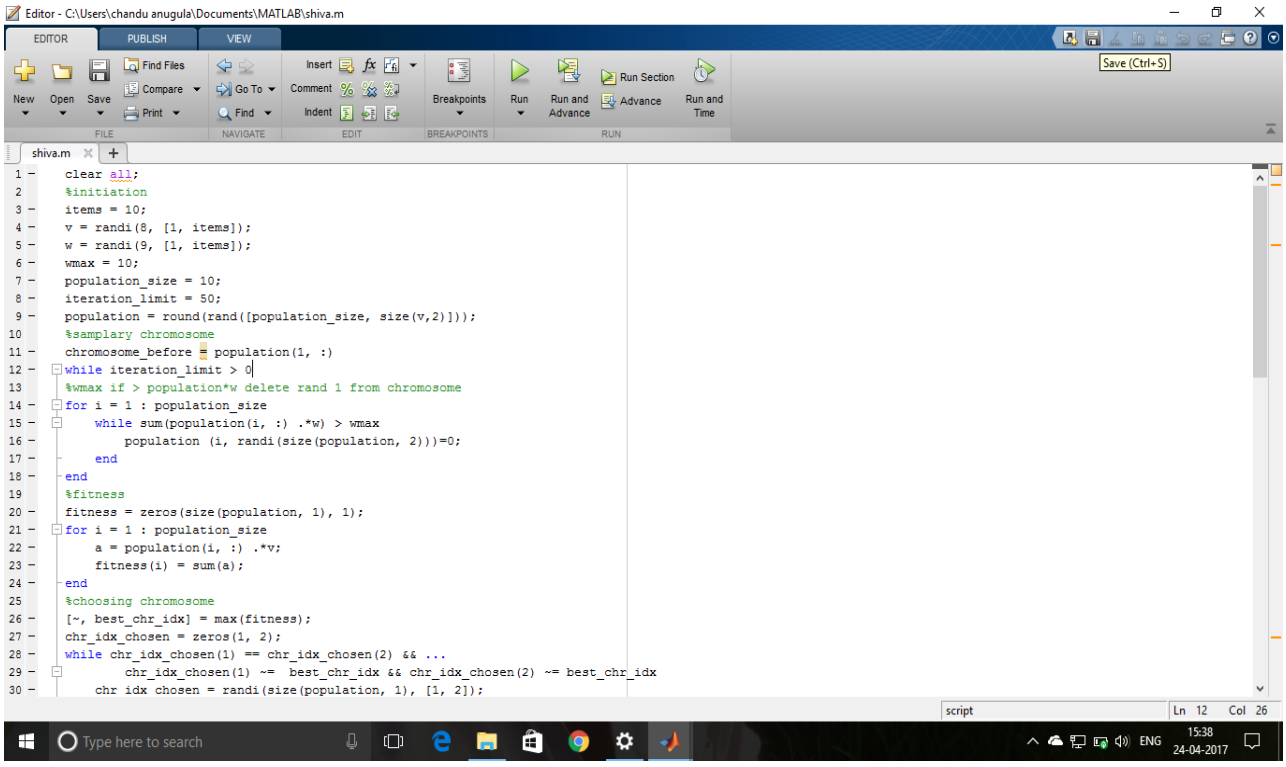
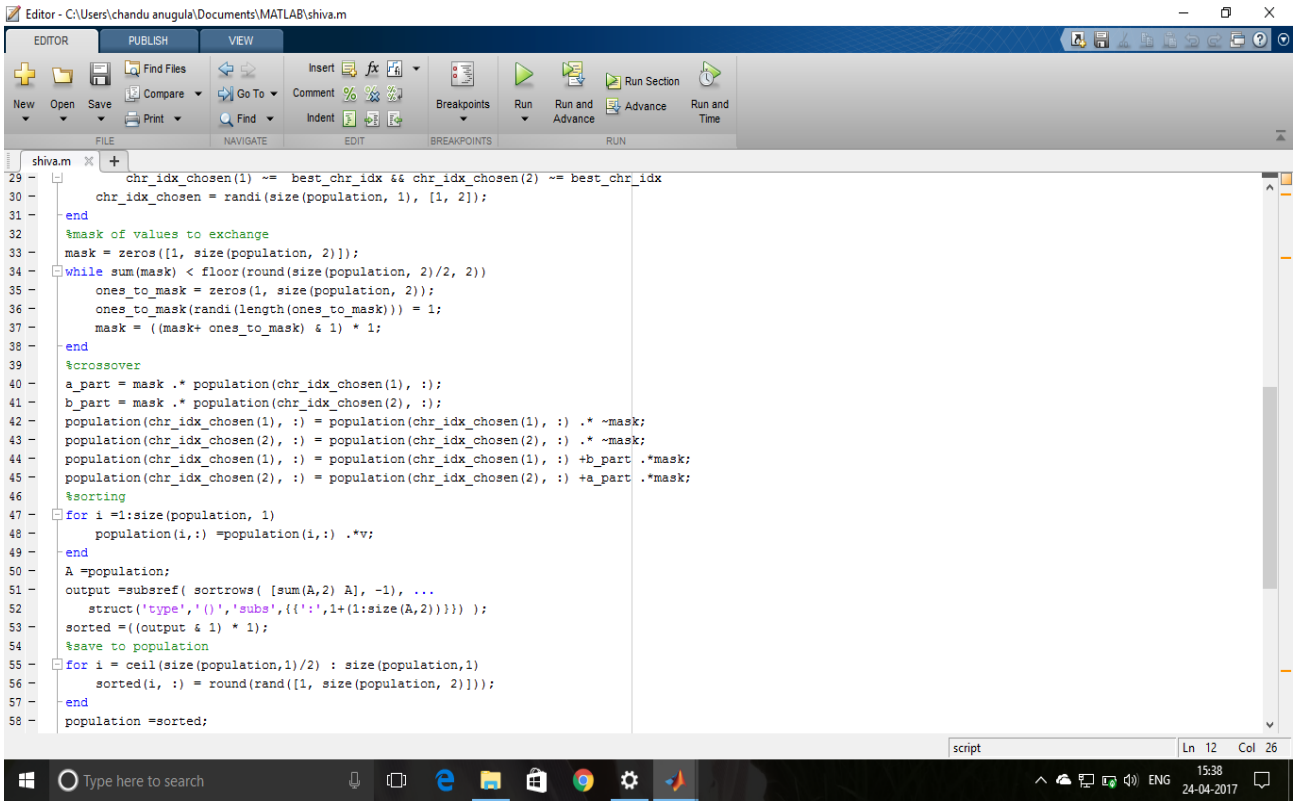
