ON CONSTRUCTION OF MAGIC RECTANGLES

SANDIP BARUI

GUIDE: DR. ASHISH DAS

(PROFESSOR AND FACULTY ADVISOR, DEPARTMENT OF MATHEMATICS, IIT BOMBAY)

CO-GUIDE: DR. MURALI K. SRINIVASAN

(H.O.D, MATHEMATICS DEPARTMENT, IIT BOMBAY)

ROLL NO. 08528005

M.SC. SECOND YEAR PROJECT

COURSE: APPLIED STATISTICS AND INFORMATICS

DEPARTMENT OF MATHEMATICS

INDIAN INSTITUTE OF TECHNOLOGY, BOMBAY

ACKNOWLEDGEMENT:

Magic Rectangles are well-known for their very interesting and entertaining combinatorics. The main motivation of the project "***On Magic Squares and Magic Rectangles***" pertains to its huge applicability in the fields of Design of Experiments. It has illuminated many fields like solving Rubic's cube problem and finds its application in the elimination of trend effects in certain classes of one-way, factorial, Latin-square, and Graeco-Latin-square designs. As highly balanced structures , magic rectangles can be potential tools for the use in situations yet unexplored.

The encouragement to work on the project is greatly attributed to my Guide ***Dr.Ashish Das*** and my Co-Guide ***Dr. Murali.K.Srinivasan***. The entire planning of my work has been supervised by them . It's really a great opportunity for me to have them throughout my project. Every minor details and trifles regarding the project have been dealt with great precision by them. Even if they could not be present with me everytime, they have always helped me via e-mails, telephones etc. The materials they have provided me are undoubtedly the best ones in this area.

My colleagues and my friends have also helped me with various codes and methods regarding the implementation of the techniques in C++.

Truly, without their help and support this project would not have been successful.

<u>CONTENTS</u>:

1. INTRODUCTION

Since remote times magic configurations of integers have been of interest to mathematicians. The magic cubes, rectangles, spheres, pencils, crosses, squares etc. The best known and most interesting of these is the magic square and rectangle which still merits attention. Its most obvious- its magic property-is that the sum of all elements in each row and each column are the same.

We found the magic square and rectangle a simple problem with a very rich Combinatorics: there are $n^2!$ ways to fill **nxn** matrix with integers between 1 to n^2 , without repetitions, but only very few of them are magic squares. Similarly we can also fill an **mxn** matrix with integers between 1 to **mn**, without repetitions but only a few are magic rectangles. For **n=4**, generating all 16! permutations, we found **7040** magic squares and **549504** relaxed magic squares. Many such magic rectangles and squares have been used in designing experiments.

For example, **Phillips (1964,1968a,1986b)** illustrated the use of these magic figures for the elimination of trend effects in certain classes of one-way , factorial, Latin-square and Graeco-Latin square designs. Magic rectangles are a generalization of magic squares that have been investigated recently by **Bier** and **Rogers (European J. Combin. 14(1993) 285-299)**; and **Bier** and **Kleinschmidt (Discrete Math. 176(1997) 29-42)**. As highly balanced structures, magic rectangles can be the potential tools for use in situation yet unexplored.

From a paper: Construction and use of magic squares and magic rectangles (by J. P. De Los Reyes, Ahmad Pourdarvish, Chand K. Midha & Ashish Das) We have the following example:

Consider a contractor who has taken a job that has three components of activities. The job is required to be completed in 3 periods of time, say, day, month or the like. The contractor has fixed a given total number of man hours for the job. These man hours have to be so distributed over the 3 components in different periods that in each of the 3 periods of work equal total numbers of man-hours are used.
We use a 3 × 3 magic square shown below to provide a distribution of man-hours in each of the 3 periods so as to satisfy the above requirements. The elements in the magic square are the man-hours, the rows denote the job components (JC) and the columns denote the periods (P).
The 3×3 magic square ***M*0** with 9 elements from 0 to 8 as man-hours is presented below.

Periods	**P1**	**P2**	**P3**	**Total**
JC1	7	0	5	12
JC2	2	4	6	12
JC3	3	8	1	12
Total	12	12	12	36

The total of 36 man-hours are distributed equally over the levels of both the job components and periods. The number 0 in Job comp. 1 and Period 2 indicates that no man hour is employed in this component-period combination. As the restriction for equality of diagonal totals is not there, each of the 3 rows as a whole can be shifted over the components suitably. Similarly, each of the columns as a whole can also be suitably shifted over the periods. Furthermore, the total man-

hours and the individual man-hours can be transformed suitably, as needed, by transforming the elements using the relation $a+bi$ where i stands for the man-hours in the above square and a and b are suitable constants to determine the given total man-hours and also the individual man-hours in the square.
Apart from these, several types of orthogonal Latin squares have been suggested by **Denes** and **Keedwell** (1974). Such orthogonal squares have been used to construct magic squares.

Magic squares of order N with consecutive and distinct numbers from $\mathbf{0}$ to $\boldsymbol{N^2-1}$ is called a proper magic square. It is sometimes a limitation to have such squares and rectangles with consecutive integers starting from 0. To increase the scope of their use it is necessary to allow them to have varying numbers. This would help to apply these magic figures for practical work problems requiring suitable distributions of sets of varying, or even specified, numbers in form of such arrangements. The final aim could be that given a set of numbers how to arrange them in form of a magic square or rectangle. Variation among a set of numbers arranged in form of magic squares or rectangles can be achieved through a linear transformation. The final aim could be that given a set of numbers how to arrange them in form of a magic square or rectangle.
A magic square is an arrangement of the integers 1 to n^2 in a double dimensional array of n rows and n columns, where the sum of all elements of the rows, of the columns and of the diagonals are equal and are equal to what is called a magic sum. The total of n^2 numbers is $n^2 (n^2+1)/2$ the sum of each row and column will be $n(n^2+1)/2$ and that will also be the diagonal sum as well.

A magic rectangle is an arrangement of the integers 1 to mn in an array of m rows and n columns so that each row adds to the same total M and each column to the same total N. The totals M and N are termed the row magic constant and column magic constant. Since the average value of the integers is A = (mn + 1)/2, we must have M = nA and N = mA. The total of all the integers in the array is mnA = mM = nN. A magic rectangle or square may be one of the two kinds-even by even or odd by odd. If mn is even mn + 1 is odd and so for M = n(mn + 1)/2 and N = n(mn +1)/2 to be integers m and n must both be even. On the other hand, since either m or n being even would result in the product mn to be even, therefore if mn is odd then m and n must both be odd. In this case also M and N are integers since mn + 1 is even. Therefore, an odd by even magic rectangle is not possible. (***reference [4]&[6]&[7]&[9]***)

A simple construction method for any even by even magic rectangle was recently provided by **De los Reyes, Das, Midha and Vellaisamy (2009)**. The construction of an odd by odd magic rectangle is even more difficult.

The Project mainly involves the detailed procedure for the construction of Magic Rectangle of different orders by studying their properties as the construction directly follows from their existence, uniqueness etc. The construction involves an elaborate knowledge and theories of discrete mathematics.

The construction has been implemented using C++ programming in Linux. The coding part mainly concerned with the construction of magic rectangles and squares (with the exception that square diagonal element does not add up to the magic constant). It is being broadly classified into three groups- odd by odd magic rectangle, even by even magic rectangle and nearly magic rectangles. Other than construction the coding also provides several queries that needed by user to check whether the given rectangle is a magic rectangle or not. For instance, the program immediately outputs “Please provide smaller values as the no. of rows and columns.” in case the user enters the number of rows or number of columns to be large, so that the capacity of memory allocation gets exceeded. The main extension of the stage I is that of constructing nearly magic rectangles and creating webpage, so that the program can be executed online and users from any part of the world.

