

King Saud University

College of Engineering

Industrial Engineering Department



# Facilities Analysis & Design

## IE 550

### Blocplan Algorithm

**Submitted to:**

Dr. Abdulaziz Al-Tamimi

**By:**

Muhammad Dzulqarnain Al Firdausi

(438106660)

**Table of Contents**

<b>Table of Contents .....</b>	<b>2</b>
<b>List of Figures.....</b>	<b>3</b>
<b>List of Tables .....</b>	<b>4</b>
<b>1. Introduction .....</b>	<b>5</b>
<b>2. Literature review .....</b>	<b>5</b>
<b>3. BLOCPLAN.....</b>	<b>7</b>
<b>3.1. Overview .....</b>	<b>7</b>
<b>3.2. Methodology .....</b>	<b>8</b>
<b>3.3. BLOCPLAN development.....</b>	<b>14</b>
<b>4. Study Case.....</b>	<b>19</b>
<b>4.1. Problem Statement.....</b>	<b>19</b>
<b>4.2. Collecting data.....</b>	<b>19</b>
<b>4.3. Using the new version of BLOCPLAN.....</b>	<b>21</b>
<b>4.4. Result comparison .....</b>	<b>28</b>
<b>4.5. Discussion.....</b>	<b>29</b>
<b>6. References.....</b>	<b>30</b>
<b>APPENDIX.....</b>	<b>31</b>

### List of Figures

Figure 1 Blocplan Algorithm Flowchart.....	8
Figure 2 Department Placement.....	9
Figure 3 Sub-dividing .....	9
Figure 4 Relationship Chart .....	10
Figure 5 Automatic search algorithm .....	10
Figure 6 Input department data form .....	15
Figure 7 Relationship chart input.....	16
Figure 8 Score for each code .....	17
Figure 9 Relationship score calculation for each department .....	18
Figure 10 Length width ratio for layout.....	18
Figure 11 Layout generation with random layout algorithm [4] .....	19
Figure 12 Production line of the company.....	20
Figure 13 Cell data of the ammunition division .....	22
Figure 14 Relationship chart of the company .....	23
Figure 15 Score for each relationship code.....	24
Figure 16 Score for each relationship code.....	25
Figure 17 Ratio selection .....	26
Figure 18 Initial layout from blocplan .....	27
Figure 19 Improved layout.....	27

**List of Tables**

Table 1 Literature review.....	6
Table 2 BLOCPLAN pros and cons .....	7
Table 3 Code for Relationship Chart and its score .....	9
Table 4 Area of each machine cell.....	20
Table 5 Relationship chart for ammunition division .....	21
Table 6 R.Score comparison .....	29

## 1. Introduction

Facility layout problem (FLP) is a fundamental problem which seeks to reduce material handling costs, work in process, lead times, utilize existing space more effectively, make plant adaptive to future changes, provide a healthy, convenient and safe environment for employees, and generally increase productivity through determining an efficient arrangement (layout) of the required facilities within the organization. This problem has many applications in real life situations.

However, the main attentions of the research in this area have been focused on the manufacturing applications. It is noteworthy to know that material handling cost constitutes 20–50% of the total operating expenses in manufacturing environments which can considerably be reduced through designing an efficient layout configuration. Material handling cost is mainly dependent on the amount of material flow and distances between locations of the facilities. Accordingly, minimizing the material handling cost has been considered as the common objective in the literature [8].

In other hand, although most facility layout programs seek the layout that will attempt to minimize the material handling activity for the problem, this objective approach to the problem is not the only solution technique that may apply in many situations. There may be other factors that have to be considered. There could be vibration problems that occur in some department that make it impractical for it to be near another department. Similarly, it would be undesirable to locate a painting department near a grinding or machining department. These factors can very often enter into the layout design. There are other scoring procedures to evaluate a layout and the results obtained from these procedures can be calculated and displayed for each layout that is created.

Facility layout involves the arrangement of departments in a facility. The objective of facility layout analysis is to develop layouts which maximize the advantageous placement while minimizing the disadvantageous placement of departments. Computers have been utilized since the 1960's and 1970's to assist with the high computational demands required for the calculations necessary for producing these types of facility layouts.

## 2. Literature review

The table below shows literatures related to facility layout problem.

Table 1 Literature review

No.	TITLE	DESCRIPTION & ABSTRACT	TECHNIQUE	Year of Publication
1	Production facility layout by comparing moment displacement using BLOCPLAN and ALDEP Algorithms (M Tambunan)	The purpose of this study is comparing movement displacement using BLOCPLAN and ALDEP algorithm in order to redesign existing layout.	BLOCPLAN and ALDEP	IOP Conference Series: Materials Science and Engineering 309 012032 (2018)
2	Blocplan for windows: Computer aided facility layout methodology and enhancements (Donaghey)	BLOCPLAN for Windows is an evolutionary development of its DOS version. It is able to result more alternative layout, e.g. for a nine department problem the number of possible layouts is close to 20 million	BLOCPLAN	Proceedings of the 2007 Industrial Engineering Research Conference
3	FLEXPART: facility layout expert system using fuzzy linguistic relationship codes (Badiru)	An expert system generates facility layout alternatives based on the fuzzy logic theory and the results provided by BLOCPLAN (software for buildingplant layouts)	BLOCPLAN Fuzzy Logic	IIE Transactions, 28, 295–308 (1996)
4	Production facility layout design using blocplan algorithm (Puspita)	The implementation of the BLOCPLAN algorithm to <b>improve</b> the production <b>facility layout</b> . Total movement moment and the Benefit-cost ratio (BCR) are used to compare existing and alternative layout	Benefit-cost ratio (BCR) Blocplan	American Scientific Publishers (2015)
5	Comparing alternative plant layouts based on Craft and Blocplan algorithms (Leonardo)	The paper <b>compares</b> two proposed layout models using CRAFT and BLOCPLAN algorithms. An initial layout is analyzed and the two algorithms are applied to the layout therefore the best score resulted from a layout is selected.	Craft and Blocplan	Metris Journal Vol. 15 (2014) 55–64

No.	TITLE	DESCRIPTION & ABSTRACT	TECHNIQUE	Year of Publication
6	An interactive layout heuristic based on hexagonal adjacency graphs (Goetschalckx)	The paper presents an efficient and interactive two-stage heuristic for the generation of block layouts.	Optimization, Graph algorithms, Block layout algorithms	European Journal of Operational Research 63 (1992) 304-321

### 3. BLOCPLAN

#### 3.1. Overview

BLOCPLAN is an interactive program developed by Donaghey and Pire (1991) that can develop a single-story or multistory layout and has a number of useful features. It offers a number of heuristic algorithms for solving the layout problem and can handle quantitative as well as qualitative data. It allows the user to enter product routing data. BLOCPLAN provides ample opportunity to view and edit data. The user provides flow data in the form of a relationship chart, which is then used to develop a layout.

The literature review which has been done and software testing of BLOCPLAN algorithm (BPLAN90.exe) give the author several insights related to the pros and cons of the algorithm [9].

Table 2 BLOCPLAN pros and cons

Pros	Cons
<ul style="list-style-type: none"> <li>• Providing the square shape for each department</li> <li>• Resulting up to 20 alternatives with different scores for a layout</li> <li>• Combining qualitative and quantitative data</li> <li>• Able to change the position of departments manually</li> </ul>	<ul style="list-style-type: none"> <li>• 3 tiers may be limiting</li> <li>• Can lead to poor department shapes (long and skinny departments)</li> <li>• Not providing the total movement score for a layout</li> </ul>

The DOS version of BLOCPLAN allows user to input product data and their department sequence. However, the software is not providing the total movement score. Puspita work [7] count the total movement score manually which it should be provided by BLOCPLAN automatically.

### 3.2. Methodology

The user provides the **required area** for the layout that is to be created. He/she does this by specifying the number of "departments" that are to be included in the layout, and then gives the names and required areas for each of these departments. The system will find the total area for all of these departments, and this will be the required area for the layout.

The user also gives the desired **length to width ratio** (L/W) for the layout. This is the L/W ratio for the entire facility that will contain the departments. This allows for efficient aisle placement and materials handling. The **length and the width** of the facility can be calculated using the L/W ratio and the total area. BLOCPLAN uses tiers and zones to develop a layout.

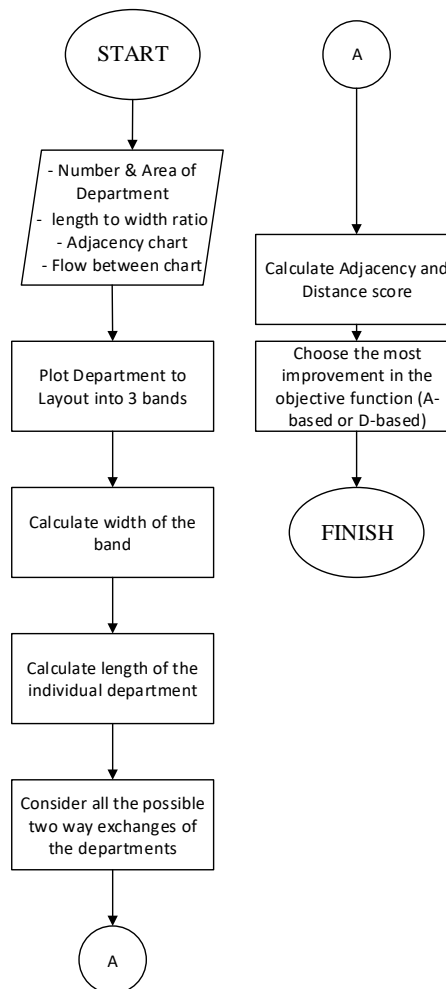


Figure 1 Blocplan Algorithm Flowchart

The facility is divided into **three tiers** (bands) containing three zones each. The top tier has zones A, B, and C. The mid-tier has zones D, E, and F. The third or bottom tier has zones G, H, and I [4].



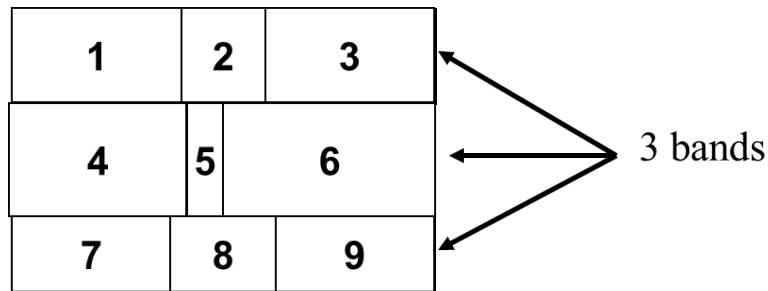


Figure 2 Department Placement

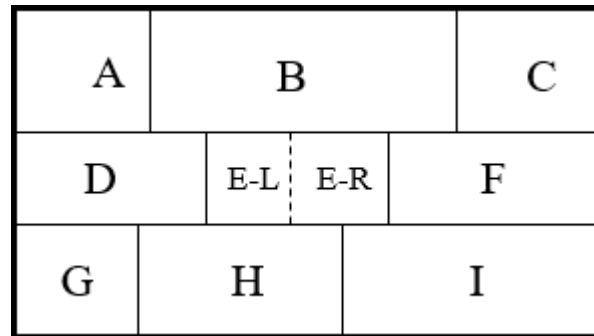


Figure 3 Sub-dividing

Each zone is also sub-divided into two sub-zones, L and R. AL refers to the area in the left portion of zone A, and AR refers to the right side of zone A. Using this notation there are 6 positions that departments can use in Tier 1, 6 positions for Tier 2, and 6 positions for Tier 3.

