Q1

a) Since every row, every column and both diagonals have to be equal size and equal total amount, we can represent each of them in a vector. This is a vector of length n^2 *= 9.* For given example for the homework, the solution representation as vector can be as follows:

S(i)	2	7	6	9	5	1	4	3	8

The reason it is represented as an array that every cell must be distinct. So it is easy to generate an n^2 with unique numbers. Also it is a magic square, summation of all rows, columns and diagonals can easily computed with dynamic indexes.

b) It is known that a summation of all columns, rows and diagonals can be calculated as:

$$M(n) = \frac{(n^3 + n)}{2}$$

Let s_i be solution vector i, main aim is to satisfy:

$$|M(n) - sum(s_i)| = 0$$

Therefore, fitness function is:

$$\sum_i^S |M(n) - sum(s_i)|$$

c) We can select Tournament Selection strategy. Since our aim is to minimize the fitness value, (the length of a solution vector grows polynomially, and total number of permutations is huge even for *n = 3*, which is *9!*) we can compare K solutions randomly selected, and select best one of them based on their fitness value.

d) Mutation: We can randomly mutate two genes. Basically swapping positions of two genes. There will be no feasibility problem since we are just swapping two genes, every element will be unique.

Crossover: It is required to use a permutation based crossover operator since magic square does not allow duplicate numbers. *Order Crossover Operator(OX)* builds by choosing sub-tour of parent and preserve the order of the other parents.

For example:

P_1 = (2 7 6 | 9 5 1 | 4 3 8)

P_2 = (3 1 8 | 6 2 4 | 5 9 7)

After this, we mask the other parts of the parents except the middle. Starting from the second cut point of a parent, select numbers respectively, excluding the middle part of other parent.

If we select P_1, from the second cut point, we should select in the order:

4,3,8,2,7,6,9,5,1

After we remove 6, 2 and 4, since they are in the middle of other parent, the order is:

3,8,7,9,5,1

The rest is placed starting from second cut point. So, the P_1 and P_2 becomes:

P_1 = (6 2 4 | 9 5 1 | 7 3 8)

P_2 = (9 5 1 | 6 2 4 | 3 8 7)

Q2

Setting the initial Temperatures by preliminary experiments is as follows:

- Calculate the average difference between the objective function values at the beginning of the search (say, $\bar{\boldsymbol{\delta}}$).
- Initialize T_0 such that accepting rate of non-improving solutions is in a specified interval (e.g., 90% in this case).

So the formula is,

$$\boldsymbol{e}^{-\frac{\bar{\boldsymbol{\delta}}}{T_0}} \approx 0.\mathbf{9}$$

The absolute average difference of the objective function values of non-improving solutions is 72.5. So the initial temperature value is around 700.

Q3

Improved Simulated Annealing Algorithm Solving for 0/1 Knapsack Problem

a) 0-1 Knapsack Problem is a kind of combinatorial problem. There are many solving methods in literature, including exact and heuristic methods. Due to its complexity, large dimensions of this problem can be solved by heuristics. In this paper, the problem solved by a main algorithm which is Simulated Annealing. It is based on annealing process in physics, using hill-climbing moves to escape local optima. It is a stochastic algorithm which based on probability. Although it is popular for large-sized knapsack problems, sometimes it is not preferrable due to stucking in local optima, or consuming large amount of time on finding a promising feasible solution.

b) The main disadvantage of the SA is that it searches all the space based on randomity and mostly the solutions are useless in terms of feasibility and time. The paper tries to remove the useless solutions before applying SA. The algorithm basically selects n objects with their weight, m, and same weighted objects goes into same class. It is assumed that there exist at most n' (n'<n) objects with different weights in same class. Each weight class n', includes m_i objects with index which determined by profit. More profit means smaller index. If the number of elements in weight class exceeds c / n_i', only consider first objects into SA. The rest of the objects don't selected since that the aim is to maximize profit.
The second disadvantage is mentioned in the paper is that the metropolis rule cause to prior solution loss. Authers suggest that by setting up several variables on saving the best solution, profits and weights' sum up to this iteration. This guarantees on obtaining the best solution on annealing process rather than normal SA.

c) The experiment runs 10 times and they took average computation time results. It shows that improved algorithm has better approximation on optimization profit solutions. The algorithm also have the best computation time overhead. As shown in the figure, Improved SA is better

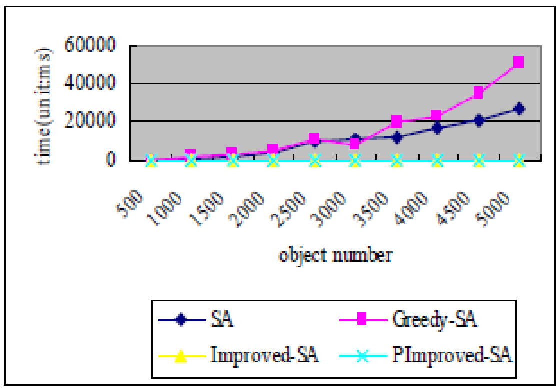

Figure 2. Computation time contrast curve

than normal SA and Greedy-SA in terms of computational time. Also, as can be shown in table 1, Improved SAs are generally outperforming normal SA in terms of profit solutions. Note that Improved and PImproved solutions can be seen as same which they can be controlled by a coefficient *k,* which is 1 in this case.

Table 1. Profit solutions of four algorithms

object number	SA	Greedy -SA	Improved -SA	PImproved -SA
500	15617	15688	15891	15835
1000	22037	22621	22817	22817
1500	29386	29609	30034	30019
2000	28364	28566	28616	28840
2500	31587	31421	31545	31545
3000	39017	39016	39353	39032
3500	42910	42669	43052	43284
4000	38916	38698	39191	39255
4500	42810	43412	43233	43571
5000	45998	46020	46746	47362