The code is embedded in HTML using PHP. PHP stands for "PHP: Hypertext Preprocessor". PHP is a scripting language through which you can generate web pages dynamically. PHP code is directly inserted in HTML documents through specific TAGs declaring the code presence and then executed when a user demands the page. PHP is a server-side language, that's to say that PHP code is directly executed by the server, while the user receives processed results as an HTML document. This way of working is different from that of other scripting languages as JavaScript, whose code is first loaded onto the user machine and then executed by the user (the browser).

A general code for embedding a program in PHP goes like:

```
<HTML>
<BODY>
<?

echo("<H1>Hello World!</H1>");

?>
</BODY>
</HTML>
```

If the code is written in notepad and if uploaded on some browser then a page would be created with the text “Hello World”.

So, this procedure can utilized to create a webpage in which the program for constructing magic rectangle can be executed online and desired result can be displayed as output, and so, the required magic rectangle of various orders can be obtained from anywhere in the world.

The project mainly involves the construction of magic rectangles and nearly magic rectangles of different orders and so we are mainly going to focus on the construction part.

2. CONSTRUCTION OF ODD ORDERED AND EVEN ORDERED MAGIC RECTANGLE:

Here, we consider magic rectangle with m rows and n columns where m and n both odd. The construction is being divided into four different functions separately.

1. *Construction of odd ordered magic rectangles. (reference [1])*

A. The foremost procedure involves m and n where m=n=p(say). Here, matrices i.e. double dimensional arrays named N1 and N2 are taken.

$$N1 = \begin{pmatrix} 1 & 2 & 3 \ldots\ldots & p-1 & p \\ 2 & 3 & 4 \ldots\ldots & p & 1 \\ : & : & : \ldots\ldots & : & : \\ p & 1 & 2 \ldots\ldots & p-2 & p-1 \end{pmatrix} \qquad N2= \begin{pmatrix} p & p-1 \ldots\ldots 2 & 1 \\ 1 & p \ldots\ldots 3 & 2 \\ : & : \ldots\ldots : & : \\ p-1 & p-2 \ldots\ldots 1 & p \end{pmatrix}$$

We can notice that N1 and N2 are orthogonal latin squares. The construction states that the magic rectangle **M** is given by $\mathbf{M=N1+(N2-J_{pp})*p}$ is a magic rectangle where $\mathbf{J_{pp} = ((1))_{pxp}}$.

A pair of latin squares $\mathbf{A=(a_{ij})}$ and $\mathbf{B=(b_{ij})}$ are *orthogonal* iff the ordered pairs $\mathbf{(a_{ij},b_{ij})}$ are distinct for all i and j. Here are a pair of orthogonal latin squares of order 3.

1 2 3	**1 2 3**	**1 1 2 2 3 3**
2 3 1	**3 1 2**	**2 3 3 1 1 2**
3 1 2	**2 3 1**	**3 2 1 3 2 1**
A	**B**	**A and B superimposed**

A and B are clearly latin squares and, when superimposed, you can see that all ordered pairs from corresponding square entries are distinct.

As a part of the coding, it involved input of two arrays(double-dimensional) with the property that the (i,j)th element of N1 is (i+j-1) and that of N2 is (p+i-j) initially. A check is being done that if the (i,j)th element is greater than p, then we change it to i+j-1-p for N1. For N2, we deduct (i,j)th element by p. Then the resultant matrix M is being printed as an output where **M(i,j)=N1(i,j)+p*(N2(i,j)-1).**

This part is being kept as a separate function named "process0".

B. The second function deals with the magic rectangle of odd order with the column number (q), not a multiple of 3 and the no. of rows is p.

For this, we define or rather construct matrices **R,S** and **T(**which are being constructed in every other parts also**).**

$$R= \begin{pmatrix} 1 & 2 \ldots\ldots q-1 & q \\ q & q-1 \ldots\ldots 2 & 1 \end{pmatrix} = \begin{pmatrix} R_U \\ R_S \end{pmatrix} \quad , \; S= \begin{pmatrix} 1 & 2 \ldots\ldots q' & q'+1 \ldots\ldots q \\ q' & q'+1 \ldots q & 1 \ldots\ldots q'-1 \\ q & q-2 \ldots 1 & q-1 \ldots\ldots 2 \end{pmatrix}$$

$$T= \begin{pmatrix} q & q-2 & \ldots & 1 & q-1 & q-3 & \ldots & 2 \\ 1 & 2 & \ldots & q' & q'+1 & q'+2 & \ldots & q \\ 1 & 3 & \ldots & q & 2 & 4 & \ldots & q-1 \\ q & q-1 & \ldots & q' & q'-1 & q'-2 & \ldots & 1 \end{pmatrix} = \begin{pmatrix} T_U \\ T_L \end{pmatrix}$$

where **q'=q+1/2;**

(Here, each of T_U and T_L are of order 2×q.) Then the rows of R, S and T are permutations of (1, 2, 3, . . . , q) and the column sums of R, S and T are q + 1, 3(q + 1)/2 and 2(q + 1) respectively..

After having the above construction of the matrices being done, matrix $\mathbf{G_p}$ is defined.

The construction involved four conditions:

i) **P=3 => G_3 =S.**

ii) **P=5 => G_5=** $\begin{pmatrix} R_U \\ S \\ R_L \end{pmatrix}$

iii) **p>5 ; if p'=(p-3)/4 is an integer , then G_p=** $\begin{pmatrix} 1_{p'} \otimes T_U \\ S \\ 1_{p'} \otimes T_L \end{pmatrix}$

If p'= (p-5)/4 is an integer , then G_p = $\begin{pmatrix} R_U \\ G_{p-2} \\ R_L \end{pmatrix}$

Then, for p -3, Gp is a p × q matrix with rows being permutations of (1, 2, 3, . . . , q) and every column sum being equal to p(q + 1)/2.

Here,⊗ implies Kronecker product between matrices.

Let **A=** $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ **B=** $\begin{pmatrix} e & f \\ g & h \end{pmatrix}$ **A ⊗ B=** $\begin{pmatrix} ae & be & af & bf \\ ce & de & cf & df \\ ag & bg & ah & bh \\ cg & dg & ch & dh \end{pmatrix}$

i.e. A⊗ B = $\begin{pmatrix} Ae & Af \\ Ag & Ah \end{pmatrix}$

This can be implemented easily by taking the range of the outer loop from 1 to pr and that of the inner loop from 1 to qs, where pxq and rxs are the orders of the two matrices. Then, the required matrix can be obtained from this by printing the elements in every such pxq blocks of the larger matrix of order pr x qs, along with the each element of the matrix of order rxs being multiplied to each of these block.

Every time the 4 conditions on p being implemented, we check that whether p' is an integer. For this we check whether p-3 mod 4=0 or p -5 mod 4=0 using the in -built mod operator in the libraries.

To generate the matrix G_p when p= 4p'+5, we construct rather call the function for the matrix G with suffix p-2 at first, store the matrix into an array and then attach to it, the first row having R_U

and last row having R_L . In this way the construction of G matrix is done.

