***NAME : YASH GHULE***

***ROLL NO : 2193289***

***CLASS : CSE CORE 2***

***ASSINGMENT – 4***

4. IMPLEMENT TRAVELLING SALEMAN ALGORITHM .

#include <bits/stdc++.h>

#include <limits.h>

using namespace std;

#define V 5

#define GENES ABCDE

#define START 0

#define POP\_SIZE 10

struct individual {

string gnome;

int fitness;

};

int rand\_num(int start, int end)

{

int r = end - start;

int rnum = start + rand() % r;

return rnum;

}

bool repeat(string s, char ch)

{

for (int i = 0; i < s.size(); i++) {

if (s[i] == ch)

return true;

}

return false;

}

string mutatedGene(string gnome)

{

while (true) {

int r = rand\_num(1, V);

int r1 = rand\_num(1, V);

if (r1 != r) {

char temp = gnome[r];

gnome[r] = gnome[r1];

gnome[r1] = temp;

break;

}

}

return gnome;

}

string create\_gnome()

{

string gnome = "0";

while (true) {

if (gnome.size() == V) {

gnome += gnome[0];

break;

}

int temp = rand\_num(1, V);

if (!repeat(gnome, (char)(temp + 48)))

gnome += (char)(temp + 48);

}

return gnome;

}

int cal\_fitness(string gnome)

{

int map[V][V] = { { 0, 2, INT\_MAX, 12, 5 },

{ 2, 0, 4, 8, INT\_MAX },

{ INT\_MAX, 4, 0, 3, 3 },

{ 12, 8, 3, 0, 10 },

{ 5, INT\_MAX, 3, 10, 0 } };

int f = 0;

for (int i = 0; i < gnome.size() - 1; i++) {

if (map[gnome[i] - 48][gnome[i + 1] - 48] == INT\_MAX)

return INT\_MAX;

f += map[gnome[i] - 48][gnome[i + 1] - 48];

}

return f;

}

int cooldown(int temp)

{

return (90 \* temp) / 100;

}

bool lessthan(struct individual t1,

struct individual t2)

{

return t1.fitness < t2.fitness;

}

void TSPUtil(int map[V][V])

{

int gen = 1;

int gen\_thres = 5;

vector<struct individual> population;

struct individual temp;

for (int i = 0; i < POP\_SIZE; i++) {

temp.gnome = create\_gnome();

temp.fitness = cal\_fitness(temp.gnome);

population.push\_back(temp);

}

cout << "\nInitial population: " << endl

<< "GNOME FITNESS VALUE\n";

for (int i = 0; i < POP\_SIZE; i++)

cout << population[i].gnome << " "

<< population[i].fitness << endl;

cout << "\n";

bool found = false;

int temperature = 10000;

while (temperature > 1000 && gen <= gen\_thres) {

sort(population.begin(), population.end(), lessthan);

cout << "\nCurrent temp: " << temperature << "\n";

vector<struct individual> new\_population;

for (int i = 0; i < POP\_SIZE; i++) {

struct individual p1 = population[i];

while (true) {

string new\_g = mutatedGene(p1.gnome);

struct individual new\_gnome;

new\_gnome.gnome = new\_g;

new\_gnome.fitness = cal\_fitness(new\_gnome.gnome);

if (new\_gnome.fitness <= population[i].fitness) {

new\_population.push\_back(new\_gnome);

break;

}

else {

float prob = pow(2.7,

-1 \* ((float)(new\_gnome.fitness

- population[i].fitness)

/ temperature));

if (prob > 0.5) {

new\_population.push\_back(new\_gnome);

break;

}

}

}

}

temperature = cooldown(temperature);

population = new\_population;

cout << "Generation " << gen << " \n";

cout << "GNOME FITNESS VALUE\n";

for (int i = 0; i < POP\_SIZE; i++)

cout << population[i].gnome << " "

<< population[i].fitness << endl;

gen++;

}

}

int main()

{

int map[V][V] = { { 0, 2, INT\_MAX, 12, 5 },

{ 2, 0, 4, 8, INT\_MAX },

{ INT\_MAX, 4, 0, 3, 3 },

{ 12, 8, 3, 0, 10 },

{ 5, INT\_MAX, 3, 10, 0 } };

TSPUtil(map);

}

OUTPUT :