**BLOCPLAN utilizes the Relationship chart** method. With this relationship chart method, a user examines each pair of departments and makes a judgment of how important it is that the pair of departments be close together. Certainly, the amount of material handling between the departments is a factor in this judgment. Muther used the A, E, I, O, U, and X codes. An A relationship indicated that it was "Absolutely Essential" that the departments be close, an E relationship meant that it was "Essential", an I meant that it was "Important", an O means "Ordinary Closeness OK", a U indicated that closeness is "Unimportant", and an X meant that it was "Undesirable" to have them close [2].

Table 3 Code for Relationship Chart and its score

Code	Description	Score
A	Absolutely Essential	10
E	Essential	5
I	Important	2
O	Ordinary Closeness OK	1
U	Unimportant	0
X	Undesirable	-10

	D2	D3	D4	D5	D6	D7	D8	D9
D1	A	U	O	U	U	A	O	O
D2		U	E	O	U	I	U	U
D3			U	U	A	U	U	O
D4				O	U	U	O	U
D5					U	O	U	U
D6						U	A	U
D7							U	I
D8								U

Figure 4 Relationship Chart

The desired relationship chart shown in Figure 4 has four A's, 1 E, 2 I's, 1 E, 8 O's, and 21 U's. Suppose the worth vector of the codes are A: 10, E: 5, I: 2, O: 1, U: 1, X: -10. Therefore, the total worth of all the positive relationship codes for the problem is 57.

If a user has material handling information concerning a layout, this information can be converted to a relationship chart by BLOCPLAN. It shows material handling activity, in Moves/Time Period, between each pair of departments. BLOCPLAN can develop a Muther type relationship chart from this data. It takes the maximum number of loads between departments in the chart and divides this value by 5. For the nine department layout of Figure 2, suppose the maximum number of loads is 300. Depts 3 and 6 and Depts 6 and 8 each have 300 loads per time period between them. Thus,  $300/5 = 60$ . A relationship code of "A" would be assigned to pairs of departments that have a From/To value in the inclusive interval [241, 300]. An "E" would be assigned for values in the inclusive interval [181, 240], an "I" for values within the interval [121, 180], an "O" for the interval [61, 120], and a "U" for values that are equal to or less than 60. There would be no "X" values (Undesirable) assigned for the chart.

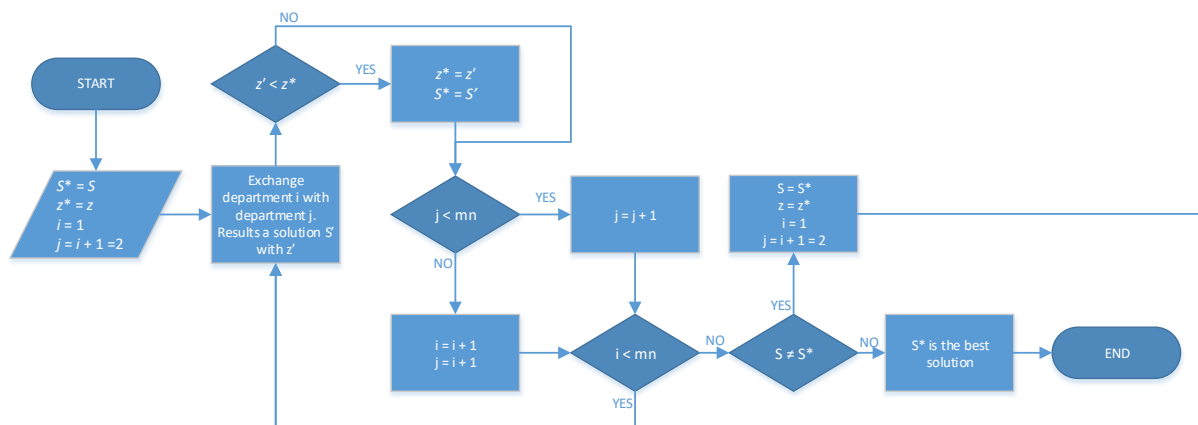


Figure 5 Automatic search algorithm

BLOCPPLAN can develop a layout randomly or using an automatic search algorithm (Figure 5). The random layout algorithm generates layouts without considering the flow or interaction between departments. The automatic search algorithm generates an initial layout randomly using the random layout algorithm and then uses the improvement algorithm to find a better layout. It calculates the rel-dist score of the new layout. The steps of automatic search algorithm are:

- *Step 1:* Let  $S$  be the initial solution provided by the user and  $z$  is its distance score. Set  $S^* = S$ ,  $z^* = z$ ,  $i = 1$ ;  $j = i + 1 = 2$ .
- *Step 2:* Consider the exchange between the positions of departments  $i$  and  $j$  in the solution  $S$ . If the exchange results in a solution  $S'$  that has an OFV  $z' < z^*$ , set  $z^* = z'$  and  $S^* = S'$ . If  $j < mn$ , set  $j = j + 1$ ; otherwise, set  $i = i + 1$ ,  $j = i + 1$ . If  $i < mn$ , repeat step 2; otherwise, go to step 3.
- *Step 3:* If  $S \neq S^*$ , set  $S = S^*$ ,  $z = z^*$ ,  $i = 1$ ,  $j = i + 1 = 2$  and go to step 2. Otherwise, return  $S^*$  as the best solution to the user. Stop

To determine the rel-dist score, BLOCPPLAN computes the actual distance times the numeric value of the corresponding relationship between each pair of departments. Then it adds these computed values to obtain the rel-dist score. Mathematically, adjacency score is given by [4]

$$\frac{\sum_{i=1}^{n-1} \sum_{j=i+1}^n R_{ij} D_{ij}}{\sum_{i=1}^{n-1} \sum_{j=i+1}^n R_{ij}} \quad (3.1)$$

and rel-dist score is given by

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^n d_{ij} R_{ij} \quad (3.2)$$

Where

$$D_{ij} = \begin{cases} 1 & \text{if departments } i \text{ and } j \text{ are on the same floor and adjacent} \\ 0 & \text{otherwise} \end{cases}$$

$R_{ij}$  is the numeric value assigned to the relationship code between departments  $i$  and  $j$

$n$  is the total number of departments

$d_{ij}$  is the rectilinear distance between the centers of departments  $i$  and  $j$

Department pairs with a negative numeric value are not included in the summation in the denominator of the adjacency score formula. Instead of printing the rel-dist score directly,

BLOCPLAN normalizes the score and prints the resulting value. To do so, it computes a simple lower and upper bound on the value of the rel-dist score by multiplying two vectors—a distance vector whose elements are arranged in nondecreasing order and a relationship code or flow vector whose elements are arranged in nonincreasing order. The first vector includes the distance, whereas the second includes the numeric values corresponding to the relationship code between each pair of departments. Because we are trying to minimize the sum of the product of flow times distance calculated for each pair of departments, it is well known that a lower bound can be obtained by arranging elements of the flow vector in nonincreasing order and the distance vector elements in nondecreasing order. Thus, we are trying to match the largest flow elements with the smallest distance element. Although such a matching may not lead us to the OFV of a feasible solution, it provides us a reasonably tight lower bound to compare the OFV's of other feasible solutions. Obviously, the optimal solution is the one whose OFV is closest to the lower bound (among all possible feasible solutions). To get an upper bound, we may do the reverse of what we did to compute the lower bound—arrange elements of the two vectors so that the largest flow element is matched with the largest distance element and multiply the two vectors. This means that both the vectors are to be arranged so that their elements are in nondecreasing order. BLOCPLAN thus computes the lower and upper bounds and determines the normalized rel-dist score, called R-score, as [4]

$$\text{R-score} = 1 - \frac{(\text{rel-dist score} - \text{lower bound})}{(\text{upper bound} - \text{lower bound})} \quad (3.3)$$

Thus, an R-score of 1 would mean that the corresponding solution is optimal. Of course, such a value for R-score is highly unlikely because it means that the rel-dist score of the corresponding layout is equal to the lower bound. Similarly, an R-score value of 0 (also highly unlikely) means that the corresponding layout is the worst possible layout because its OFV is equal to the upper bound.

Method of scoring a layout is the "Relationship-Distance" procedure. With this procedure the distance (Rectilinear) between the centroids of each pair of departments is multiplied by the numeric worth of the relationship code for the departments. The sum of all of these values is the "Unnormalized Relationship Distance Score" (URDS) for the layout, where the lower the value the better the layout. With this philosophy, departments that have an "A" relationship will be as close together as possible. Departments with an "X" relationship will be as far apart as possible. BLOCPLAN attempts to normalize this score. For a given layout a vector of all the distances between departments is created. All of the values in this

vector are put in ascending order. A vector of the numeric relationship chart values is also created. These are also put into ascending order from the lowest to the highest. A minimum, or lower bound (LB), for the Relationship Distance Score would be the sum of the products with the highest values in the relationship vector multiplied by the lowest values in the distance vector. BLOCPLAN also creates an upper bound (UB) for the layout. This value is found by multiplying the largest code values in the numeric relationship vector by the largest values in the distance vector, and summing the results. This value is called the upper bound (UB) for the layout. The normalized relationship distance score is formula (3.3)

Another scoring procedure that BLOCPLAN presents is simply the total of the material handling activity for the layout. This value is given in Load-Feet, and assumes that the user has given a From/To chart that shows the material handling activity between departments. Again the distance between departments for a layout uses rectilinear distance between department centroids. For any layout that is created by BLOCPLAN, the scores from each of the scoring procedures are reported.

There was a great deal of time spent trying to develop a procedure that would allow BLOCPLAN to create a near optimal layout from just the information provided by the user. It seemed logical to have departments with a large number of positive relationship code values put in the center of the layout, and pairs of departments that have an "X" relationship code placed on opposite corners of the layout. There was still some random placement of departments with the algorithm, so there was some difference in the layouts each time the option was called. However, the option did just not obtain consistently "good" layouts. The procedure that was put in to replace it was the "Automatic Search" algorithm. When this option is selected, BLOCPLAN will place all of the departments into the layout randomly. There are 18 sites where a department can be placed. (AL, AR, BL, ..., FL). For a ten department layout, there would be 43,758 possible site selections (Combinations of 18 things taken 10 at a time  ${}_{18}C_{10}$ ). BLOCPLAN will find the adjacency score for the random layout. It will then switch the locations of each pair of departments. For example, suppose Dept 1 is in DR and Dept 2 is in CL. It will place Dept 1 in CL and Dept 2 in DR. It will calculate the adjacency score of this revised layout. If the score is better than the original layout, it will use the revised layout as the starting point and again begin to switch each pair of departments. If there is no improvement for the layout after the switch for departments 1 and 2, it will continue and switch 1 and 3 to see if an improvement can be made. For a 10 department layout, the number of switches that can be made would be 45 [ ${}_{10}C_2$ ]. Any time an improvement in the score is made, the revised layout will be the used, and the switching process will begin again with the revised layout. The

final layout that will be presented will be the one where all 45 switches have been made with no improvement in the score. The centroids and the dimensions for every department in each layout will be presented when it is created.

Each time the "Automatic Search" algorithm is used, the user must specify the number of layouts that are desired. The maximum number is 20. If 20 layouts are specified, the system will start with a random layout for the first one and keep switching department locations until no further improvements can be made. It will then start with a random layout for the second one and repeat the process. After 20 layouts have been created, they will be presented to the user. The adjacency score, the relationship distance score, and the material handling required for each layout is shown. The user can then ask for 20 more layouts to be created. In a short amount of time the user may review several hundred good layouts. We have found it a good idea to keep a record of the "extra good" layouts that are presented. This scorecard record shows the data of layouts that were created and saved by the user. The adjacency, the relationship distance, and the product movement criteria were used for all the layouts that were created.