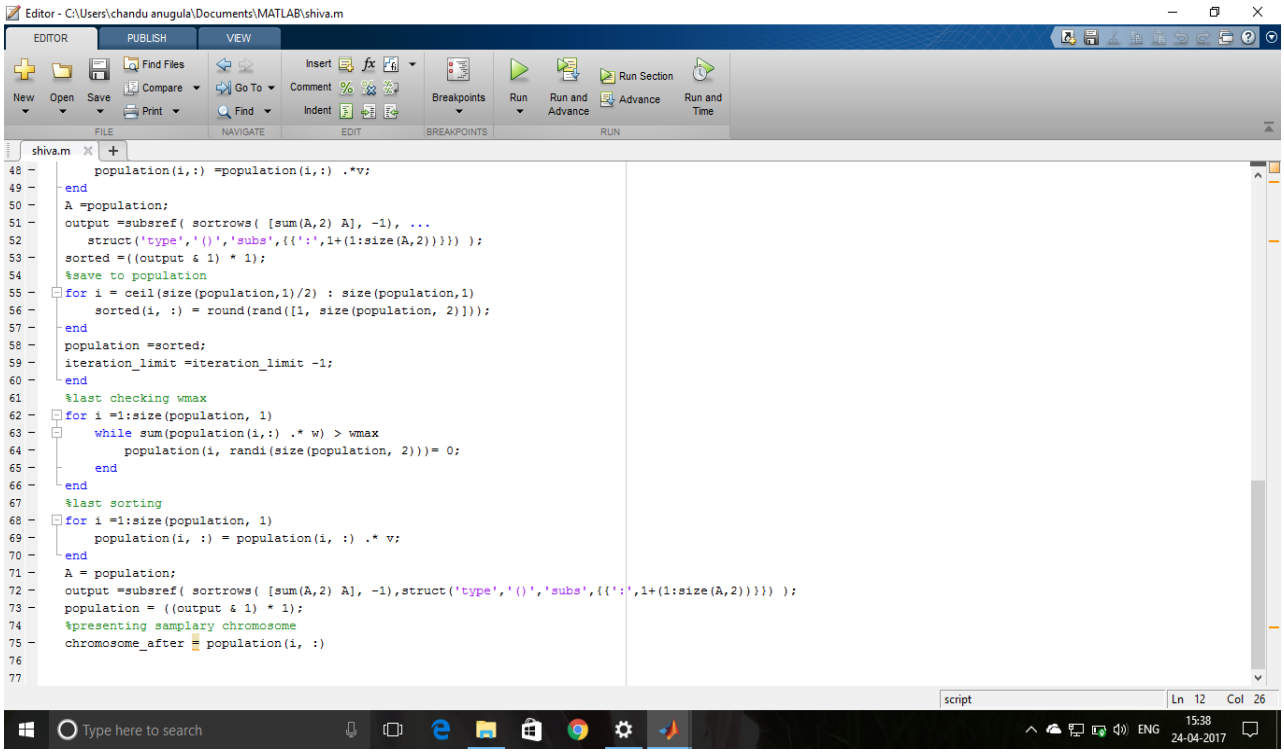
REPORT ON GENETIC ALGORITHM