From the G matrix , the matrix H can be created very easily. The matrix $\mathbf{H_p = ((h_{ij}))}$ where

$\mathbf{h_{ij} = (g_{ij}-1)p + i, 1 \leq i \leq p}$ and $\mathbf{1 \leq j \leq q}$ and $\mathbf{G_p = ((g_{ij})).}$

Then the $\mathbf{p \times q}$ matrix **Hp** has the following properties:
(i) each of the numbers 1 through pq appear once,
(ii) each column sum equals $\mathbf{p(pq+1)/2.}$
(iii) the i-th row sum is $\mathbf{pq(q-1)/2 + qi,}$
(iv) for $\mathbf{1 \leq i \leq (p-1)/2}$, the difference between the **(p+1−i)**-th row sum and the i-th row sum is $\mathbf{q(p + 1 - 2i),}$
(v) the **(p+1/2)**-th row sum is $\mathbf{q(pq+1)/2}$.

Once the matrix H is created the next step is to construct the magic rectangle involving various conditions. The construction basically evolved following a theorem, which states that
"For a given odd integers p and q (p<q) , there exists a pxq magic rectangle M."
In this function we attempt to construct magic rectangles which have a column number q, that is does not have a factor 3.It involves various interchange of positions of the matrix elements:

We define **l=(q-p)/2;**

i) If (p-5)/4 = floor((p-5)/4) i.e we check whether (p-5)/4= int((p-5)/4) and there exists an integer y such that y=(l-1)/2 or y=(l-2)/2 then we interchange the following:

- $\mathbf{h_{1,j}}$ with $\mathbf{h_{p,j}}$
  $\mathbf{h_{1,q+1-j}}$ with $\mathbf{h_{p,q+1-j}}$; if $1 \leq \mathbf{j \leq y,\ y>0.}$
- $\mathbf{h_{1,(2q+3-p/4)}}$ with $\mathbf{h_{p,\ (2q+3-p/4)}}$ **; if y=(l-1)/2**
- $\mathbf{h_{1,(2q+3-p/4)}}$ with $\mathbf{h_{p,\ (2q+3-p/4)}}$
  $\mathbf{h_{1,(q+1/2)}}$ with $\mathbf{h_{p,\ (q+1/2)}}$ **; if y=(l-2)/2.**

ii) If y be an integer such that either y=(l-1)/2 or y=(l-2)/2 then we exchange the elements of H matrix in the following way in the i-th and (p+1-i)-th row for all the i in the range given as $\mathbf{((p-1)/2-2*floor((p-3)/4) \le i \le p-3/2)}$:

- $\mathbf{h}_{i,(q+3)/2-j}$ with $\mathbf{h}_{p+1-j,(q+3)/2-j}$
  $\mathbf{h}_{i,(q+1)/2+j}$ with $\mathbf{h}_{p+1-j,(q+1)/2+j}$; if $1 \le \mathbf{j} \le \mathbf{y}$, **y>0.**

- $\mathbf{h}_{i,(p+3)/2-i}$ with $\mathbf{h}_{p+1-i,(p+3)/2-i}$ **; if y=(l-1)/2.**

- $\mathbf{h}_{i,1}$ with $\mathbf{h}_{p+1-i,1}$
  $\mathbf{h}_{i,(p+3)/2-i}$ with $\mathbf{h}_{p+1-i,(p+3)/2-i}$ **; if y=(l-2)/2.**

iii) If (q+1)/6 = floor((q+1)/6) and there exists an integer y such that y=(l-i)/4 for i=1,2,3,4 then we interchange the element of the matrix H in the following way in the (p-1)/2-th and (p+3)/2-th rows :

- $\mathbf{h}_{(p-1)/2,j}$ with $\mathbf{h}_{(p+3)/2,j}$
  $\mathbf{h}_{(p-1)/2,(q+1)/6+j}$ with $\mathbf{h}_{(p+3)/2,(q+1)/6+j}$
  $\mathbf{h}_{(p-1)/2,(q+3)/2-j}$ with $\mathbf{h}_{(p+3)/2,(q+3)/2-j}$
  $\mathbf{h}_{(p-1)/2,(q+1)-j}$ with $\mathbf{h}_{(p+3)/2,(q+1)-j}$; if $1 \le \mathbf{j} \le \mathbf{y}$, **y>0.**

- $\mathbf{h}_{(p-1)/2,(q+1)/3}$ with $\mathbf{h}_{(p+3)/2,(q+1)/3}$ **; if y=(l-1)/4.**

- $\mathbf{h}_{(p-1)/2,(q+1)/3}$ with $\mathbf{h}_{(p+3)/2,(q+1)/3}$
  $\mathbf{h}_{(p-1)/2,2(q+1)/3}$ with $\mathbf{h}_{(p+3)/2,2(q+1)/3}$ **; if y=(l-2)/4.**

- $\mathbf{h}_{(p-1)/2,(q+1)/3}$ with $\mathbf{h}_{(p+3)/2,(q+1)/3}$
  $\mathbf{h}_{(p-1)/2,(2q-1)/3}$ with $\mathbf{h}_{(p+3)/2,(2q-1)/3}$
  $\mathbf{h}_{(p-1)/2,(2q+5)/3}$ with $\mathbf{h}_{(p+3)/2,(2q+5)/3}$ **; if y=(l-3)/4.**

- $\mathbf{h}_{(p-1)/2,(q+1)/3}$ with $\mathbf{h}_{(p+3)/2,(q+1)/3}$
  $\mathbf{h}_{(p-1)/2,(2q-1)/3}$ with $\mathbf{h}_{(p+3)/2,(2q-1)/3}$
  $\mathbf{h}_{(p-1)/2,(2q+5)/3}$ with $\mathbf{h}_{(p+3)/2,(2q+5)/3}$
  $\mathbf{h}_{(p-1)/2,2(q+1)/3}$ with $\mathbf{h}_{(p+3)/2,2(q+1)/3}$ **; if y=(l-4)/4.**

iv) If (q-1)/6 = floor((q-1)/6) and there exists an integer y such that y=(l-i)/4 for i=1,2,3,4 then we interchange the element of the matrix H in the following way in the (p-1)/2-th and (p+3)/2-th rows :

- $\mathbf{h}_{\mathbf{(p-1)/2,j}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,j}}$
  $\mathbf{h}_{\mathbf{(p-1)/2,(q+1)/2+j}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (q+1)/2+j}}$
  $\mathbf{h}_{\mathbf{(p-1)/2,(5q+7)/6-j}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (5q+7)/6-j}}$
  $\mathbf{h}_{\mathbf{(p-1)/2,(q+1)-j}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (q+1)-j}}$ **;** if $1 \leq \mathbf{j} \leq \mathbf{y}$, $\mathbf{y>0}$.

- $\mathbf{h}_{\mathbf{(p-1)/2,(2q+1)/3}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,(2q+1)/3}}$ **; if y=(l-1)/4.**

- $\mathbf{h}_{\mathbf{(p-1)/2,(2q+1)/3}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (2q+1)/3}}$
  $\mathbf{h}_{\mathbf{(p-1)/2,(q+2)/3}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (q+2)/3}}$ **; if y=(l-2)/4.**

- $\mathbf{h}_{\mathbf{(p-1)/2,(2q+1)/3}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (2q+1)/3}}$
  $\mathbf{h}_{\mathbf{(p-1)/2,(q-1)/3}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (q-1)/3}}$
  $\mathbf{h}_{\mathbf{(p-1)/2,(q+5)/3}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (q+5)/3}}$ **; if y=(l-3)/4.**

- $\mathbf{h}_{\mathbf{(p-1)/2,(2q+1)/3}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (2q+1)/3}}$
  $\mathbf{h}_{\mathbf{(p-1)/2,(q-1)/3}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (q-1)/3}}$
  $\mathbf{h}_{\mathbf{(p-1)/2,(q+5)/3}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (q+5)/3}}$
  $\mathbf{h}_{\mathbf{(p-1)/2,(q+2)/3}}$ with $\mathbf{h}_{\mathbf{(p+3)/2,\ (q+2)/3}}$ **; if y=(l-4)/4.**

After these processes of interchanges, we rename the matrix H as M.
Then the p × q matrix M has the following properties,
(i) each of the numbers 1 through pq appear once,
(ii) each row sum equals q(pq+1)/2 ,
(iii) each column sum equals p(pq+1)/2 .