### **3.3. BLOCPLAN development**

In order to make a better understanding about BLOCPLAN algorithm, here is a practical example to develop a layout using BLOCPLAN. Recall the BLOCPLAN flowchart in Figure 1.

#### **Step 1:**

Give name and Area of Departments as inputs for the BLOCPLAN with maximum number of department is 18. Departments and their area are in the following picture.

The screenshot shows a Windows-style application window titled 'formDept'. It contains a data entry form with three columns: 'Number', 'Department', and 'Area'. The form is divided into two main sections: 'New problem' and 'Enter or modify'. The 'New problem' section contains a table with 10 rows of data. The 'Enter or modify' section contains a table with 8 rows of empty input fields. At the bottom, there are summary statistics: 'Average' (6666.67) and 'Std Dev. Area' (2981.42). A 'Total' field shows 60000. There are three buttons at the bottom: 'Continue', 'Print', and 'Back'. Numbered callouts 1 through 6 highlight specific UI elements: 1 points to the 'Number' column header, 2 points to the 'Department' column header, 3 points to the 'Area' column header, 4&5 point to the 'Average' and 'Std Dev. Area' fields, and 6 points to the 'Total' field.

Number	Department	Area
1	1	8000
2	2	4000
3	3	8000
4	4	10000
5	5	2000
6	6	12000
7	7	6000
8	8	4000
9	9	6000
10		
11		
12		
13		
14		
15		
16		
17		
18		

Average: 6666.67  
Std Dev. Area: 2981.42  
Total: 60000

Figure 6 Input department data form

We use Visual Basic for developing the BLOCPLAN. The code is in the appendix. Figure 5 shows department form. This version of BLOCPLAN is able to show the **average and standard deviation** for total area automatically. This **new feature** is important because it can inform the user not to exceed the available area when he is writing down the department data. Input for Area of department should be number, therefore we prevent an input that is not a number.

Variables that we use in this code are:

```
Public Shared txtNo(18), txtDept(18), txtArea(18), txtAvg, txtDev, txtTotal As
    TextBox
Private lblNo, lblDept, lblArea, lblTotal, lblEnter, lblAvg, lblDev As Label
```

```

Private noPos, deptPos, areaPos, lblTotalPos, totalPos, problemPos, enterPos,
avgPos, devPos, continuePos, printPos, backPos As Point
Private btnContinue, btnPrint, btnBack As Button

```

Where:

1. txtNo(18) = variable to store the number from 1 to 18 in Number column
2. txtDept(18) = variable to store the department name from 1 to 18 in Department column
3. txtArea(18) = variable to store the department area from 1 to 18 in Area column
4. txtAvg = variable to store the average of all input area
5. txtDev = variable to store the standard deviation of all input area
6. txtTotal = variable to store the total area from all departments

## Step 2:

When a user has inputted the departments info and click continue button, he/she will see the next form of adjacency score input.

The screenshot shows a window titled "scoreForm" with a grid for entering scores. The grid has 9 rows and 9 columns. The first column contains numbers 1 through 9. The second column contains numbers 2 through 9. The remaining columns contain letters A, E, O, I, U. The legend at the bottom indicates: A=Absolutely, E=Essential, I=Immediately, O=Ordinary, U=Unimportant, X=Undesirable. Buttons for "Continue", "Print", and "Edit Score" are at the bottom.

	2	3	4	5	6	7	8	9
1	A	E	O	U	I	U	U	U
2	...	A	I	O	E	U	U	U
3	...	...	E	I	O	U	U	U
4	...	...	...	U	O	I	U	U
5	...	...	...	...	A	E	I	I
6	...	...	...	...	...	O	O	O
7	...	...	...	...	...	...	U	U
8	...	...	...	...	...	...	...	U
9	...	...	...	...	...	...	...	...

Enter or change Code      A=Absolutely      I=Immediately      U=Unimportant  
E=Essential      O=Ordinary      X=Undesirable

Continue    Print    Edit Score

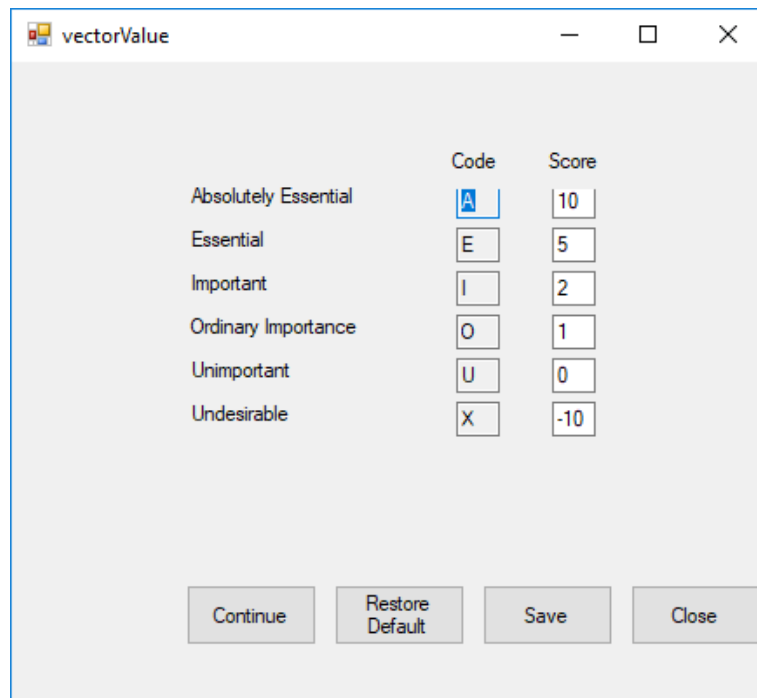
Figure 7 Relationship chart input



For adjacency score input, we prevent improper input so the user can only input letters: A, E, I, O, U which are capital letters. Whenever the user input it in lowercase, the program change it to uppercase automatically.

### Step 3:

Click edit score button to set the score for each relationship code. User can manually edit the score then save the change. This version provides Restore Default button for getting the score of each code [2].



The screenshot shows a Windows-style dialog box titled 'vectorValue'. Inside, there is a table with two columns: 'Code' and 'Score'. The rows represent different relationship levels: 'Absolutely Essential' (Code: A, Score: 10), 'Essential' (Code: E, Score: 5), 'Important' (Code: I, Score: 2), 'Ordinary Importance' (Code: O, Score: 1), 'Unimportant' (Code: U, Score: 0), and 'Undesirable' (Code: X, Score: -10). Each code is in a small text box, and each score is in a small text box. At the bottom of the dialog, there are four buttons: 'Continue', 'Restore Default', 'Save', and 'Close'.

	Code	Score
Absolutely Essential	A	10
Essential	E	5
Important	I	2
Ordinary Importance	O	1
Unimportant	U	0
Undesirable	X	-10

Figure 8 Score for each code

User can obtain the relationship score for each department by clicking continue button of the relationship diagram input. Then, count score button is provided to calculate the score. This **new form** can show which scores contribute to the total value of a department and it helps user knowing the algorithm well because he can trace which number contributes for next calculation, and so on

The 'vectorScore' application window displays a table with three columns: 'No', 'Score', and 'Value'. The table contains data for nine departments. Below the table are two buttons: 'Continue' and 'Count Score'.

No	Score	Value
1	10,5,1,0,2,0,0,0,0,	18
2	10,10,2,1,5,0,0,0,0,	28
3	5,10,5,2,1,0,0,0,0,	23
4	1,2,5,0,1,2,0,0,0,	11
5	0,1,2,0,10,5,2,2,	22
6	2,5,1,1,10,1,1,1,	22
7	0,0,0,2,5,1,0,0,	8
8	0,0,0,0,2,1,0,0,	3
9	0,0,0,0,2,1,0,0,	3

Figure 9 Relationship score calculation for each department

The 'ratioForm' application window prompts the user to 'Select a length (horizontal) to width (vertical) ratio'. It offers five selection options (Sel. 1 to Sel. 5) and a 'Specify Ratio' option. The 'Chosen Ratio' is currently set to 0 X 0. Buttons for 'Back' and 'Execute' are at the bottom.

Select a length (horizontal) to width (vertical) ratio

Sel. 1: 1.35 x 1.00

Sel. 2: 2.00 x 1.00

Sel. 3: 1.00 x 1.00

Sel. 4: 1.00 x 2.00

Sel. 5: Specify Ratio

**Chosen Ratio :**  
0 X 0

Back Execute

Figure 10 Length width ratio for layout

The available ratio for a layout is given in the figure 9. If user wants another ratio, he can choose Specify Ratio and then input the length and width of the area.

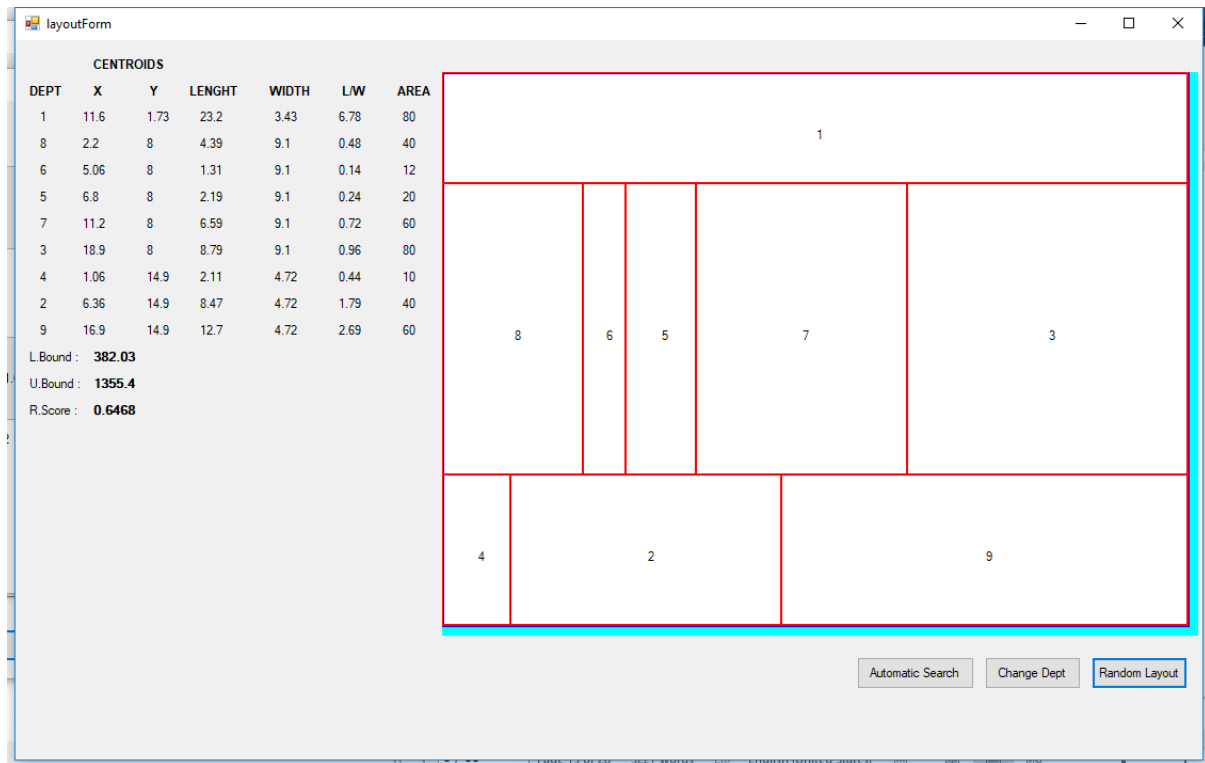


Figure 11 Layout generation with random layout algorithm [4]

Automatic Search with random layout button, user can change the position of department in the layout randomly.

