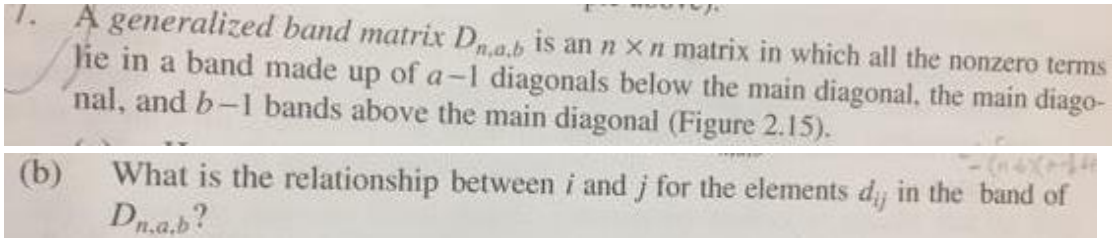
Data Structure Assignment 2

Paper Homework

(Textbook p.101 Exercises 7(b))



7. A *generalized band matrix* $D_{n,a,b}$ is an $n \times n$ matrix in which all the nonzero terms lie in a band made up of $a-1$ diagonals below the main diagonal, the main diagonal, and $b-1$ bands above the main diagonal (Figure 2.15).

(b) What is the relationship between i and j for the elements d_{ij} in the band of $D_{n,a,b}$?

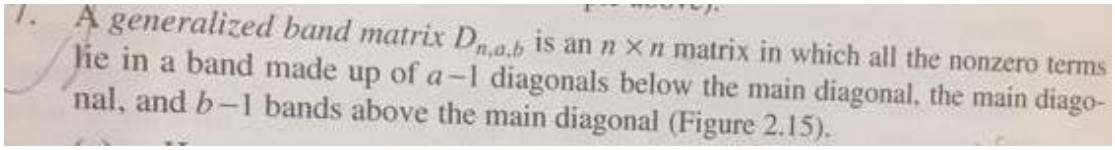
General Information:

- Deadline：2016/11/2 (Please submit to TA after class)
- Late homework will not be accepted.
- Please write on A4 papers.
- Notice：You won't get any point if you only write the answer, please list your process and reason.
- Any copies will be scored as zero. Do not plagiarize

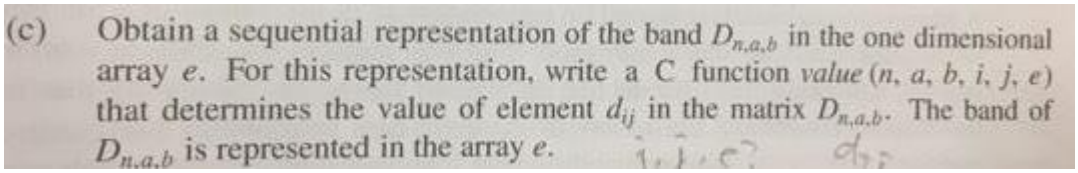
Data Structure Assignment 2

Programming Homework1

(Textbook p.101 Exercises 7(c))



7. A *generalized band matrix* $D_{n,a,b}$ is an $n \times n$ matrix in which all the nonzero terms lie in a band made up of $a-1$ diagonals below the main diagonal, the main diagonal, and $b-1$ bands above the main diagonal (Figure 2.15).



(c) Obtain a sequential representation of the band $D_{n,a,b}$ in the one dimensional array e. For this representation, write a C function *value* (n, a, b, i, j, e) that determines the value of element d_{ij} in the matrix $D_{n,a,b}$. The band of $D_{n,a,b}$ is represented in the array e.