Thus M is a magic rectangle.
This entire process of construction is being called through the function “process”. The M matrix is printed at last with the check that its row sum and column sum being equal to q(pq+1)/2 and p(pq+1)/2 respectively. The floor criteria is being checked by taking the ‘int’ of the required expression and comparing with the ‘float’ of that.

C. The third function is concerned about the magic rectangle(odd ordered) whose column number q is a multiple of 3,whereas the row number is not a multiple of 3.

It basically follows from a theorem which states:

- **Suppose p, q and n are odd. Let $\mathbf{M_1}$ be a p×n magic rectangle. Then there exists p × qn magic rectangle M.**

Proof.
For $\mathbf{1 \leq i \leq p, 1 \leq j \leq q}$, let $\mathbf{M_j = M_1 + (j - 1)*pn*J_{pn}}$ and $\mathbf{m_{ij}}$ represent the i-th row of $\mathbf{M_j}$. Then **M** is obtained from $\mathbf{G_p}$ by replacing the element j in the i-th row of $\mathbf{G_p}$ by $\mathbf{m_{ij}}$, $\mathbf{1 \leq i \leq p, 1 \leq j \leq q}$.

Using the fact that $\mathbf{M_1}$ is $\mathbf{p \times n}$ magic rectangle and using the properties of the matrix $\mathbf{G_p}$, it is easy to see that each row sum of **M** equals:

$$\mathbf{(n(pn+1)/2\,)q + npn + n2pn + \cdots + n(q-1)pn = qn(pqn+1)/2}$$

and each column sum of **M** equals:

$$\mathbf{p(pn+1)/2 + (\,p(q+1)/2 - p)pn = p(pqn+1)/2}\ .$$

Hence the constructed **M** is a $\mathbf{p \times nq}$ magic rectangle.

If q is a multiple of 3 , then for a positive integer **k** , we have **q=3(2k+1).** At first in this part we start with the construction of magic rectangle of order px3, which can be constructed from a magic rectangle of order 3xp by taking its transpose .This can be obtained by calling the function 'process' with the arguments (3,p). We store the resultant matrix in a matrix array M_1.

Here, the process is not so simple and as a result a three dimensional matrix has to be defined for the construction of the matrices $\mathbf{M_j}$ where $\mathbf{1 \le j \le q}$. The three dimensions are –one for the row identification, one for the column identification and the last one for the index of the series of matrices that has to be created using the result $\mathbf{M_j = M_1 + (j-1)*pn*J_{pn}}$.

Here m_{ij} denotes the i-th row of M_j. We are to replace the element j in the i-th row of G_p by $m_{ij.}$

This task cannot be implemented directly. However, what we can do is to take each of the elements of the 3-D array into a linear array.

Suppose $\mathbf{A1} = \begin{pmatrix} \mathbf{a}_{111} & \mathbf{a}_{112} \\ \mathbf{a}_{121} & \mathbf{a}_{122} \end{pmatrix}$ $\mathbf{A2} = \begin{pmatrix} \mathbf{a}_{211} & \mathbf{a}_{212} \\ \mathbf{a}_{221} & \mathbf{a}_{222} \end{pmatrix}$

then $\mathbf{L} = (\mathbf{l_1, l_2, \ldots\ldots\ldots l_8})$ (In general we will have up to l_{pqr}) where $\mathbf{l_1 = a_{111}}$,$\mathbf{l_2 = a_{112}}$,$\mathbf{l_3 = a_{121}}$,$\mathbf{l_4 = a_{122}}$,$\mathbf{l_5 = a_{211}}$,$\mathbf{l_6 = a_{212}}$,$\mathbf{l_7 = a_{221}}$,$\mathbf{l_8 = a_{222}}$.

It can be held using a general formula as:

$$\mathbf{l_i = a_{4*(i-1)+2(j-1)+k}}$$

If we have m different matrices each having n rows and c columns then

$$\mathbf{l_i = a_{(mc)*(i-1)+n*(j-1)+k}}$$

Then we convert this linear array L into a 2-D array B of order 2x4 such that $\mathbf{b_{ij} = l_{(j+4(i-1))}}$ where $\mathbf{1 \le i \le 2}$ **and** $\mathbf{1 \le j \le 4}$.

In general we will do $\mathbf{b_{ij} = l_{(j+(i-1)*q)}}$ where q is the no. of columns.

Thus, what is being done is that we are replacing the element j of i-th row of G_p by m_{ij} and m_{ij} is the i-th row of M_j. So, we have constructed M_{ijk} where i represent the index of the matrix, j represents row number and k represents the column number of the matrices. Then applying the above technique we convert into M_{ij} and printed the corresponding matrix.

The entire process is done in function named '**processp7q21**' to represent that p is not a multiple of 3 whereas q is a multiple of 3.

D. The fourth part of the program deals with odd ordered magic rectangle, with number of rows(p) and columns(q) both having a factor 3.

e.g. suppose p=9 and q=27 which both have a factor 3 in common. We ,at first, construct magic rectangle M1 of order 3x3 by calling the function "process0" and storing the values in a matrix. Hence we can find a magic rectangle M2 of order 3x9 easily following the algorithm defined in "processp7q21" (since matrix 3x3 exists we can construct a matrix of order 3x3*3=3x9). We take the transpose of the matrix M2 and it becomes of order 9x3.Say we name it as M3. Now, we have a magic rectangle of order 9x3 , then following the same theorem as stated in part 3 , we can construct a magic rectangle M of order 9x27 as 27 = 3x9.

Basically ,in this part, we assemble the previous parts and rearrange them. This process is being called in function "**processp3q3**".

2. *Construction of even by even magic rectangle. (reference [2]&[3])*

So far we have seen the construction of magic rectangle having odd orders. In this case, we will deal with the construction of magic rectangle having even orders i.e. even number of rows and even number of columns. The construction involves some simple matrix operations. The method has been shaped in form of an algorithm that is very convenient for writing a computer program for constructing such rectangles. Furthermore, it is also presented in a ready-to-write form since the magic rectangles of lower orders are embedded in a magic rectangle of higher order.The construction of even ordered magic rectangles can be classified into two larger subgroups which involve two different method.

i) The Kronecker-partition matrix technique:

For any given integers p and q we define the following matrices (p<=q).

$$Qe= \begin{pmatrix} 0 & 2 \\ 3 & 1 \end{pmatrix} \qquad Qa= \begin{pmatrix} 2 & 3 \\ 1 & 0 \end{pmatrix} \qquad Qb= \begin{pmatrix} 0 & 1 \\ 3 & 2 \end{pmatrix}$$

$$\text{and } Q = \begin{pmatrix} 0 & 3 & 1 & 2 \\ 3 & 0 & 2 & 1 \end{pmatrix}$$

A matrix A is also being defined containing sequence of elements from 0 to pq-1.

$$A = \begin{pmatrix} 0 & 1 & 2 & \ldots\ldots. & q-1 \\ q & q+1 & q+2 & \ldots\ldots. & 2q-1 \\ : & : & : & & : \\ (p-1)q & (p-1)q+1 & (p-1)q+2 & \ldots\ldots. & pq-1 \end{pmatrix}$$

Let $\mathbf{1_t}$ represents a tx1 column vector of all ones and so $\mathbf{1_t=(1\ \ 1\ \ \ldots..\ \ 1)'}$.