#### 4. Study Case

For implementing the new version of BLOCPLAN, we do a small research for getting the appropriate data. We get data from Kustriyanto [10] which studied layout improvement in strategic manufacturing company, ammunition division and we will compare the result from the paper with our result.

##### 4.1. Problem Statement

How to improve the existing layout of the company by using BLOCPLAN algorithm.

##### 4.2. Collecting data

Existing layout of the department in the company. This department has several production lines with different machine generation/ technology and capacity.

Those machines are grouped based on their process type (Figure 12). The groups are:

- Cell 1 : Drawing I, drawing II, degreasing
- Cell 2 : Drawing I, drawing II, cutting

- Cell 3 : Annealing dan cleaning
- Cell 4 : Indenting, necking, drilling
- Cell 5 : Cleaning, drilling, grooving
- Cell 6 : Gauging
- Cell 7 : Annealing
- Cell 8 : Visual inspection

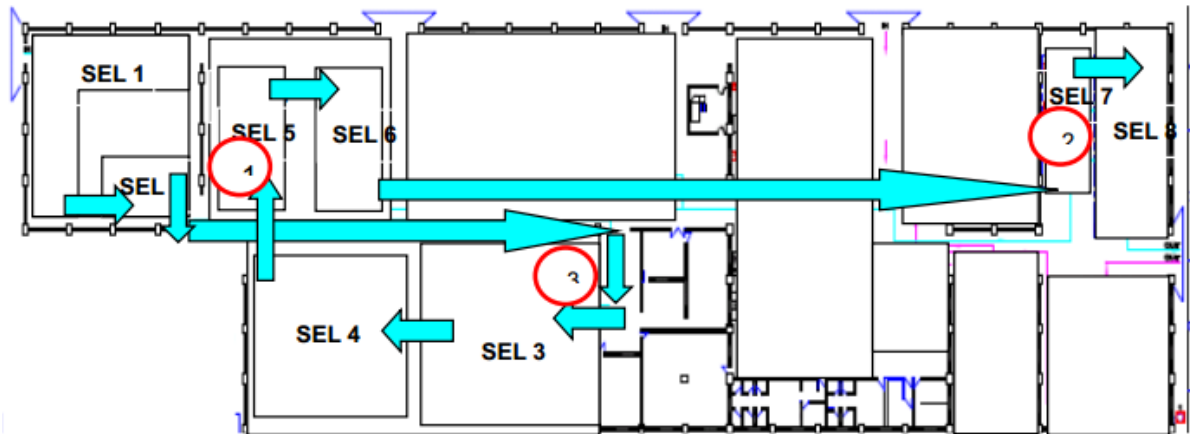


Figure 12 Production line of the company

Data for each cell in the company with its area are in the following table:

Table 4 Area of each machine cell

No.	Cell	Cell Name	Dimension, (l x w) m	Area m <sup>2</sup>	Scale :10
1	Cell 1	Drawing	15,66x9,00	140,94	14
2	Cell 2	Cutting	15,66x4,74	74,23	7.4
3	Cell 3	Annealing and cleaning	21,63x15,63	338,08	33.8
4	Cell 4	Indenting, necking, drilling	16,72x15,63	261,33	26.1
5	Cell 5	Cleaning, Drilling, grooving	11.1x11,16	123.89	12.3
6	Cell 6	Gauging	15,60x5,58	87,06	8.7
7	Cell 7	Annealing	4,89x11,30	55,26	5.5
8	Cell 8	Visual inspection	7,65X14,19	108,55	10.8

Relationship between departments is obtained by qualitative assessment. The relationship chart is depicted in the following table:

Table 5 Relationship chart for ammunition division

Cell	1	2	3	4	5	6	7	8
1		A	O	X	X	X	X	X
2			A	X	X	X	E	U
3				A	I	U	X	X
4					A	E	U	U
5						A	I	E
6							A	U
7								A
8								

#### 4.3. Using the new version of BLOCPLAN

Generating layout using BLOCPLAN needs some data input. We can generate the initial layout and then use automatic search algorithm to get an improvement. The steps are:

##### Step 1:

Input data of department name and the area. BLOCPLAN will automatically calculate the total area, average of the area, and standard deviation (Figure 13)

Department

Number	Department	Area
1	Drawing	14
2	Cutting	7.4
3	Annealing & cleanin	33.8
4	Indenting, necking,	26.1
5	Cleaning, Drilling & C	12.3
6	Gauging	8.7
7	Annealing	5.5
8	Visual inspection	10.8
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		

New problem

Enter or modify

Average 15.4

Std Dev. Area 9.79

Total 107.8

Continue Print Back

Figure 13 Cell data of the ammunition division

**Step 2:**

Set the Relationship chart based on the data acquired from the paper

scoreForm

		2	3	4	5	6	7	8
1	Drawing	A	O	X	X	X	X	X
2	Cutting	...	A	X	X	X	E	U
3	Annealing & cleaning	...	...	A	I	U	X	X
4	Indenting, necking,	...	...	...	A	E	U	U
5	Cleaning, Drilling & C	...	...	...	...	A	I	E
6	Gauging	...	...	...	...	...	A	U
7	Annealing	...	...	...	...	...	...	A
8	Visual inspection	...	...	...	...	...	...	...

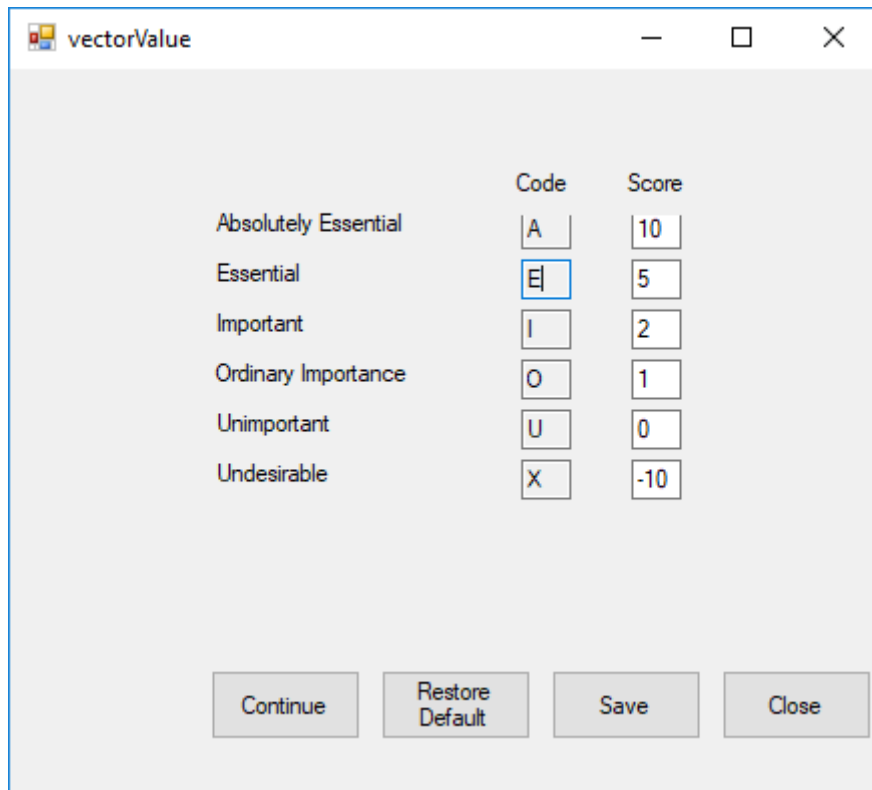
Enter or change Code      A=Absolutely  
E=Essential      I=Immediately  
O=Ordinary      U=Unimportant  
X=Undesirable

Continue    Print    Edit Score

Figure 14 Relationship chart of the company

**Step 3:**

Check the score for each code whether there is a change or just using the default score. In this case study we use the default score provided by Donaghey [2]. Then, we can close the form and click on the continue button on relationship chart window.



The screenshot shows a window titled 'vectorValue' with a table of relationship codes and scores. The table has two columns: 'Code' and 'Score'. The rows are: Absolutely Essential (A, 10), Essential (E, 5), Important (I, 2), Ordinary Importance (O, 1), Unimportant (U, 0), and Undesirable (X, -10). The 'E' in the 'Code' column for 'Essential' is highlighted. At the bottom of the window are four buttons: 'Continue', 'Restore Default', 'Save', and 'Close'.

	Code	Score
Absolutely Essential	A	10
Essential	E	5
Important	I	2
Ordinary Importance	O	1
Unimportant	U	0
Undesirable	X	-10

Continue Restore Default Save Close

Figure 15 Score for each relationship code

**Step 4:**

We can get the score for each department after we click on the continue button from the relationship chart.



vectorScore

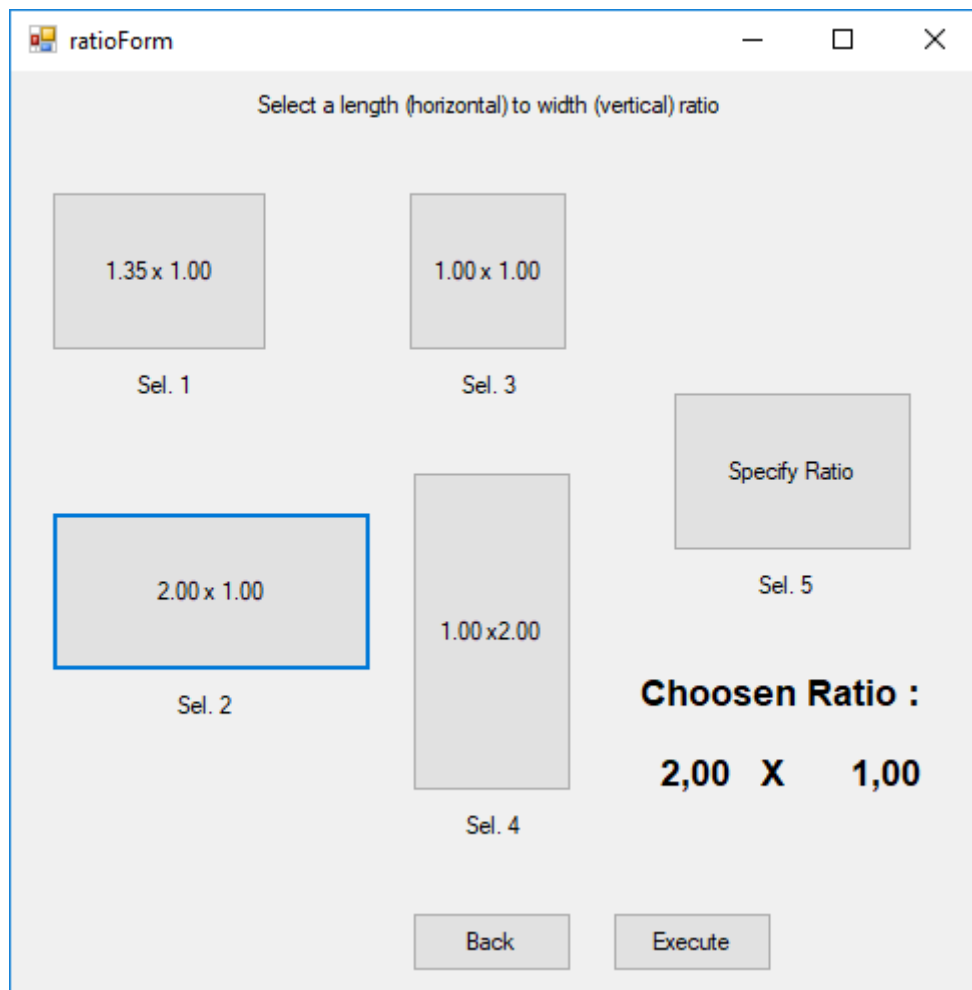
No		Score	Value
1	Drawing	10,1,-10,-10,-10,-10,-10,	-39
2	Cutting	10,10,-10,-10,-10,5,0,	-5
3	Annealing & cleaning	1,10,10,2,0,-10,-10,	3
4	Indenting, necking,	-10,-10,10,10,5,0,0,	5
5	Cleaning, Drilling & C	-10,-10,2,10,10,2,5,	9
6	Gauging	-10,-10,0,5,10,10,0,	5
7	Annealing	-10,5,-10,0,2,10,10,	7
8	Visual inspection	-10,0,-10,0,5,0,10,	-5

Continue

Figure 16 Score for each relationship code

**Step 5:**

From the relationship score, we will select one of the available ratios for the area. We choose the ratio 2.00 x 1.00 (selection 2) for this case, based on the actual data of ammunition department.