此程式之 input 需以讀檔的方式輸入，詳見 sample input 和 read file introduction

Sample Input:

Please input the file name: Sample_input.txt
a: 3
b: 3

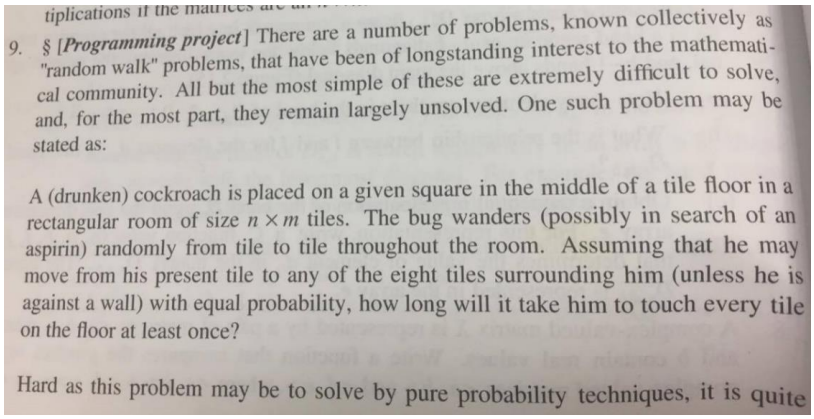
Sample Output:

e[0] = 9 d(2,0)
e[1] = 7 d(3,1)
e[2] = 8 d(1,0)
e[3] = 3 d(2,1)
e[4] = 6 d(3,2)
e[5] = 6 d(0,0)
e[6] = 0 d(1,1)
e[7] = 2 d(2,2)
e[8] = 8 d(3,3)
e[9] = 7 d(0,1)
e[10] = 4 d(1,2)
e[11] = 9 d(2,3)
e[12] = 8 d(0,2)
e[13] = 4 d(1,3)

Data Structure Assignment 2

Programming Homework2

(Textbook p.102 Exercises 9)