CODE:



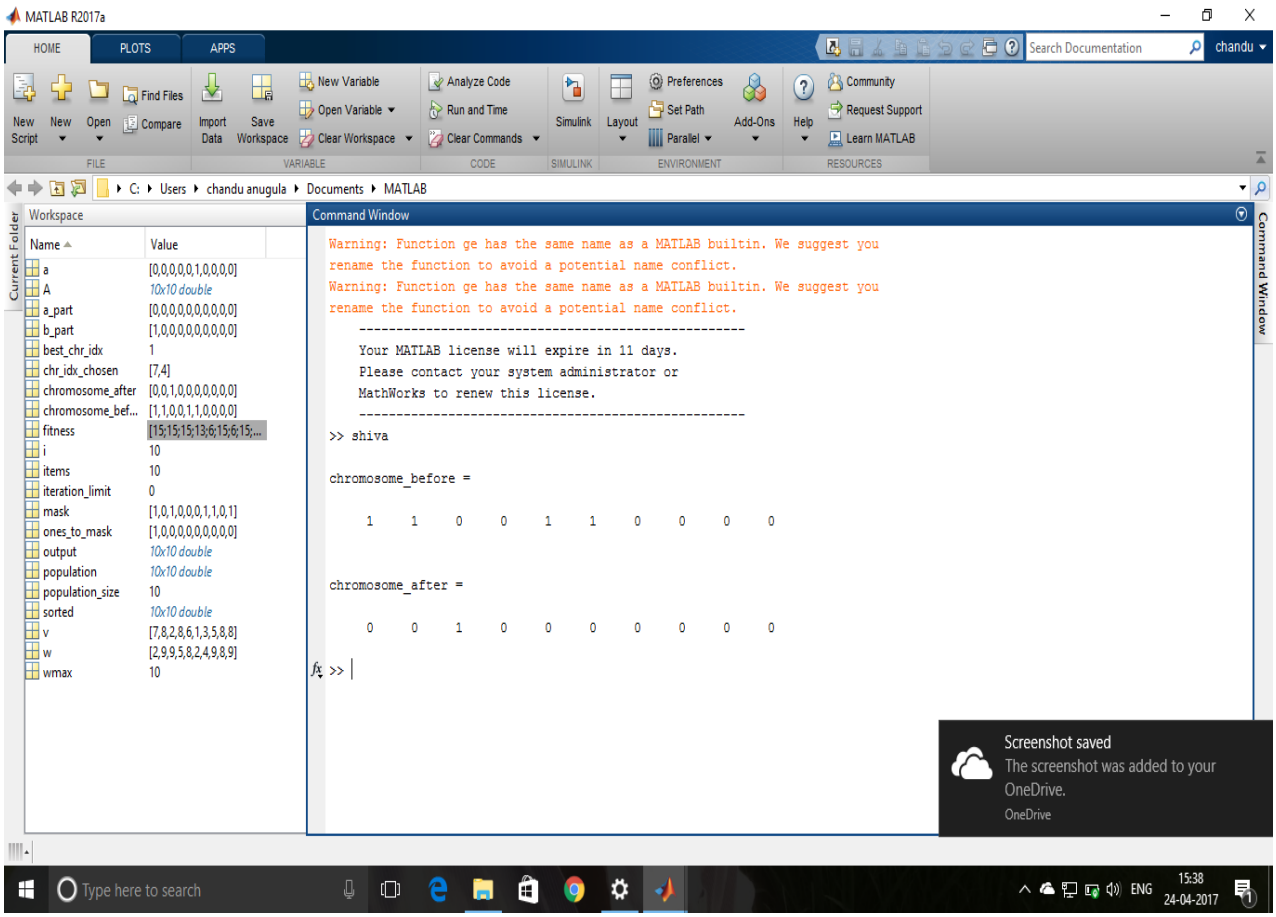





```
clear all;
%initiation
items = 10;
v = randi(8, [1, items]);
w = randi(9, [1, items]);
wmax = 10;
population_size = 10;
iteration_limit = 50;
population = round(rand([population_size, size(v,2)]));
%samplary chromosome
chromosome_before = population(1, :)
while iteration_limit > 0
%wmax if > population*w delete rand 1 from chromosome
for i = 1 : population_size
    while sum(population(i, :) .*w) > wmax
        population (i, randi(size(population, 2)))=0;
    end
end
%fitness
fitness = zeros(size(population, 1), 1);
for i = 1 : population_size
    a = population(i, :) .*v;
    fitness(i) = sum(a);
end
%choosing chromosome
[~, best_chr_idx] = max(fitness);
chr_idx_chosen = zeros(1, 2);
while chr_idx_chosen(1) == chr_idx_chosen(2) && ...
        chr_idx_chosen(1) ~=  best_chr_idx && chr_idx_chosen(2) ~=
best_chr_idx
    chr_idx_chosen = randi(size(population, 1), [1, 2]);
end
%mask of values to exchange
```

```
mask = zeros([1, size(population, 2)]);
while sum(mask) < floor(round(size(population, 2)/2, 2))
    ones_to_mask = zeros(1, size(population, 2));
    ones_to_mask(randi(length(ones_to_mask))) = 1;
    mask = ((mask+ ones_to_mask) & 1) * 1;
end
%crossover
a_part = mask .* population(chr_idx_chosen(1), :);
b_part = mask .* population(chr_idx_chosen(2), :);
population(chr_idx_chosen(1), :) = population(chr_idx_chosen(1), :) .*
~mask;
population(chr_idx_chosen(2), :) = population(chr_idx_chosen(2), :) .*