The screenshot shows a software window titled "ratioForm" with standard Windows window controls (minimize, maximize, close). The main instruction is "Select a length (horizontal) to width (vertical) ratio". There are five selection options, each represented by a rectangle with its ratio and a label below it:

- Sel. 1: 1.35 x 1.00
- Sel. 2: 2.00 x 1.00 (This option is highlighted with a blue border)
- Sel. 3: 1.00 x 1.00
- Sel. 4: 1.00 x 2.00
- Sel. 5: Specify Ratio

At the bottom right, the "Chosen Ratio" is displayed as "2,00 X 1,00". At the bottom center, there are two buttons: "Back" and "Execute".

Figure 17 Ratio selection

**Step 6:**

Execute the ratio. The randomized department location in layout will be appeared.

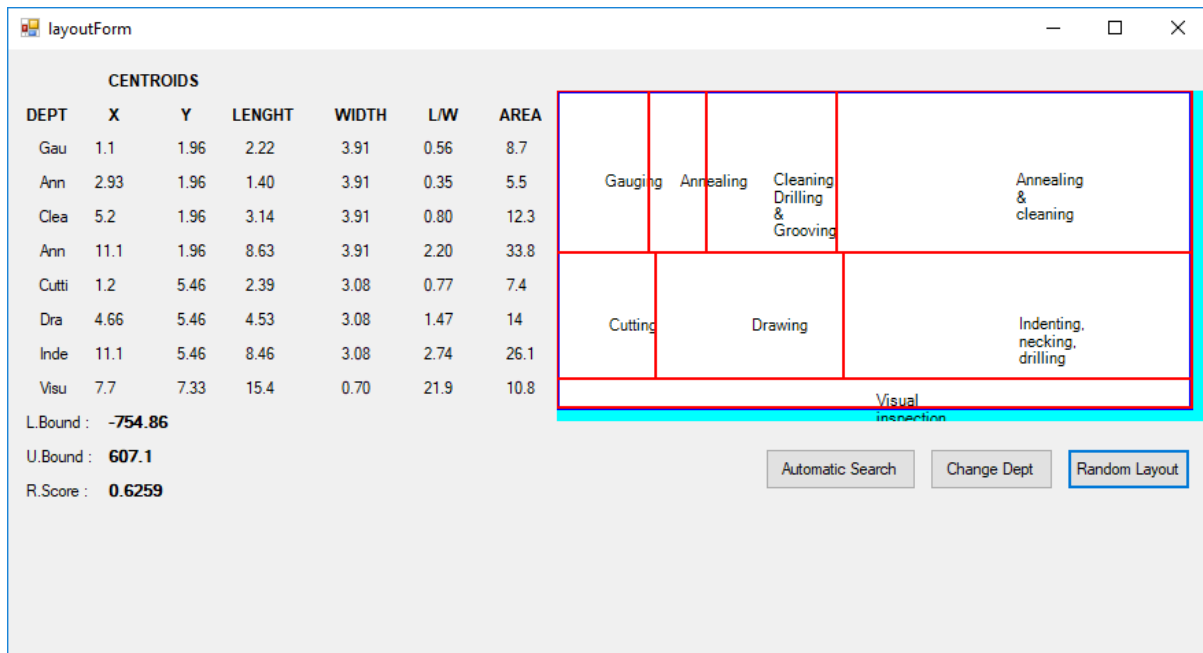


Figure 18 Initial layout from blocplan

BLOCPLAN shows the value of Lower Bound, Upper Bound, and R.Score of the layout. The value is -754.86; 607.1; 0.6259 respectively.

### Step 7:

Do automatic search algorithm with clicking on the Automatic Search. This button makes a hundred replication of change. Whenever it gets a higher R.Score, it keeps the change and continues the search.

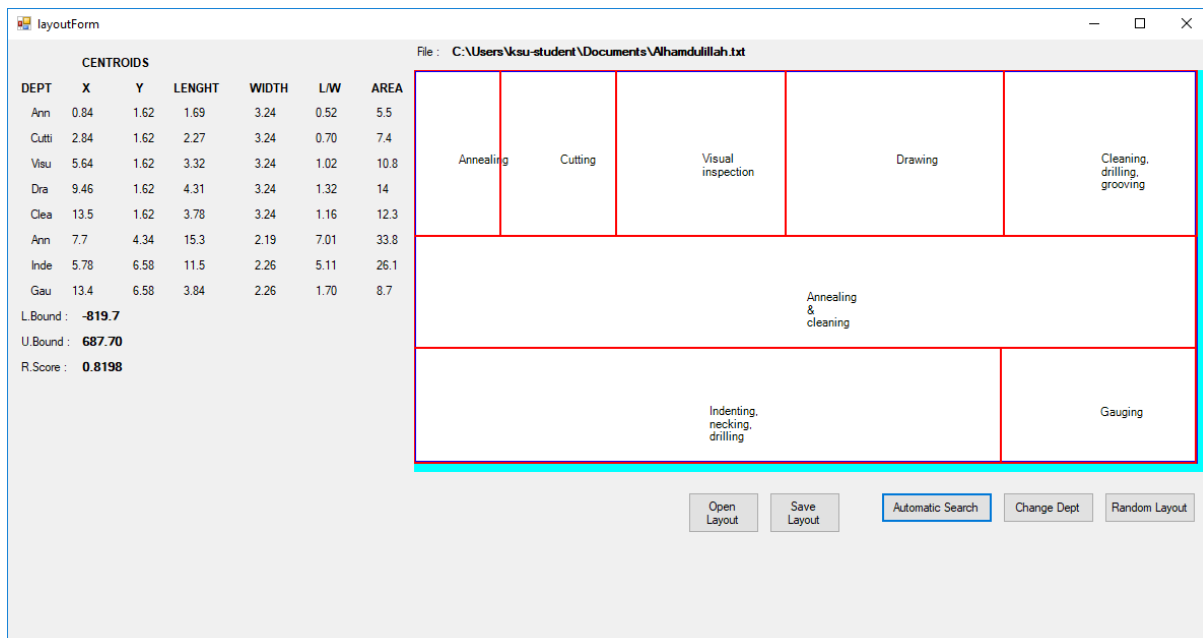


Figure 19 Improved layout

We can see that after the improvement of layout, the R.Score is 0.7989 or there is a **27.6%** improvement from the previous layout.

#### 4.4. Result comparison

Result from the paper [10] for R.Score is compared with the R.Score from the new version of BLOCPLAN. The alternative layout from the paper are:

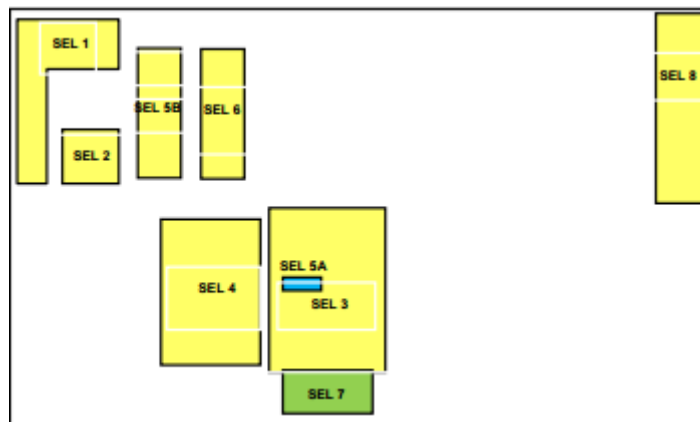


Figure 20 Alternative layout 1 from paper

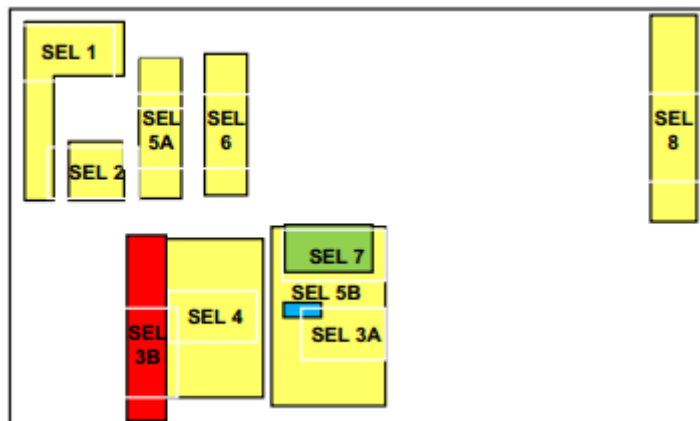


Figure 21 Alternative layout 2 from paper

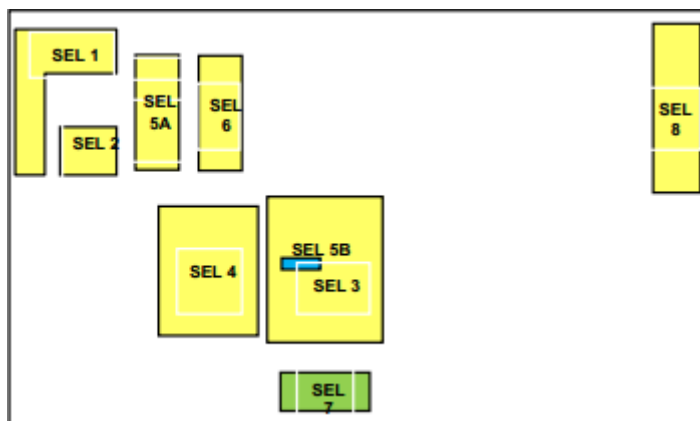


Figure 22 Alternative layout 3 from paper

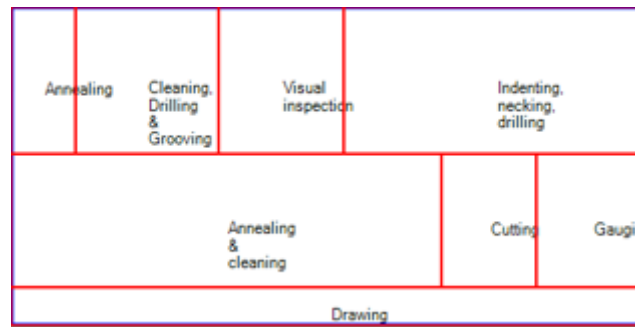


Figure 23 Improved layout from new blocplan

The R.Score for each layout:

Table 6 R.Score comparison

No.	Alternative Layout	R.Score
1	Alternative 1	0.67
2	Alternative 2	0.67
3	Alternative 3	0.33
4	Blocplan	0.82

Table 6 shows us the R.Score of each alternative layout. The R.Score has the nature, bigger number means better. It is obvious that the bigger value is for the new version of Blocplan.