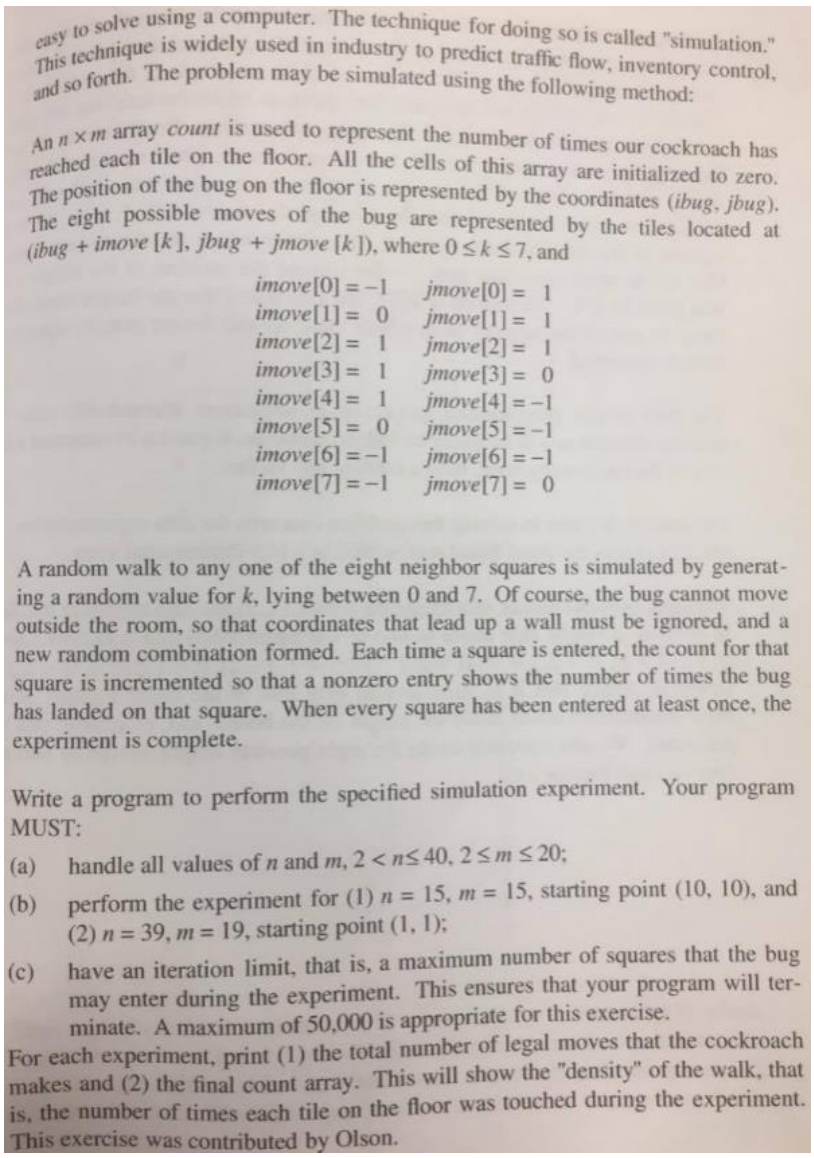


tiplications if the matrices are an

9. § [*Programming project*] There are a number of problems, known collectively as "random walk" problems, that have been of longstanding interest to the mathematical community. All but the most simple of these are extremely difficult to solve, and, for the most part, they remain largely unsolved. One such problem may be stated as:

A (drunken) cockroach is placed on a given square in the middle of a tile floor in a rectangular room of size $n \times m$ tiles. The bug wanders (possibly in search of an aspirin) randomly from tile to tile throughout the room. Assuming that he may move from his present tile to any of the eight tiles surrounding him (unless he is against a wall) with equal probability, how long will it take him to touch every tile on the floor at least once?

Hard as this problem may be to solve by pure probability techniques, it is quite



easy to solve using a computer. The technique for doing so is called "simulation." This technique is widely used in industry to predict traffic flow, inventory control, and so forth. The problem may be simulated using the following method:

An $n \times m$ array *count* is used to represent the number of times our cockroach has reached each tile on the floor. All the cells of this array are initialized to zero. The position of the bug on the floor is represented by the coordinates (*ibug*, *jbug*). The eight possible moves of the bug are represented by the tiles located at (*ibug* + *imove* [*k*], *jbug* + *jmove* [*k*]), where $0 \le k \le 7$, and

imove[0] = −1	*jmove*[0] = 1
imove[1] = 0	*jmove*[1] = 1
imove[2] = 1	*jmove*[2] = 1
imove[3] = 1	*jmove*[3] = 0
imove[4] = 1	*jmove*[4] = −1
imove[5] = 0	*jmove*[5] = −1
imove[6] = −1	*jmove*[6] = −1
imove[7] = −1	*jmove*[7] = 0

A random walk to any one of the eight neighbor squares is simulated by generating a random value for *k*, lying between 0 and 7. Of course, the bug cannot move outside the room, so that coordinates that lead up a wall must be ignored, and a new random combination formed. Each time a square is entered, the count for that square is incremented so that a nonzero entry shows the number of times the bug has landed on that square. When every square has been entered at least once, the experiment is complete.

Write a program to perform the specified simulation experiment. Your program MUST:

(a) handle all values of n and m, $2 < n \le 40$, $2 \le m \le 20$;

(b) perform the experiment for (1) $n = 15$, $m = 15$, starting point (10, 10), and (2) $n = 39$, $m = 19$, starting point (1, 1);

(c) have an iteration limit, that is, a maximum number of squares that the bug may enter during the experiment. This ensures that your program will terminate. A maximum of 50,000 is appropriate for this exercise.

For each experiment, print (1) the total number of legal moves that the cockroach makes and (2) the final count array. This will show the "density" of the walk, that is, the number of times each tile on the floor was touched during the experiment. This exercise was contributed by Olson.

Input:

能任意輸入陣列 n*m 的值及 initial bug position

範圍限制：$2 < n \leq 40,\ 2 < m \leq 20$

Example

N: 2

M: 2

Initial bug position X: 0

Initial bug position Y: 0

Output:

1. The total number of legal moves that the cockroach makes

2. The final count array

註：請以陣列的方式輸出到 txt 檔顯示答案，並且陣列各項必須對齊

Sample_Output.txt

General Information:

- Deadline：2016/11/2 23:55.
- Upload your assignment to Moodle system.
- Upload file format: Student-Id_Name.rar , Ex.P76991094_王小明.rar
- Your file should consist of the following items: Source Code & Readme file (Program description)
- Late homework will not be accepted.
- Any copies will be scored as zero. Do not plagiarize