And a corner has two "outside" values. After you complete each traversal you should print out the two 2-D array of updated temperatures. Then replace the values of the old temperature by the new one for the next cycle. The simulation of traversals continues until no element in the array changes by a significant relative amount, where the significant amount is determined by the value of the last input item (see below). A relative amount means a percent change. Thus a value of .01 signifies a 1% change; it does not mean a change of .01 degree.

## Data Structure

To store the simulated transfer of heat, you will store the heat temperatures in a pair of 2-D (8x6) array of doubles, one for the previous temperature values and the other for the new ones being computed. You can assume that we are always working with an 8x6 array (row and column sizes are constants in the header file).

## Input

All input values will come from an input file. User supplies the name of the file when prompted.

The input includes 5 real (double) values:

The first heater/cooler temperature (A)

The second heater/cooler temperature (B)

The first initial plate temperature (T1)

The second initial temperature (T2)

The stabilizing criterion value

The stabilizing criterion value is a value to indicate WHEN to stop the simulation. The halting condition is defined as: “If NO element in the array changes by a relative amount MORE THAN the stabilizing criterion then the simulation can stop”. (This means that the temperature in the plate is settling down to its final temperature and continued traversals will only slightly change the temperature distribution). See the posted files for examples of input.

## Output

The output will first echo the input data then display the heat-plate temperatures. You should display each plate as the temperatures change until the halt criterion is satisfied. Be sure to look at the posted output examples and follow them closely. Make sure that you have the correct number of iterations.

You can send output to the screen – no need to print the output to a file.

## Details

- Remember that the stabilization criterion is a relative change, not a degree change. Relative change is defined as:

$$|t_o - t_n|/t_o$$

where  $t_o$  is the old temperature and  $t_n$  is the new temperature

- You must use at least the following functions (all of them must be commented):
  - A function called `read_data()` will read the input values from the input file (file name supplied by user input).
  - A function named `print_plate()` that will display the temperatures as in the examples. You can assume that the temperature values will be in the range 0 – 99.99 and your display should look good for all temperatures in this range. Your display should look similar to my sample output even if it's not exactly the same. Display temperatures with 2 decimal places.
  - A function called `initialize_plate()` to initialize the plate
  - A function called `compute_plate()` that does one update of the plate temperatures
- The program must be split into multiple files:
  - The main must be the only function in one code file (.c) called `heatflow.c`
  - All the other functions should be in a second code file (.c) called `heatflow_functions.c`

- The other functions should be forward declared (prototype declarations) in a header file (.h) and included in both code files (the header file is given to you, don't make any changes to it). This file is called heatflow.h
- Submit files heatflow.c and heatflow\_functions.c

## Mimir Unit Tests

- There are several unit tests in Mimir that check to see how your compute\_plate() function works. As given by the header file compute\_plate() has the following signature:  
*int compute\_plate( double A[][COL] , double B[][COL],double h1, double h2,double stab);*
- Always make sure that the array A has the new values computed, and B contains the previous cycle values. In other words at the beginning (or end) of the function make sure to copy the value of A into B for the computation that follows after. (performing this copy from A to B in your main function will cause several tests to fail since it's expected that compute\_plate() will do this operation.