#### 4.5. Discussion

Result from the paper has a relative smaller value than the result from new version of Blocplan.

This is because:

1. Methodology of paper for improvement is placing the cell to a certain location and then calculate the R.Score with Blocplan
2. Blocplan ability of automatically searching a higher value of R.Score is not applied.

#### 5. Conclusion

- Blocplan is able to deal with the location size of workplace and keep it rectangular
- Several features that user need are available in this version of Blocplan such as: information about total area when department data form is on, average and standard deviation and department change one by one with pairwise exchange
- Result of the new version of blocplan is better than result of the paper due to the difference in method for applying the Blocplan

## 6. References

- [1] Badiru, Adedeji B. & Alaa Arif (2007). FLEXPART: facility layout expert system using fuzzy linguistic relationship codes. IIE Transactions Volume 28, 1996 - Issue 4. Abstract from: <https://www.tandfonline.com/doi/abs/10.1080/07408179608966277> (accessed: April 25<sup>th</sup> 2018)
- [2] Donaghey, Charles E., Christopher A. Chung & Hayian Kong (2007). Blocplan For Windows: Computer Aided Facility Layout Methodology And Enhancements. Proceedings of the 2007 Industrial Engineering Research Conference.
- [3] Goetschalckx, M (1992). An interactive layout heuristic based on hexagonal adjacency graphs. European Journal of Operational Research 63. 304-321
- [4] Heragu, Sunderesh S. (2016). Facility Design p. 124-125. CRC Press.
- [5] Leonardo, Hutahaean, H.A., & Wee, Hui-Ming (2014). Comparing alternative plant layouts based on Craft and Blocplan algorithms. Metris Journal Vol. 15. 55–64
- [6] M Tambunan, et al. (2018). Production facility layout by comparing moment displacement using BLOCPLAN and ALDEP Algorithms. IOP Conference Series: Materials Science and Engineering 309 012032.
- [7] Puspita, I.A., et al. (2015). Production facility layout design using blocplan algorithm. American Scientific Publishers.
- [8] Ahmadi, A., Pishvae, M. S., Jokar, M. R. A. (2017). A survey on multi-floor facility layout problems Computers & Industrial Engineering 107 (2017) 158–170.
- [9] Universidad ICESI. Facility Layout 5. Powerpoint material from <ftp://200.3.193.30/leonardo/DistPlanta/Slides/Layout-5-05166.ppt> (accessed March 9<sup>th</sup> 2018)
- [10] Kustriyanto, E., Pambuditama, I., Irawan, Y.S.(2016). Layout improvement of ammunition sleeves production machine with systematic layout planning method and blocplan. Mechanical Engineering Journal Vol.7, No.3 Year 2016: 103-112.

## APPENDIX

- Code for Figure 5

```

Me.Size = New Size(500, 625) : Me.CenterToScreen()
noPos = New Point(180, 30)
Dim lblNo As New Label With {.Text = "Number", .Location = noPos, .Width = 50}
deptPos = New Point(noPos.X + lblNo.Width + 10, noPos.Y)
Dim lblDept As New Label With {.Text = "Department", .Location = New
Point(deptPos)}
areaPos = New Point(deptPos.X + lblDept.Width + 10, noPos.Y)
Dim lblArea As New Label With {.Text = "Area", .Location = New Point(areaPos),
.Width = 50}
Me.Controls.Add(lblNo) : Me.Controls.Add(lblDept) : Me.Controls.Add(lblArea)
Dim i As Integer
For i = 1 To 18 'Create a new textbox and set its properties26.27.
txtNo(i) = New TextBox : txtDept(i) = New TextBox : txtArea(i) = New
TextBox
txtNo(i).Size = New Drawing.Size(50, 20) : txtDept(i).Size = New
Drawing.Size(100, 20) : txtArea(i).Size = txtNo(i).Size
txtNo(i).Location = New Point(noPos.X, noPos.Y + 25 * (i)) : txtNo(i).Text
= i.ToString : txtNo(i).ReadOnly = True
txtDept(i).Location = New Point(deptPos.X, deptPos.Y + 25 * (i)) :
txtDept(i).Name = "dept" & i
txtArea(i).Location = New Point(areaPos.X, areaPos.Y + 25 * (i)) :
txtArea(i).Name = i
Me.Controls.Add(txtNo(i)) : Me.Controls.Add(txtDept(i)) :
Me.Controls.Add(txtArea(i))
AddHandler txtArea(i).KeyPress, AddressOf area_keypressed
AddHandler txtArea(i).Leave, AddressOf area_textchanged
Next
'MessageBox.Show(deptPos.Y + 25 * i)
lblTotalPos = New Point(deptPos.X, deptPos.Y + 25 * i) : totalPos = New
Point(areaPos.X, areaPos.Y + 25 * i)
Dim lblTotal As New Label With {.Text = "Total", .Location = lblTotalPos,
.Width = 100, .TextAlign = ContentAlignment.MiddleRight}
txtTotal = New TextBox() With {.Location = totalPos, .Width = 100, .Text = ""}
Me.Controls.Add(lblTotal) : Me.Controls.Add(txtTotal)
' For object out of input department
problemPos = New Point(10, 100)
Dim lblProblem As New Label With {.Text = "New problem", .Location =
problemPos}
enterPos = New Point(problemPos.X, problemPos.Y + 220)
Dim lblEnter As New Label With {.Text = "Enter or modify problem data",
.Location = enterPos}
avgPos = New Point(problemPos.X, problemPos.Y + 360)
Dim lblAvg As New Label With {.Text = "Average", .Location = avgPos, .Width =
80}
txtAvg = New TextBox() With {.Location = New Point(avgPos.X + 90, avgPos.Y),
.Width = 60, .Height = 20}
devPos = New Point(problemPos.X, problemPos.Y + 390)
Dim lblDev As New Label With {.Text = "Std Dev. Area", .Location = devPos,
.Width = 80}
txtDev = New TextBox() With {.Location = New Point(devPos.X + 90, devPos.Y),
.Width = 60, .Height = 20}
Dim btnContinue As New Button With {.Location = New Point(devPos.X + 110,
devPos.Y + 50), .Width = 75, .Height = 30, .Text = "Continue"}
Dim btnPrint As New Button With {.Location = New Point(devPos.X + 210,
devPos.Y + 50), .Width = 75, .Height = 30, .Text = "Print"}
Dim btnBack As New Button With {.Location = New Point(devPos.X + 310, devPos.Y
+ 50), .Width = 75, .Height = 30, .Text = "Back"}
AddHandler btnContinue.Click, AddressOf continue_click

```

```

        Me.Controls.Add(lblProblem) : Me.Controls.Add(lblEnter) :
Me.Controls.Add(lblEnter) : Me.Controls.Add(lblAvg) : Me.Controls.Add(txtAvg)
        Me.Controls.Add(lblDev) : Me.Controls.Add(txtDev) :
Me.Controls.Add(btnContinue) : Me.Controls.Add(btnPrint) : Me.Controls.Add(btnBack)
End Sub

```

- Input for Area of department should be number, therefore we prevent an input that is not a number. Code for preventing non-numerical input is:

```

Private Sub area_keypressed(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs)
    If e.KeyChar <> ControlChars.Back Then
        Dim allowedChars As String = "0123456789.,"
        If allowedChars.IndexOf(e.KeyChar) = -1 Then
            ' Invalid Character
            e.Handled = True
        End If
    End If
End Sub

```

- In this form, we provide the total area of departments, average and standard deviation of them. The code for calculation is:

```

Private Sub area_textchanged(ByVal sender As Object, ByVal e As EventArgs)
    Dim area As TextBox = sender
    If txtDept(area.Name).Text = "" Then
        MessageBox.Show("Department Name must be filled")
        txtDept(area.Name).Focus()
    End If
    Dim total, totalSquare, stddev As Double : total = 0 : totalSquare = 0
    Dim count As Integer : count = 0
    For i As Integer = 1 To 18
        If txtArea(i).Text = "" Then

        Else
            total += Cdbl(txtArea(i).Text)
            totalSquare += Cdbl(txtArea(i).Text) ^ 2
            count += 1
        End If
    Next
    stddev = Math.Sqrt((totalSquare - (total * total / count)) / count)
    txtTotal.Text = Math.Round(total, 2)
    txtAvg.Text = Math.Round(total / count, 2)
    txtDev.Text = Math.Round(stddev, 2)
End Sub

```

- When a user has inputted the departments info and click continue button, he/she will see next form of adjacency score input. The code for Figure 6 is:

```

Private Sub continue_click(ByVal sender As Object, ByVal e As EventArgs)
    scoreForm.Show()
End Sub

```



The code for adjacency form is:

```

Me.Size = New Size(650, 560)
Dim firstHorizPoint As Point = New Point(firstPoint.X + 130, firstPoint.Y - 10)
    For k As Integer = 2 To count
        textNoH(k) = New TextBox() With {.Location = New Point(firstHorizPoint.X +
(k - 2) * 25, firstHorizPoint.Y), .Width = 20, .Text = k, .ReadOnly = True}
        Me.Controls.Add(textNoH(k))
    Next
    For i As Integer = 1 To count
        textNoV(i) = New TextBox() With {.Location = New Point(firstPoint.X,
firstPoint.Y + i * 20), .Width = 20, .Text = i}
        textDept(i) = New TextBox() With {.Location = New Point(firstPoint.X + 25,
firstPoint.Y + i * 20), .Width = 100, .Text = formDept.txtDept(i).Text}
        Me.Controls.Add(textNoV(i)) : Me.Controls.Add(textDept(i))
        For j As Integer = 2 To count
            textVal(i, j) = New TextBox() With {.Location = New Point(firstPoint.X
+ 130 + (j - 2) * 25, firstHorizPoint.Y + 10 + i * 20), .Width = 20, .MaxLength = 1,
.CharacterCasing = CharacterCasing.Upper}
            blk = New Label With {.Location = New Point(firstPoint.X + 130 + (j -
2) * 25, firstHorizPoint.Y + 10 + i * 20), .Width = 20, .Text = "..."}
            If i >= j Then
                Me.Controls.Add(blk)
            Else
                Me.Controls.Add(textVal(i, j))
                AddHandler textVal(i, j).KeyPress, AddressOf scoreVal_value
            End If
        Next
    Next
    enterCode = New Label() With {.Location = New Point(firstPoint.X + 25,
firstPoint.Y + 410), .Width = 1000, .Height = 25, _
        .Text = "Enter or change Code" & Space(19) &
"A=Absokutely" & Space(35) & "I=Immediately" & Space(35) & _
        "U=Unimportant" & Chr(13) & Space(54) &
"E=Essential" & Space(38) & "O=Ordinary" & Space(36) & "X=Undesirable"}
    Dim btnContinue As New Button With {.Location = New Point(firstPoint.X + 250,
firstPoint.Y + 460), .Width = 75, .Height = 30, .Text = "Continue"}
    AddHandler btnContinue.Click, AddressOf continue_click
    Dim btnPrint As New Button With {.Location = New Point(firstPoint.X + 350,
firstPoint.Y + 460), .Width = 75, .Height = 30, .Text = "Print"}
    Me.Controls.Add(enterCode) : Me.Controls.Add(btnContinue) :
Me.Controls.Add(btnPrint)
End Sub
Private Sub continue_click(ByVal sender As Object, ByVal e As EventArgs)
    'vectorValueForm.Show()
End Sub

Private Sub scoreVal_value(ByVal sender As Object, ByVal e As KeyPressEventArgs)
    If e.KeyChar <> ControlChars.Back Then
        Dim allowedChars As String = "aiueoxAIUEOX"
        If allowedChars.IndexOf(e.KeyChar) = -1 Then
            e.Handled = True
        End If
    End If
End Sub