Let $\mathbf{S_t'= (0\ \ 1\ \ 2\ \ \ldots\ldots..\ \ t-1)}$ be a row vector of order t with elements being the sequence of numbers from 0 to t-1.

Let $\mathbf{I_t}$ be an identity matrix of order t .

Moreover, $\mathbf{K_t}$ a square matrix of order t given by,

$\mathbf{K_t = ((k_{ij}))}$ **with** $\mathbf{k_{ij} = 1}$ **if** $\mathbf{i + j = t + 1}$ **,**
$\mathbf{= 0}$ **, o.w.**

All the above Matrices are defined and constructed following the above mentioned specifications.
The matrix A can be expressed in the following way too:

$$\mathbf{A= 1_p \otimes s_q' + 1_q' \otimes qs_p}$$

where $\otimes$ implies kronecker product between two matrices.

A matrix X is now being created, where

$$\mathbf{X= A \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} + A^*K_q \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix}}$$

The next job is to define two matrices A and B such that their sum can be represented as the required magic rectangle.

Here, according to the construction B matrix is a 2px2q matrix with the property :

$$\mathbf{B = 1_p \otimes (Q_e | Q_a | 1'_{q\text{-}3/2} \otimes (Q_b | Q_a) | Q_e) \quad ; \text{for } q \text{ odd;}}$$
$$\mathbf{= (1_p 1'_{q/2}) \otimes Q \quad ; \text{for } q \text{ even:}}$$

The construction of the above matrix involved storing the sub-matrices together in a larger matrix. We store elements of $\mathbf{(Q_b | Q_a)}$ in a matrix say Q1,then we store $\mathbf{1'_{q\text{-}3/2} \otimes Q1}$ in a matrix say Q2,($\mathbf{Q_e | Q_a | Q2 | Q_e)}$ in Q3 and finally ($1_p \otimes$ Q3) in B for odd q.
It can be noted that , for q = 3 , B reduces to $\mathbf{1_p \otimes (Qe | Qa | Qe)}$. Here, $B1_{2q} = 3q1_{2p}$ and $1'_{2p}B = 3p1'_{2q}$, i.e., the row sums of B are 3q and the column sums are equal to 3p .
The matrix C is constructed according to the definition as :

$$\mathbf{C = (Y_1 | (Kp \otimes I_2) Y_2);}$$

Where Y1 and Y2 are the partition of a matrix Y(of order 2px2q) and

$$\mathbf{Y= 4 (X \otimes (1 \ \ 1)) + 1 \otimes (1 \ 1 \ \ldots.. \ 1)^{1x2q}} \begin{pmatrix} \mathbf{1} \\ \mathbf{:} \\ \mathbf{1} \end{pmatrix}_{\mathbf{2px1}}$$

And **Y=(Y1|Y2)** where each of **Y1** and **Y2** are of order **2pxq**.

Again, it can be shown that the row sums of C are 2q(2pq - 1) and column sums are equal to 2p(2pq-1) . The proof is not required for the construction. So, R = B+C consists of the 4pq distinct numbers from 1 through 4pq ,it is a 2px2q magic rectangle with magic constants M = q(4pq +1) and N = p(4pq +1) .

ii) Serpentine matrix method:

An alternative way for creating a matrix of even order can be constructed using serpentine matrix method. We construct magic rectangle of sides m = 2p and n = 2q for given positive integers p and q . We consider separately the cases (i) at least one of p , q is even, and (ii) both p and q are odd.

Case I. At least one of p and q is even.

We can consider without loss of generality that p can be even. It is because even if q is even we can take the transpose of the matrix and make p (i.e. row number) to be even. Here, a check is being performed which checks whether p/2 is an integer or q/2 ia an integer and transpose of the matrix is taken if q is even.

After the check a matrix of the given form S is constructed.

$$S = \begin{pmatrix} 1 & 2m & 2m+1 & \ldots.. & nm \\ 2 & 2m-1 & 2m+2 & \ldots.. & nm-1 \\ 3 & 2m-2 & 2m+3 & \ldots.. & nm-2 \\ : & : & : & & : \\ m & m+1 & 3m & \ldots.. & (n-1)m+1 \end{pmatrix}$$

Here, s_{ij}**=m*j-i+1 if j is even.**

=m*j+i if j is odd; Where S=((s_{ij}**)).**

The matrix S is called serpentine form of matrix. To construct magic rectangle M from S we simply interchange middle p rows where p=m/2 i.e. we interchange m/4+i th row with 3m/4-i+1 th row where i=1,2,…..,m/4. This can be implemented easily by swap( ) fuction.

Case II. Both p and q are odd.

Without loss of generality, if either p or q equals 1, let q = 1 .We note that the case p = q = 1 does not arise since it amounts to a 2x2 magic rectangle which is impossible.

The steps involved in the construction are:

- The serpentine matrix S is being created like previous case with order mxn.
- The first q-1 columns are reversed. It means that q-j+1-th column will occupy j th row, j=1,2,…..,q-1.We name the matrix as S1.
- We reverse first p rows of the resultant matrix S1=(($s1_{ij}$)) except the middle two elements in each p rows i.e. $\mathbf{s1_{p-i+1,j}}$ becomes $\mathbf{s1_{ij}}$ where i=1,2,…..,p and j=1,2,…,(n/2)-1,(n/2)+2,……,n. The resultant matrix is S2.
- The corresponding middle p-3 elements of the q th and q+1th columns of S2 are interchanged i.e. the elements $\mathbf{s2_{iq}}$ and $\mathbf{s2_{i(q+1)}}$ are interchanged for all i=(m/2)-1,………,(m/2)+p-3.
- Also, we interchange the element in the {(1+(p-3)/2),q} th position with the element in the {(1+(p-3)/2) q+1} -th position and interchange the element in the {3+(p-3)/2, q} -th position with the element in the {3 + (p - 3)/2, q+1} -th position.

The resultant matrix obtained after above interchanges results in a magic rectangle of even order. In this part, the order of the matrix ,that is entered by the user, is being divided by 2 and then taken as the argument of the function in order to keep the order of the rectangle intact.
The entire process of constructing the even order rectangle is done using the function call "even". With this we end the external function calls outside the main.

3.CONSTRUCTION OF NEARLY MAGIC RECTANGLE:

(reference [11])

A nearly magic rectangle is a magic rectangle having some additional properties. Suppose p is even and q is odd. A matrix $\mathbf{N_{pxq} = ((n_{lh})), 1<=l<=p, 1<=h<=q}$, with order **pxq** is called a nearly magic rectangle if

- **(i)** the natural number 1 to **pq** appears exactly once in **Npxq**;
- **(ii)** $\mathbf{\sum_l n_{hl}=p(1+pq)/2}$ for $\mathbf{l<=h<= q}$ and $\mathbf{1<=l<=p}$,
- **(iii)** $\mathbf{\sum_h n_{hl}=(q(1+pq)-1\ )/2}$ for $\mathbf{l=1,3,5,....,p-1}$ and $\mathbf{\sum_h n_{hl}=(q(1+pq)+1)/2}$, for $\mathbf{l=2,4,....,p}$ where $\mathbf{1<=h<=q}$.

