INDEX

S.NO	PROGRAMS	SIGN
1.	Write a program to implement Breadth First and Depth First Search.	
2.	Write a Program for the Best First Search and A* search algorithm.	
3.	Write a program to implement Water Jug Problem.	
4.	Write a program to implement 4- Problem.	
5.	Write a program to implement AO* algorithm.	
6.	Write a program to implement hill climbing & steepest ascent hill climbing algorithm.	
7.	Write a program to implement Travelling Salesman Problem.	
8.	Write a program to implement Genetic Algorithm for different types of gene representation.	

PROGRAMS

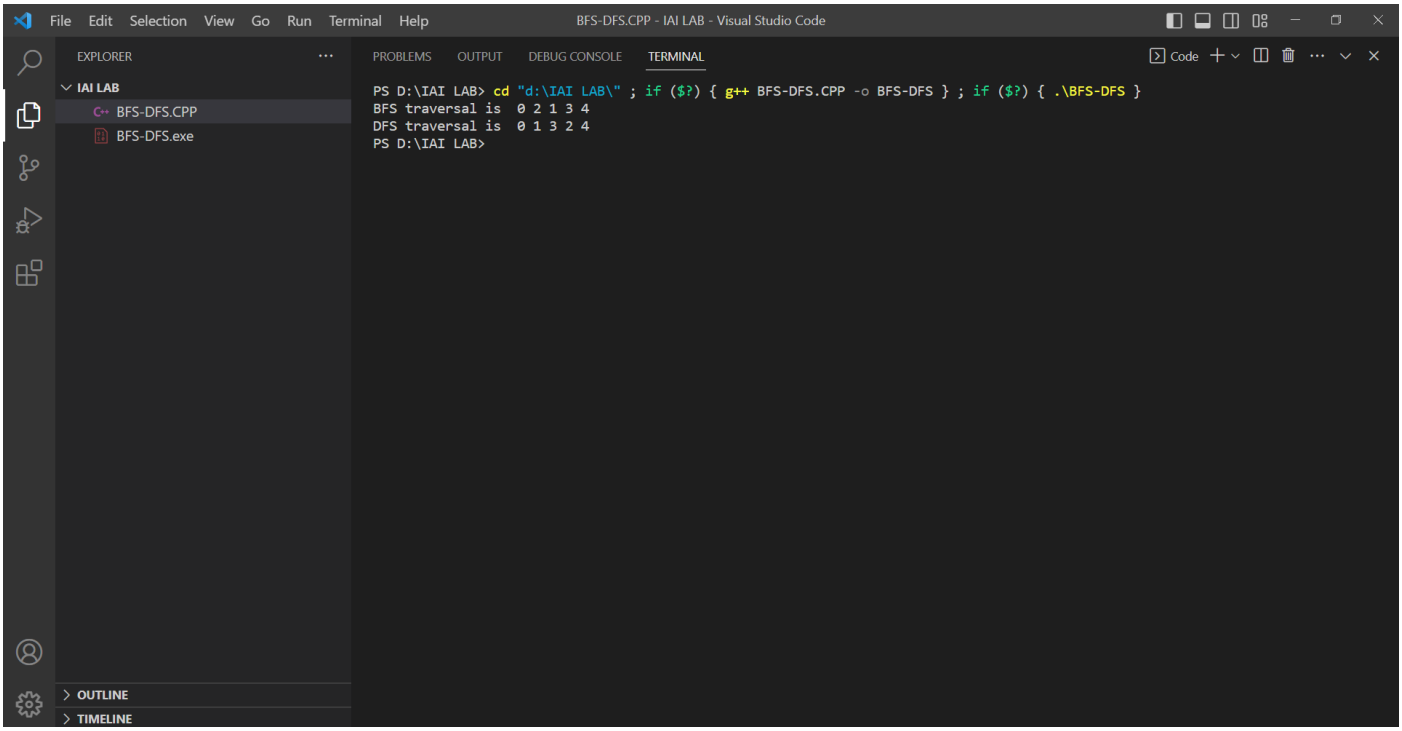
1. Write a program to implement Breadth First and Depth First Search.