```

- For adjacency score input, we prevent improper input so the user can only input letters: A, E, I, O, U which are capital letters. Whenever the user input it in lowercase, the program change it to uppercase automatically with this code:

```

If e.KeyChar <> ControlChars.Back Then
    Dim allowedChars As String = "aiueoAIUEOX"
    If allowedChars.IndexOf(e.KeyChar) = -1 Then
        e.Handled = True
    End If
End If

For j As Integer = 2 To count
    textVal(i, j) = New TextBox() With {.Location = New Point(firstPoint.X
+ 130 + (j - 2) * 25, firstHorizPoint.Y + 10 + i * 20), .Width = 20, .MaxLength = 1,
.CharacterCasing = CharacterCasing.Upper}
    blk = New Label With {.Location = New Point(firstPoint.X + 130 + (j -
2) * 25, firstHorizPoint.Y + 10 + i * 20), .Width = 20, .Text = "..."}
    If i >= j Then
        Me.Controls.Add(blk)
    Else
        Me.Controls.Add(textVal(i, j))
    End If
End If

```

- If we click edit score button, the score of each relationship codes are appeared. The code for Figure 7 is:

```

Public Class vectorValueForm
    Private scoreA, scoreE, scoreI, scoreO, scoreU, scoreX As TextBox
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Dim firstPos As Point = New Point(100, 50)
        Dim lblCode, lblAbs, lblEss, lblImp, lblOrd, lblUnim, lblUndes As Label
        Dim codeA, codeE, codeI, codeO, codeU, codeX As TextBox
        lblCode = New Label With {.Location = New Point(firstPos.X + 150, firstPos.Y),
.Width = 200, .Text = "Code" & Space(10) & "Score"}
        lblAbs = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 20),
.Width = 150, .Text = "Absolutely Essential"}
        lblEss = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 45),
.Width = 150, .Text = "Essential"}
        lblImp = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 70),
.Width = 150, .Text = "Important"}
        lblOrd = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 95),
.Width = 150, .Text = "Ordinary Importance"}
        lblUnim = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 120),
.Width = 150, .Text = "Unimportant"}
        lblUndes = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 145),
.Width = 150, .Text = "Undesirable"}
        Me.Controls.Add(lblCode) : Me.Controls.Add(lblAbs) : Me.Controls.Add(lblEss) :
Me.Controls.Add(lblImp)
        Me.Controls.Add(lblOrd) : Me.Controls.Add(lblUnim) : Me.Controls.Add(lblUndes)
        codeA = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
20), .Width = 25, .Text = "A", .ReadOnly = True}
        codeE = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
45), .Width = 25, .Text = "E", .ReadOnly = True}
        codeI = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
70), .Width = 25, .Text = "I", .ReadOnly = True}
        codeO = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
95), .Width = 25, .Text = "O", .ReadOnly = True}
        codeU = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
120), .Width = 25, .Text = "U", .ReadOnly = True}
        codeX = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
145), .Width = 25, .Text = "X", .ReadOnly = True}
        Me.Controls.Add(codeA) : Me.Controls.Add(codeE) : Me.Controls.Add(codeI) :
Me.Controls.Add(codeO) : Me.Controls.Add(codeU) : Me.Controls.Add(codeX)
    End Sub
End Class

```

```

        scoreA = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y +
20), .Width = 25, .Text = "10"}
        scoreE = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y +
45), .Width = 25, .Text = "5"}
        scoreI = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y +
70), .Width = 25, .Text = "2"}
        scoreO = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y +
95), .Width = 25, .Text = "1"}
        scoreU = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y +
120), .Width = 25, .Text = "0"}
        scoreX = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y +
145), .Width = 25, .Text = "-10"}
        Me.Controls.Add(scoreA) : Me.Controls.Add(scoreE) : Me.Controls.Add(scoreI) :
Me.Controls.Add(scoreO) : Me.Controls.Add(scoreU) : Me.Controls.Add(scoreX)
        Dim btnContinue As New Button With {.Location = New Point(firstPos.X + 30,
firstPos.Y + 250), .Width = 75, .Height = 35, .Text = "Continue"}
        Dim btnRestore As New Button With {.Location = New Point(firstPos.X + 130,
firstPos.Y + 250), .Width = 75, .Height = 35, .Text = "Restore Default"}
        Dim btnPrint As New Button With {.Location = New Point(firstPos.X + 230,
firstPos.Y + 250), .Width = 75, .Height = 35, .Text = "Print"}
        AddHandler btnRestore.Click, AddressOf restore_default
        Me.Controls.Add(btnContinue) : Me.Controls.Add(btnRestore) :
Me.Controls.Add(btnPrint)
    End Sub

    Private Sub restore_default(sender As Object, e As EventArgs)
        scoreA.Text = "10"
        scoreE.Text = "5"
        scoreI.Text = "2"
        scoreO.Text = "1"
        scoreU.Text = "0"
        scoreX.Text = "-10"
    End Sub

End Class

```

- User can edit the value of each relationship code. The code is:

```

Public Class vectorValue
    Shared firstPos As Point = New Point(100, 50)
    Dim WithEvents txtscoreA, txtscoreE, txtscoreI, txtscoreO, txtscoreU, txtscoreX As
TextBox
    Dim WithEvents btnContinue, btnRestore, btnSave, btnClose As Button
    Private Sub vectorValue_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Me.Size = New Size(450, 400)
        Dim lblCode, lblAbs, lblEss, lblImp, lblOrd, lblUnim, lblUndes As Label
        Dim codeA, codeE, codeI, codeO, codeU, codeX As TextBox
        lblCode = New Label With {.Location = New Point(firstPos.X + 150, firstPos.Y),
.Width = 200, .Text = "Code" & Space(10) & "Score"}
        lblAbs = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 20),
.Width = 150, .Text = "Absolutely Essential"}
        lblEss = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 45),
.Width = 150, .Text = "Essential"}
        lblImp = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 70),
.Width = 150, .Text = "Important"}
        lblOrd = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 95),
.Width = 150, .Text = "Ordinary Importance"}
        lblUnim = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 120),
.Width = 150, .Text = "Unimportant"}
        lblUndes = New Label With {.Location = New Point(firstPos.X, firstPos.Y + 145),
.Width = 150, .Text = "Undesirable"}
    End Sub
End Class

```

```

        Me.Controls.Add(lblCode) : Me.Controls.Add(lblAbs) : Me.Controls.Add(lblEss) :
Me.Controls.Add(lblImp)
        Me.Controls.Add(lblOrd) : Me.Controls.Add(lblUnim) : Me.Controls.Add(lblUndes)
        codeA = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
20), .Width = 25, .Text = "A", .ReadOnly = True}
        codeE = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
45), .Width = 25, .Text = "E", .ReadOnly = True}
        codeI = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
70), .Width = 25, .Text = "I", .ReadOnly = True}
        codeO = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
95), .Width = 25, .Text = "O", .ReadOnly = True}
        codeU = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
120), .Width = 25, .Text = "U", .ReadOnly = True}
        codeX = New TextBox With {.Location = New Point(firstPos.X + 155, firstPos.Y +
145), .Width = 25, .Text = "X", .ReadOnly = True}
        Me.Controls.Add(codeA) : Me.Controls.Add(codeE) : Me.Controls.Add(codeI) :
Me.Controls.Add(codeO) : Me.Controls.Add(codeU) : Me.Controls.Add(codeX)
        txtscoreA = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y
+ 20), .Width = 25, .Text = scoreForm.scoreA, .Name = "A"}
        txtscoreE = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y
+ 45), .Width = 25, .Text = scoreForm.scoreE, .Name = "E"}
        txtscoreI = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y
+ 70), .Width = 25, .Text = scoreForm.scoreI, .Name = "I"}
        txtscoreO = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y
+ 95), .Width = 25, .Text = scoreForm.scoreO, .Name = "O"}
        txtscoreU = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y
+ 120), .Width = 25, .Text = scoreForm.scoreU, .Name = "U"}
        txtscoreX = New TextBox With {.Location = New Point(firstPos.X + 210, firstPos.Y
+ 145), .Width = 25, .Text = scoreForm.scoreX, .Name = "X"}
        Me.Controls.Add(txtscoreA) : Me.Controls.Add(txtscoreE) :
Me.Controls.Add(txtscoreI) : Me.Controls.Add(txtscoreO) : Me.Controls.Add(txtscoreU) :
Me.Controls.Add(txtscoreX)
        Dim btnContinue As New Button With {.Location = New Point(firstPos.X, firstPos.Y
+ 250), .Width = 75, .Height = 35, .Text = "Continue"}
        Dim btnRestore As New Button With {.Location = New Point(firstPos.X + 85,
firstPos.Y + 250), .Width = 75, .Height = 35, .Text = "Restore Default"}
        Dim btnSave As New Button With {.Location = New Point(firstPos.X + 170,
firstPos.Y + 250), .Width = 75, .Height = 35, .Text = "Save"}
        AddHandler btnSave.Click, AddressOf save_score
        Dim btnClose As New Button With {.Location = New Point(firstPos.X + 255,
firstPos.Y + 250), .Width = 75, .Height = 35, .Text = "Close"}
        AddHandler btnClose.Click, AddressOf close_form
        AddHandler btnRestore.Click, AddressOf restore_default
        Me.Controls.Add(btnContinue) : Me.Controls.Add(btnRestore) :
Me.Controls.Add(btnSave) : Me.Controls.Add(btnClose)
    End Sub
    Private Sub restore_default(sender As Object, e As EventArgs)
        txtscoreA.Text = "10"
        txtscoreE.Text = "5"
        txtscoreI.Text = "2"
        txtscoreO.Text = "1"
        txtscoreU.Text = "0"
        txtscoreX.Text = "-10"
    End Sub
    Sub save_score(sender As Object, e As EventArgs) Handles btnSave.Click
        For i = 1 To scoreForm.count
            For j = 2 To scoreForm.count
                If i >= j Then
                    'nothing
                Else
                    Select Case scoreForm.textVal(i, j).Text
                        Case "A"

```

```

        scoreForm.scoreVal(i, j) = CInt(txtscoreA.Text)
    Case "E"
        scoreForm.scoreVal(i, j) = CInt(txtscoreE.Text)
    Case "I"
        scoreForm.scoreVal(i, j) = CInt(txtscoreI.Text)
    Case "O"
        scoreForm.scoreVal(i, j) = CInt(txtscoreO.Text)
    Case "U"
        scoreForm.scoreVal(i, j) = CInt(txtscoreU.Text)
    Case "X"
        scoreForm.scoreVal(i, j) = CInt(txtscoreX.Text)
    End Select
End If
Next
Next
scoreForm.scoreA = CInt(txtscoreA.Text)
scoreForm.scoreE = CInt(txtscoreE.Text)
scoreForm.scoreI = CInt(txtscoreI.Text)
scoreForm.scoreO = CInt(txtscoreO.Text)
scoreForm.scoreU = CInt(txtscoreU.Text)
scoreForm.scoreX = CInt(txtscoreX.Text)
MessageBox.Show("Score has been saved")
End Sub
Sub close_form(sender As Object, e As EventArgs) Handles btnClose.Click
    Me.Close()
End Sub