The construction of a nearly magic rectangle $\mathbf{M_{pxq}}$ involves several fuctions which take into account the following:

- p=2 and q is even and q>2.
- p=2 and q is odd, q>1.
- p is even and q odd but p/2 is odd.
- p is even and q odd but p/2 is even.

The last two functions takes care of the fact that if q is even and p is odd it swaps the value of p and q, runs the entire procedure and prints the transpose of the obtained nearly magic rectangle.

A. If p=2 and q is even and q>2.

The function named "**p2qeven( )**" is employed for this task.

First a matrix M is constructed using two dimensional array where

$$\mathbf{M} = \begin{pmatrix} \mathbf{M_1} & \mathbf{M_2} & \mathbf{M_3} & \mathbf{M_4} \\ \mathbf{M_8} & \mathbf{M_7} & \mathbf{M_6} & \mathbf{M_5} \end{pmatrix}$$

where $\mathbf{M_i = [(i\text{-}1)q1+1, (i\text{-}1)q1+2, ..., iq1], 1<=i<=4}$ and $\mathbf{M_i = [iq1, (i\text{-}1)q1+q1\text{-}1, ..., (i\text{-}1)q1+1]}$, for $\mathbf{5<=i<=8}$ if q1 is an integer such that $\mathbf{q1=q/4}$. Now the record for the matrices $\mathbf{M_i}$ s are being kept using a two dimensional array **M'[][]** where the first index indicates the suffix **i**, second index indicates the position of the element i.e. the $\mathbf{j^{th}}$ element of the vector. As a result it is quite easy to rearrange the $\mathbf{M_i}$ s of **M** such that

$$\mathbf{M} = \begin{bmatrix} \mathbf{M_8\ M_2\ M_3\ M_5} \\ \mathbf{M_1\ M_7\ M_6\ M_4} \end{bmatrix}$$ which is indeed a magic rectangle.

In general, **M(i,j) = (i-1)*q1+ j if 1<=i<=4**

= (i-1)*q1+q1-j+1 if 5<=i<=8 and j=1(1)q1.

This can be verified from the fact that each row has row sum=$\mathbf{(2q^2+q)/2}$ and each column sum is **(2q+1)**.For the case q1 is integer and q1 = (q-6)/4, then two sub cases are considered using if-then condition which are q1=0 and q1 >=1.

When q1=0 then M has order 2x6 and

$$\mathbf{M} = \begin{bmatrix} \mathbf{12} & \mathbf{2} & \mathbf{10} & \mathbf{4} & \mathbf{5} & \mathbf{6} \\ \mathbf{1} & \mathbf{11} & \mathbf{3} & \mathbf{9} & \mathbf{8} & \mathbf{7} \end{bmatrix}$$

and when q1>=1 we construct two more two dimensional array A and B ; **A=((4q1+j))** and A is of order 1x6 ,**B=((4q1+j))** and it is of order 1x6, where j are elements of M of order 2x6 just explained above.For the matrix A, j s are the element of the first row of M of order 2x6 and for B, j s are the elements of the second row of M of order 2x6 . Ultimately, the magic rectangle of order 2xq is obtained as :

$$\mathbf{M} = \begin{bmatrix} \mathbf{M8} & \mathbf{M2} & \mathbf{M3} & \mathbf{M5} & \mathbf{A} \\ \mathbf{M1} & \mathbf{M7} & \mathbf{M6} & \mathbf{M4} & \mathbf{B} \end{bmatrix}$$

In general, **M(i,j) = (i-1)*q1+ j ; if 1<=i<=4**

= (i-1)*q1+12+q1-j+1 ;if 5<=i<=8 and j=1(1)q1.

Here also each row has row sum = $\mathbf{(2q^2+q)/2}$ and each column sum is **(2q+1)**.

B. If p=2 and q is odd and q>2.

The construction follows from a **lemma** which says:

There exists a pxq matrix A such that (i) each row is a permutation of [1,2,3,......,q] and (ii) the column sums are equal.

The function named "**p2qodd( )**" is employed for this task. First a matrix N is constructed as follows:

* ***<u>Case 1: q1 is integer and q1=(q-1)/4 where q1>=1</u>***.

The matrix $\mathbf{N_i}$ s are defined as **N[i] = [(i-1)q1+1, (i-1)+2,......, iq1]** for **1<=i<= 4** and **N[i] = [iq1+2,iq1+1,.......,(i-1)q1+3]** for 5<=i<=8. Like previous techniques a two dimensional matrix keeps the record of these $\mathbf{N_i}$ s. So a matrix **N** is thus constructed such that

$$\mathbf{N^*} = \begin{pmatrix} \mathbf{N_8} & \mathbf{N_2} & \mathbf{N_3} & \mathbf{N_4} & \mathbf{4q1+1} \\ \mathbf{N_1} & \mathbf{N_7} & \mathbf{N_6} & \mathbf{N_5} & \mathbf{4q1+2} \end{pmatrix}$$

where the first row sum is $\mathbf{(2q^2+q-1)/2}$ and the second row sum is $\mathbf{(2q^2+q+1)/2}$, and each column sum is **2q+1** .

$$\text{In general, } \mathbf{N(i,j)} = \begin{cases} \mathbf{(i-1)*q1+ j \,;} & \mathbf{if\ 1<=i<=4} \\ \mathbf{(i-1)*q1+q1-j+4 \,;} & \mathbf{if\ 5<=i<=8\ and\ j=1(1)q1.} \end{cases}$$

<u>*Case 2: q1 is integer and q1=(q-3)/4 where q1>=0</u>.

I) Case 2.1:**q1=0**, then we construct $\mathbf{N^*} = \begin{pmatrix} \mathbf{1} & \mathbf{5} & \mathbf{4} \\ \mathbf{6} & \mathbf{2} & \mathbf{3} \end{pmatrix}$

II) Case 2.1:**q1>=1**,

The matrix $\mathbf{N_i}$ s are defined as **N[i] = [(i-1)q1+1,(i-1)+2,......,iq1]** for **1<=i<= 4** and **N[i] = [(i-1)q1+6+q1,(i-1)q1+6+(q1-1),.......,(i-1)q1+6+1]** for 5<=i<=8. Like earlier a two dimensional matrix keeps the record of these **Ni** s. So a matrix N is thus constructed such that

$$\mathbf{N^*} = \begin{pmatrix} \mathbf{N8} & \mathbf{N2} & \mathbf{N3} & \mathbf{N4} & \mathbf{4q1+1} & \mathbf{4q1+5} & \mathbf{4q1+4} \\ \mathbf{N1} & \mathbf{N7} & \mathbf{N6} & \mathbf{N5} & \mathbf{4q1+6} & \mathbf{4q1+2} & \mathbf{4q1+3} \end{pmatrix}$$

where the first row sum is $\mathbf{(2q^2+q-1)/2}$ and the second row sum is $\mathbf{(2q^2+q+1)/2}$, and each column sum is 2q+1 .

$$\text{In general, } \mathbf{N(i,j)} = \begin{cases} \mathbf{(i-1)*q1+ j} & \mathbf{;\ if\ 1<=i<=4} \\ \mathbf{(i-1)*q1+6+q1-j+1} & \mathbf{;\ if\ 5<=i<=8\ and\ j=1(1)q1.} \end{cases}$$

<u>C. p is even and q odd but p/2 is odd.</u>

This procedure is being taken care of by the function called “**pby2oddqodd** ( )”.We define **p1=p/2** and **q*=(q+1)/2**.