CODE: -

```
#include <iostream>
#include <vector>
#include <queue>
#include <stack>
using namespace std;
void edge(vector<int> adj[], int u, int v)
{
   adj[u].push_back(v);
}
void bfs(int s, vector<int> adj[], bool visit[])
{
   queue<int> q;
   q.push(s);
   visit[s] = true;
   while (!q.empty())
   {
      int u = q.front();
      cout << u << " ";
      q.pop();
      for (int i = 0; i < adj[u].size(); i++)
      {
         if (!visit[adj[u][i]])
         {
            q.push(adj[u][i]);
            visit[adj[u][i]] = true;
         }
      }
   }
}
void dfs(int s, vector<int> adj[], bool visit[])
{
   stack<int> stk;
   stk.push(s);
   visit[s] = true;
   while (!stk.empty())
   {
      int u = stk.top();
      cout << u << " ";
      stk.pop();
      for (int i = 0; i < adj[u].size(); i++)
      {
         if (!visit[adj[u][i]])
         {
            stk.push(adj[u][i]);
            visit[adj[u][i]] = true;
         }
```

```
        }
    }
}
int main()
{
    vector<int> adj[5];
    bool visit[5];
    for (int i = 0; i < 5; i++)
    {
        visit[i] = false;
    }
    edge(adj, 0, 2);
    edge(adj, 0, 1);
    edge(adj, 1, 3);
    edge(adj, 2, 0);
    edge(adj, 2, 3);
    edge(adj, 2, 4);
    cout << "BFS traversal is"
         << "  ";
    bfs(0, adj, visit);
    cout << endl;
    for (int i = 0; i < 5; i++)
    {
        visit[i] = false;
    }
    cout << "DFS traversal is"
         << "  ";
    dfs(0, adj, visit);
}
```

OUTPUT: -


File Edit Selection View Go Run Terminal Help
BFS-DFS.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
BFS-DFS.CPP
BFS-DFS.exe
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
PS D:\IAI LAB> cd "d:\IAI LAB\" ; if ($?) { g++ BFS-DFS.CPP -o BFS-DFS } ; if ($?) { .\BFS-DFS }
BFS traversal is 0 2 1 3 4
DFS traversal is 0 1 3 2 4
PS D:\IAI LAB>
OUTLINE
TIMELINE