```

End Class

- Another new feature in this version of BLOCPLAN is its ability to show each value of relationship score, as shown in Figure 8. The code for Figure 8 is:

```

Public Class vectorScore
    Dim firstPoint As New Point(10, 20)
    Dim lblNo, lblDept, lblScore, lblTot As Label
    Dim textNo(18), textDept(18), textVal(18, 18), score(18), totVal(18) As TextBox
    'Private scoreVal(18, 18) As Integer
    Private thisCount As Integer = scoreForm.count
    Shared column, row As Integer
    Private Sub scoreForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Me.Size = New Size(450, 590)
        lblNo = New Label() With {.Location = New Point(firstPoint.X, firstPoint.Y),
        .Text = "No"}
        lblDept = New Label() With {.Location = New Point(firstPoint.X + 25,
        firstPoint.Y), .Text = "Department"}
        lblScore = New Label() With {.Location = New Point(firstPoint.X + 130,
        firstPoint.Y), .Text = "Score"}
        lblTot = New Label() With {.Location = New Point(firstPoint.X + 300,
        firstPoint.Y), .Text = "Value"}
        Me.Controls.Add(lblNo) : Me.Controls.Add(lblDept) : Me.Controls.Add(lblScore) :
        Me.Controls.Add(lblTot)
    End Sub

```

```

    For n = 1 To thisCount
        textNo(n) = New TextBox() With {.Location = New Point(firstPoint.X,
firstPoint.Y + n * 20), .Width = 20, .Text = n}
        textDept(n) = New TextBox() With {.Location = New Point(firstPoint.X + 25,
firstPoint.Y + n * 20), .Width = 100, .Text = formDept.txtDept(n).Text}
        score(n) = New TextBox() With {.Location = New Point(firstPoint.X + 130,
firstPoint.Y + n * 20), .Width = 150, .ReadOnly = True}
        totVal(n) = New TextBox() With {.Location = New Point(firstPoint.X + 300,
firstPoint.Y + n * 20), .Width = 50, .ReadOnly = True, .Text = 0}
        Me.Controls.Add(textNo(n)) : Me.Controls.Add(textDept(n))
        Me.Controls.Add(score(n)) : Me.Controls.Add(totVal(n))
    Next
    Dim firstHorizPoint As Point = New Point(firstPoint.X + 130, firstPoint.Y - 10)
    Dim btnContinue As New Button With {.Location = New Point(firstPoint.X + 100,
firstPoint.Y + 460), .Width = 75, .Height = 30, .Text = "Continue"}
    'Dim btnCount As New Button With {.Location = New Point(firstPoint.X + 200,
firstPoint.Y + 460), .Width = 75, .Height = 30, .Text = "Count Score"}
    AddHandler btnContinue.Click, AddressOf ratio_form
    Me.Controls.Add(btnContinue) 'Me.Controls.Add(btnCount)
'Me.Controls.Add(enterCode)
    count_click()
End Sub
Private Sub textVal_value(sender As Object, e As KeyPressEventArgs)
    If e.KeyChar <> ControlChars.Back Then
        Dim allowedChars As String = "aiueoXAIUEOX"
        If allowedChars.IndexOf(e.KeyChar) = -1 Then
            e.Handled = True
        End If
    End If
End Sub
Private Sub count_click()
    'Clearing the score and total value before counting
    For i = 1 To thisCount
        score(i).Text = ""
        totVal(i).Text = "0"
    Next
    'thisCount score Value for Columns
    Dim n As Integer
    For i = 1 To thisCount
        For j = 2 To thisCount
            If i >= j Then
                'nothing
            End If
        Next
    Next
End Sub

```

```

        Else
            For k = i To j
                If i = k Then
                    score(j).Text += scoreForm.scoreVal(i, j) & ","
                    n = Convert.ToInt32(totVal(j).Text)
                    n += scoreForm.scoreVal(i, j)
                    totVal(j).Text = n
                End If
            Next
        End If
    Next
Next
'Count score Value for Rows
For j = 2 To thisCount
    For i = 1 To thisCount
        If i >= j Then
            'nothing
        Else
            For k = i To j - 1
                If i = k Then
                    score(i).Text += scoreForm.scoreVal(i, j) & ","
                    n = Convert.ToInt32(totVal(i).Text)
                    n += scoreForm.scoreVal(i, j)
                    totVal(i).Text = n
                End If
            Next
        End If
    Next
End If
Next
Next
End Sub

Private Sub ratio_form(sender As Object, e As EventArgs)
    ratioForm.Show()
End Sub

End Class

```

- Area of which departments will be placed has a certain width and length. From BLOCPLAN DOS version, we find there are four selectable ratios for the area. However, user can input his own ratio with Specify Ratio selection. This feature is shown in Figure 9 and following is its code:

```

Public Class ratioForm
    Private firstPos As Point = New Point(20, 10)
    Private WithEvents btn135_1, btn1_1, btn2_1, btn1_2, btnSpy, btnBack, btnExe As
Button
    Private lblSel1, lblSel2, lblSel3, lblSel4, lblSel5, lblChoosen, lblRatioY,
lblRatioX, lblX As Label
    Public Shared ratioX, ratioY As String
    Private calib As Integer = 80
    Private Sub ratioForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Me.Size = New Size(500, 500) : ratioX = 0.0 : ratioY = 0.0
        Dim lblTitle As Label = New Label() With {.Location = New Point(firstPos.X +
100, firstPos.Y), .Width = 300, .Text = "Select a length (horizontal) to width (vertical)
ratio"}
        Dim lblSel1 As Label = New Label() With {.Location = New Point(firstPos.X + (1
* calib) / 2, firstPos.Y + 1 * calib + 60), .Width = 50, .Text = "Sel. 1", .Name =
"oke"}
        Dim lblSel2 As Label = New Label() With {.Location = New Point(firstPos.X + (1.5
* calib) / 2, firstPos.Y + 2 * calib + 140), .Width = 50, .Text = "Sel. 2"}
        Dim lblSel3 As Label = New Label() With {.Location = New Point(firstPos.X +
calib * 1.35 + 70 + (0.6 * calib) / 2, firstPos.Y + 1 * calib + 60), .Width = 50, .Text
= "Sel. 3"}
        Dim lblSel4 As Label = New Label() With {.Location = New Point(firstPos.X + 2 *
calib + 20 + (0.6 * calib) / 2, firstPos.Y + 3 * calib + 120), .Width = 50, .Text =
"Sel. 4"}
        Dim lblSel5 As Label = New Label() With {.Location = New Point(firstPos.X + 3 *
calib + 70 + (1 * calib) / 2, firstPos.Y + 2 * calib + 80), .Width = 50, .Text = "Sel.
5"}
        Me.Controls.Add(lblTitle) : Me.Controls.Add(lblSel1) : Me.Controls.Add(lblSel2)
: Me.Controls.Add(lblSel3) : Me.Controls.Add(lblSel4) : Me.Controls.Add(lblSel5)
        btn135_1 = New Button() With {.Location = New Point(firstPos.X, firstPos.Y +
50), .Size = New Size(1.35 * calib, 1 * calib), .Text = "1.35 x 1.00"}
        btn1_1 = New Button() With {.Location = New Point(firstPos.X + calib * 1.35 +
70, firstPos.Y + 50), .Size = New Size(1 * calib, 1 * calib), .Text = "1.00 x 1.00"}
        btn2_1 = New Button() With {.Location = New Point(firstPos.X, firstPos.Y + 1 *
calib + 130), .Size = New Size(2 * calib, 1 * calib), .Text = "2.00 x 1.00"}
        btn1_2 = New Button() With {.Location = New Point(firstPos.X + 2 * calib + 20,
firstPos.Y + 1 * calib + 110), .Size = New Size(1 * calib, 2 * calib), .Text = "1.00
x2.00"}
        btnSpy = New Button() With {.Location = New Point(firstPos.X + 3 * calib + 70,
firstPos.Y + 1 * calib + 70), .Size = New Size(1.5 * calib, 1 * calib), .Text = "Specify
Ratio"}
    End Sub
End Class

```



```

        btnBack = New Button() With {.Location = New Point(firstPos.X + 2 * calib + 20,
firstPos.Y + 3 * calib + 170), .Size = New Size(1 * calib, 30), .Text = "Back"}

        btnExe = New Button() With {.Location = New Point(firstPos.X + 2 * calib + 120,
firstPos.Y + 3 * calib + 170), .Size = New Size(1 * calib, 30), .Text = "Execute"}

        AddHandler btn1_1.Click, AddressOf btn_click

        Me.Controls.Add(btn135_1) : Me.Controls.Add(btn1_1) : Me.Controls.Add(btn2_1) :
Me.Controls.Add(btn1_2) : Me.Controls.Add(btnSpy) : Me.Controls.Add(btnBack) :
Me.Controls.Add(btnExe)

        lblChooosen = New Label() With {.Location = New Point(firstPos.X + 3 * calib +
50, firstPos.Y + 2 * calib + 130), .Width = 150, .Text = "Chooosen Ratio :", .Font = New
Font("Arial", 13, FontStyle.Bold)}

        lblRatioX = New Label() With {.Location = New Point(firstPos.X + 3 * calib +
60, firstPos.Y + 2 * calib + 170), .Width = 50, .Text = ratioX, .Font = New Font("Arial",
13, FontStyle.Bold)}

        lblX = New Label() With {.Location = New Point(firstPos.X + 3 * calib + 110,
firstPos.Y + 2 * calib + 170), .Width = 20, .Text = "X", .Font = New Font("Arial", 13,
FontStyle.Bold)}

        lblRatioY = New Label() With {.Location = New Point(firstPos.X + 3 * calib +
150, firstPos.Y + 2 * calib + 170), .Width = 50, .Text = ratioY, .Font = New Font("Arial",
13, FontStyle.Bold)}

        Me.Controls.Add(lblChooosen) : Me.Controls.Add(lblRatioX) :
Me.Controls.Add(lblX) : Me.Controls.Add(lblRatioY)

    End Sub

    Sub close_form() Handles btnBack.Click
        Me.Close()
    End Sub

    Sub execute_rectangle() Handles btnExe.Click
        ratioX = CSng(lblRatioX.Text) : ratioY = CSng(lblRatioY.Text)
        'areaForm.Show()
        If (ratioX = 0 Or ratioY = 0) Then
            MessageBox.Show("Area Ratio must be specified")
        Else
            layoutForm.Show()
        End If
    End Sub

    Sub btn_click(sender As Object, e As EventArgs) Handles btn1_1.Click, btn1_2.Click,
btn135_1.Click, btn2_1.Click
        Dim btn As Button = sender
        Dim str = btn.Text.Split("x") : Dim ratioX = Replace(str(0), ".", ",") : Dim
ratioY = Replace(str(1), ".", ",")
        lblRatioX.Text = ratioX
        lblRatioY.Text = ratioY

```

```
        'MessageBox.Show(ratioX.ToString)
    End Sub
    Sub spy_click(sender As Object, e As EventArgs) Handles btnSpy.Click
        Dim InputX As String = InputBox("Input length(horizontal) ratio :") : ratioX =
    CSng(Replace(InputX, ".", ","))
        Dim InputY As String = InputBox("Input widht(vertical) ratio :") : ratioY =
    CSng(Replace(InputY, ".", ","))
        lblRatioX.Text = ratioX
        lblRatioY.Text = ratioY
    End Sub
End Class
```