In the beginning we construct a matrix $\mathbf{B_1}$ which of order **2xq** using the function **p2qodd( )** just explained earlier and matrices $\mathbf{B_i}$ s are constructed which is a three dimensional matrix as follows:

B[i][j][k] = B[1][j][k]+q*(i-1), for 2<=i<=p1

Here define

$$T_1 = \begin{pmatrix} 1 & 2 \ldots\ldots q^* & q^*+1 \ldots\ldots & q \\ q^* & q^*+1 \ldots\ldots q & 1 \ldots\ldots & q^*-1 \\ q & q-2 \ldots\ldots 1 & q-1 \ldots\ldots & 2 \end{pmatrix} = \begin{pmatrix} \rho_{11} \\ \rho_{12} \\ \rho_{13} \end{pmatrix}$$

$$T2 = \begin{pmatrix} 1 & 2 \ldots\ldots \ldots\ldots & q-1 & q \\ q & q-1 \ldots\ldots\ldots\ldots & 2 & 1 \end{pmatrix} = \begin{pmatrix} \rho_{21} \\ \rho_{22} \end{pmatrix}$$

If **p=3** then we construct $\mathbf{A} = [\,\rho'_{11}\ \rho'_{12}\ \rho'_{13}\,]'$ and if **p=2n+3, n>=1**, then let $\mathbf{A} = [\rho'_{211} \ldots \rho'_{21n}\ \rho'_{11}\ \rho'_{12}\ \rho'_{13}\ \rho'_{221}\ \rho'_{222} \ldots\ldots \rho'_{22n}]'$ with $\rho_{21\,i} = \rho_{21}$ and $\rho_{22\,i} = \rho_{22}$ for **1<= i<=n**.

Here, in $\mathbf{T_1}$ and $\mathbf{T_2}$, all the column sums are equal and each row is a permutation of **[1,2,.....,q]**.Thus we can create a matrix **A** of order **qxp1** as defined above.

Now, we allot the matrix **A** and store them using indices **A[i][j]** and construct a matrix **N= [B'1 B'2 ….. B'p1]'** = **((bij))** which is of order **2p1xq** and store the elements as follows:

$\gamma[j][k] = (\, b_{jk}\, ,\, b_{(j+1)k}\,)'$ where **1<=j<=p1** and **1<=k<=q**

We here define a matrix **N* = ((n*ij))** such that

$$n^*[i][j] = \gamma[A[i][j]\,]\,[j], \text{ where } 1<=i<=p1 \text{ and } 1<=j<=q.$$

Thus **N* =**

$$\begin{pmatrix} \gamma_{a(11,1)} & \gamma_{a(21,2)} \ldots\ldots\ldots\ldots & \gamma_{a(q1,q)} \\ \gamma_{a(12,1)} & \gamma_{a(22,2)} \ldots\ldots\ldots\ldots & \gamma_{a(q2,q)} \\ : & : & : \\ \gamma_{a(1p1,1)} & \gamma_{a(2p1,1)} \ldots\ldots\ldots\ldots & \gamma_{a(qp1,q)} \end{pmatrix}$$

This N* is the required nearly magic rectangle having the following:

i)Each column sum = **(1+2*p1*q)*p1**

ii)The sum of **(2h-1)** th row = **$(2*p1*q^2+q-1)/2$**, for **h=1,2,....,(p1+1)/2**.

iii)The sum of **2h-** th row = **$(2*p1*q^2+q+1)/2$**, for **h=1,2,....,p1**.

D. p is even and q odd but p/2 is even.

This procedure is being taken care of by the function called "**pby2evenqodd** ( )".

We define **n = (q-3)/2** and **p*=p/2**.

Here define

$$S_1 = \begin{pmatrix} 1 & 2 & \ldots\ldots & p^* & p^*+1 & p^*+2\ \ldots\ldots & p \\ p^*+1 & p^*+2 & \ldots\ldots & p & 1 & 2\ \ldots\ldots & p^* \\ p & p-2\ \ldots\ldots & & 2 & p-1 & p-3\ .\ \ldots\ldots & 2 \end{pmatrix} = \begin{pmatrix} \rho_{11} \\ \rho_{12} \\ \rho_{13} \end{pmatrix}$$

$$S_2 = \begin{pmatrix} 1 & 2\ \ldots\ldots\ \ldots p^* & p^*+1\ \ldots\ldots & p \\ p & p-1\ \ldots\ldots p^*+1 & p^*\ \ldots\ldots & 1 \end{pmatrix} = \begin{pmatrix} \rho_{21} \\ \rho_{22} \end{pmatrix}$$

It can be noted that half the column sums are equal to **3*p/2+2** and second half column sums equal to **3*p/2+1** in the matrix $\mathbf{S_1}$. In $\mathbf{S_2}$, all the column sums are equal to **p+1**.

We define a matrix **A** = **((a[i][j]))** of order qxp as follows:

a[i][j] = s_1[i][j]; if n=0

$\mathbf{A} = [\rho'_{211} \ldots. \rho'_{21n}\ \rho'_{11}\ \rho'_{12}\ \rho'_{13}\ \rho'_{221}\ \rho'_{222} \ldots\ldots \rho'_{22n}]'$ with $\rho_{21\,i} = \rho_{21}$ **and** $\rho_{22\,i} = \rho_{22}$ **for1<= i<=n; if n>=1.** We re-stored the elements of the matrix A in some row vectors as

$\mathbf{A} = [\rho'_1\ \rho'_2 \ldots. \rho'_q]'$

i.e. **ρ[i] = ρ[2][1][i] for i=1(1)n**

ρ[i+n] = ρ[1][i] for i=1(1)3

ρ[i+n+3] = ρ[2][2][i] for i=1(1)n.

Here it can be easily observed that

ρ[q+1-i]- ρ[i]=[p-1,p-3,........,1,-1,-3,.....,1-p], 1<=i<=(q-3)/2 and

ρ[(q+3)/2]- ρ[(q-1)/2]=[p-1,p-4,........,2-p/2,p/2-2,p/2-5-3,.....,4-p,1-p].

After defining Kronecker product of matrix earlier, we construct **N** as:

N= A + [0, 2*p1, ……...., 2*(q-1)*p1]' $\otimes$ **1'**$_p$

where **1'**$_p$**= [1,1,..........,1]**^{px1}.

It can be observed that N has following properties:

i) The natural number 1 to pq appears once.
ii) The j^{th} column sum is **(pq**^{2}**+q+1)/2,** for **1<=j<=p1** and the **(j+p1)**th column sum is **(pq**^{2}**+q-1)/2**, for **1<=j<=p1**.
iii) The i^{th} row sum is **p(1+(2*i-1)*p)/2**, for **1<=i<=q**.