~mask;
population(chr_idx_chosen(1), :) = population(chr_idx_chosen(1), :) +b_part
.*mask;
population(chr_idx_chosen(2), :) = population(chr_idx_chosen(2), :) +a_part
.*mask;
%sorting
for i =1:size(population, 1)
    population(i,:) =population(i,:) .*v;
end
A =population;
output =subsref( sortrows( [sum(A,2) A], -1), ...
   struct('type','()','subs',{{':',1+(1:size(A,2))}}) );
sorted =((output & 1) * 1);
%save to population
for i = ceil(size(population,1)/2) : size(population,1)
    sorted(i, :) = round(rand([1, size(population, 2)]));
end
population =sorted;
iteration_limit =iteration_limit -1;
end
%last checking wmax
for i =1:size(population, 1)
    while sum(population(i,:) .* w) > wmax
        population(i, randi(size(population, 2)))= 0;
    end
end
%last sorting
for i =1:size(population, 1)
    population(i, :) = population(i, :) .* v;
end
A = population;
output =subsref( sortrows( [sum(A,2) A], -
1),struct('type','()','subs',{{':',1+(1:size(A,2))}}) );
population = ((output & 1) * 1);
%presenting samplary chromosome
chromosome_after = population(i, :)
```

RESULT:



chromosome_before =

1 1 0 0 1 1 0 0 0 0

chromosome_after =

0 0 1 0 0 0 0 0 0 0

CONCLUSION:

The various functions which includes crossover operation , fitness scaling.

Genetic Algorithm plays a important role for obtaining an optimized solution and to find the best fitness values. As we increase the population size chromosome and fitness values with respect to population size.