This leads us to exchange the corresponding elements between the **i-th** row and the **(q+1-i)-th** row in **N**, such that the resulting **i-th** and **(q+1-i)-th** row sums both equal to **p*(1+pq)/2**. The purpose of this exchange is that, if the **j-th** element say **ρ[i][j](=l+(i-1)p; l ε ρ[i])**, of the **i-th** row is exchanged with the **j-th** element say **ρ[q+1-i][j](=h+(q-i)p; h ε ρ[q+1-i]),** of the **(q+1-j)-th** row, then the sum of the **i-th** row will gain **(h-l)+p(q+1-2i)** and the sum of the **(q+1-i)-th** row will lose **(h-l)+p(q+1-2i)**. So in **N** we exchange the first **p/4** elements and last **p/4** elements of **i-th** row with the first **p/4** elements and last **p/4** elements of **(q+1-i)-th** row. This procedure is repeated for **1<=i<=(q-1)/2**. We store this swapped elements of **N** in another matrix **N*= ((n*[i][j]))**.

For the **i-th** and the **(q+1-i)-th** rows in **N**, there are **p/2** elements exchanged and the sum of the first **p/4** elements and last **p/4** elements of **ρ[q+1-j]- ρ[i]=0**. This will make the **i-th** row sum to gain **p**^{2}**(q+1-2i)/2** and **(q+1-i)-th** row sum to lose **p**^{2}**(q+1-2i)/2**.

Thus, in the resultant matrix **N***, the sum of **i-th** and **(q+1-i)-th** row both equal to **p(1+pq)/2** i.e. all rows are equal.

Now, in matrix N, we exchange the elements of **(p1+i)-th** column with **(2i-1)-th** column, for **1<=i<=p1** and the elements of the **j-th** column with that of the **2j**-th column, for **1<=j<=p1** to obtain the elements of the matrix **N** = ((n**[i][j]))**. Thus this **(N**)'** is the required magic rectangle which is of order **2p1xq** and **p1** is even.

Now the two functions **pby2oddqodd( )** and **pby2evenqodd( )** combined together give the entire algorithm for the construction of generalized nearly magic rectangle of any order.

4.CONSTRUCTION OF WEBPAGE FOR ACCESSING THE PROJECT ONLINE:

(reference [8]&[10])

The project is uploaded on the site www.math.iitb.ac.in/~sandip/ and the system administration can have access to Linux applications. This can easily be done by saving the project links and appropriate functions in the web-disk of mathematics department's personal webpage. So PHP is used so that program can be run which is compatible to Linux. The webpage has various links containing detail of the reports and theories about magic rectangles and squares. The program has a serious limitation to memory overflow, so the program doesn't execute if the product of total number of rows and total number of columns exceeds certain amount. This amount is tested and found approximately equal to 3600. The PHP is implemented embedding the C++ program and the main page contains a calculator named Magic Rectangle Calculator. The output is displayed at the bottom of the page below the calculator input bars. The input is being taken and passed over to the C++ executive file and the output is being generated in the IIT Bombay Secured Shell itself that uses Fedora Core as the Operating System. The Webpage design also involves basic knowledge of HTML.

The webpage provides various instructions of how to use the calculator and what results are expected to obtain. The PHP has provided with enormous help because of the following facts:

- All compatibility problems existing between different browsers are completely solved. The User's browser, receives a normal HTML page after the execution of a PHP code on the server, and so it is always able to display it correctly since it deals with only HTML. This does not happen with scripting languages interpreted by the user's browser. In this case the user downloads the script code and tries to process it on the local machine. This procedure works correctly only if the client is equipped with the right software (generally called plug-ins or built-in support in the browser).
- The server side code processing sees to it that the script code is never visible to the users. That prevents "thefts" of source code.
- The server side code execution requires that your web server has been well configured. It must be able to recognize HTML documents containing PHP code. In order to make this, it is necessary to install a PHP engine and to edit some lines in the web server's configuration file.
- Server side code processing needs resources (CPU time) for generating the dynamic pages. A high number of user requests could overload the server. But generally today's servers such as Apache are made stable enough to handle a relatively large number of users.

A simple code structure of PHP involving HTML applied to develop the web access:

```
<html>
<head>
<title>Php Calculator</title>
 <h3>M.Sc. Project: MAGIC RECTANGLES AND SQUARES</h3>
 </head>
<h5>Please enter row number and column number to be either both odd or both even to get a magic rectangle</h5>
<h5>For Nearly Magic Rectangle, enter one number as odd and other as even</h5>
<li><a href="http://www.math.iitb.ac.in/~sandip/Magic_report_final1.pdf" > Project Report</a>
<body background="http://www.math.iitb.ac.in/~sandip/image6.jpg">
<form action="calculator.php" method="Post">
<?php
if((int)$_POST["a"] != 0|| (int)$_POST["b"] != 0)
{
$a = $_POST["a"];
$b = $_POST["b"];
$op= $_POST["op"];
$result = shell_exec("./mrf4 ".$a." ".$b);
}
else
{
$a=0;
$b=0;
$result=0;
$op="add";
}
?>
<p align=middle>
<font color=maroon><h3>Magic Rectangle Calculator</h3></font>
<table>
<tr>
<td>Row  Number : </td>
<td><input type="text" name="a" value="<?php echo $a; ?>"/></td>
</tr>
<tr>
<td>Column Number : </td>
<td><input type="text" name="b" value="<?php echo $b; ?>"/></td>
</tr>
<tr>
<td><input type="submit" value="       Result       " /></td>
</tr>
</table>
</p>
```

```
<pre><?php echo $result; ?></pre>
</form>
</body>
</html>
```

References:

1. **"Construction of Magic Rectangles of odd order " by Dr. Ashish Das and Dr. Feng Shun Chai.**
2. **"A matrix approach to construct magic rectangles of even order" by Dr.J. P. De Los Reyes ,Dr. Ashish Das and Chand K. Midha.**
3. **"On a method to construct magic rectangles of even order" by Dr. J. P. De Los Reyes, Dr.Ashish Das ,Dr. Chand K. Midha and Dr.P. Vellaisamy.**
4. **"Magic Rectangles revisited" by Thomas R. Hagedorn (Discrete Mathematics).**
5. **"The Magic Square as a Benchmark: Comparing MIP to Improved GA and to a High Performance Minimax AI Algorithm" by Jose B. Fonseca.**
6. **"Magic Rectangles" by Marian Trenkler.**
7. **"Magic Rectangles and Modular Magic Rectangles" by Anthony V. Evans.**
8. **"A PHP Tutorial for Beginners" by *Luigi Arlotta***
9. **(The Mathematical Gazette, Vol. 52, No. 379 (Feb., 1968), pp. 9-12) – "A simple method of constructing a certain magic rectangles of even orders." By Dr. J.P.N. Phillips.**
10. **"HTML for Dummies" by Ed Tittel and Mary Bermeistar.**
11. **"Construction of Nearly Magic Rectangle " by Dr. Ashish Das.**

Apart from these various websites have also been consulted.
Some links are:

12. http://homepage2.nifty.com/googol/magcube/en/rectangles.htm
13. http://www.gpj.connectfree.co.uk/mrm.htm
14. http://www.primepuzzles.net/puzzles/puzz_266.htm
15. http://en.wikipedia.org/wiki/Magic_square
16. http://www.pballew.net/magsquar.html
17. http://home.pacific.net.au/~bangsund/magicrec.htm
18. http://www.sciencedirect.com/science?_ob=MImg&_imagekey=B6V0M-3VSNH9J-R-4&_cdi=5650&_user=444230&_orig=search&_coverDate=04%2F15%2F1996&_sk=999489997&view=c&wchp=dGLbVzb-zSkWA&md5=92e18063df45bcbf19d5b6036492c32b&ie=/sdarticle.pdf