2. Write a Program for the Best First Search and A* search algorithm.

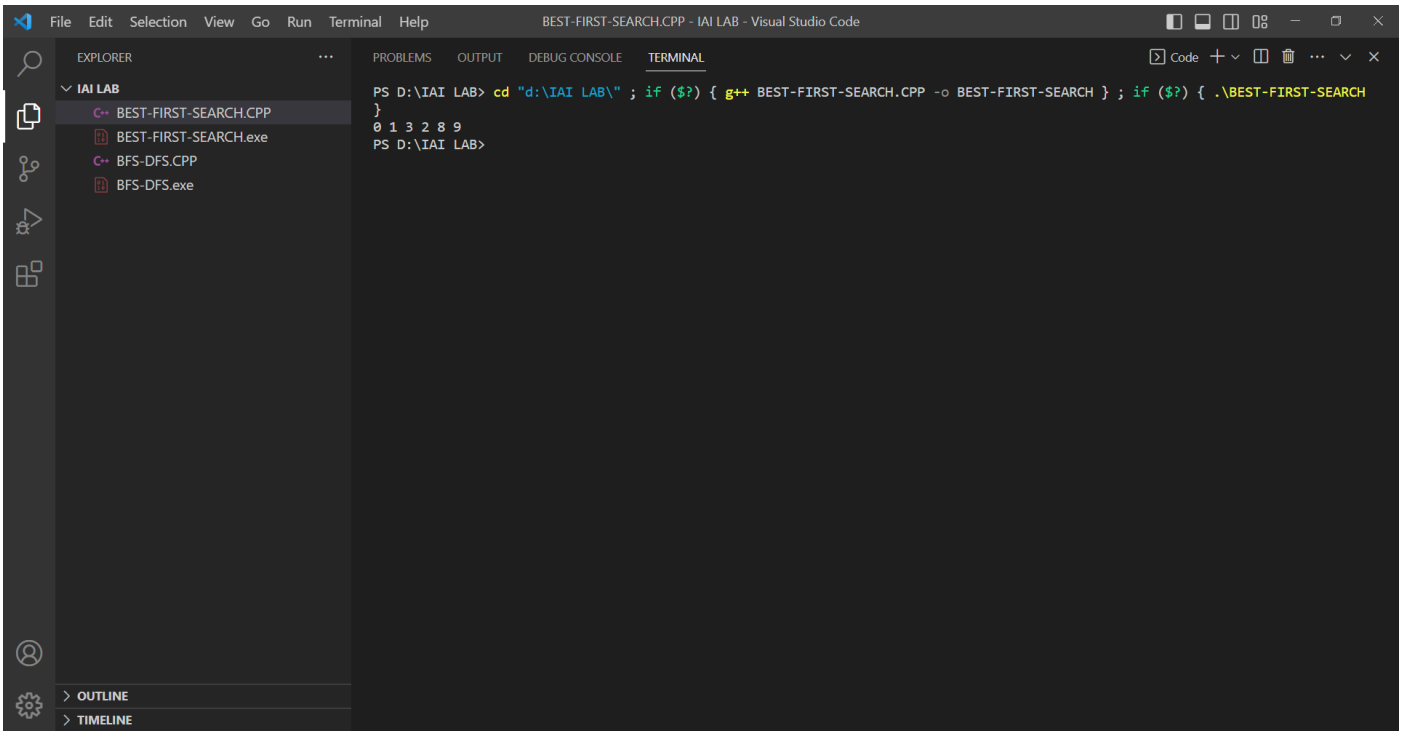
CODE: -

(I).
```
#include <bits/stdc++.h>
using namespace std;
typedef pair<int, int> pi;

vector<vector<pi>> graph;

void addedge(int x, int y, int cost)
{
   graph[x].push_back(make_pair(cost, y));
   graph[y].push_back(make_pair(cost, x));
}

void best_first_search(int actual_Src, int target, int n)
{
   vector<bool> visited(n, false);

   priority_queue<pi, vector<pi>, greater<pi>> pq;

   pq.push(make_pair(0, actual_Src));
   int s = actual_Src;
   visited[s] = true;
   while (!pq.empty())
   {
      int x = pq.top().second;

      cout << x << " ";
      pq.pop();
      if (x == target)
         break;

      for (int i = 0; i < graph[x].size(); i++)
      {
         if (!visited[graph[x][i].second])
         {
            visited[graph[x][i].second] = true;
            pq.push(make_pair(graph[x][i].first, graph[x][i].second));
         }
      }
   }
}

int main()
{
   int v = 14;
   graph.resize(v);

   addedge(0, 1, 3);
```

```
    addedge(0, 2, 6);
    addedge(0, 3, 5);
    addedge(1, 4, 9);
    addedge(1, 5, 8);
    addedge(2, 6, 12);
    addedge(2, 7, 14);
    addedge(3, 8, 7);
    addedge(8, 9, 5);
    addedge(8, 10, 6);
    addedge(9, 11, 1);
    addedge(9, 12, 10);
    addedge(9, 13, 2);

    int source = 0;
    int target = 9;

    best_first_search(source, target, v);

    return 0;
}
```

OUTPUT:

(I).


File Edit Selection View Go Run Terminal Help
BEST-FIRST-SEARCH.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
BEST-FIRST-SEARCH.CPP
BEST-FIRST-SEARCH.exe
BFS-DFS.CPP
BFS-DFS.exe
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
PS D:\IAI LAB> cd "d:\IAI LAB\" ; if ($?) { g++ BEST-FIRST-SEARCH.CPP -o BEST-FIRST-SEARCH } ; if ($?) { .\BEST-FIRST-SEARCH
}
0 1 3 2 8 9
PS D:\IAI LAB>

(II).

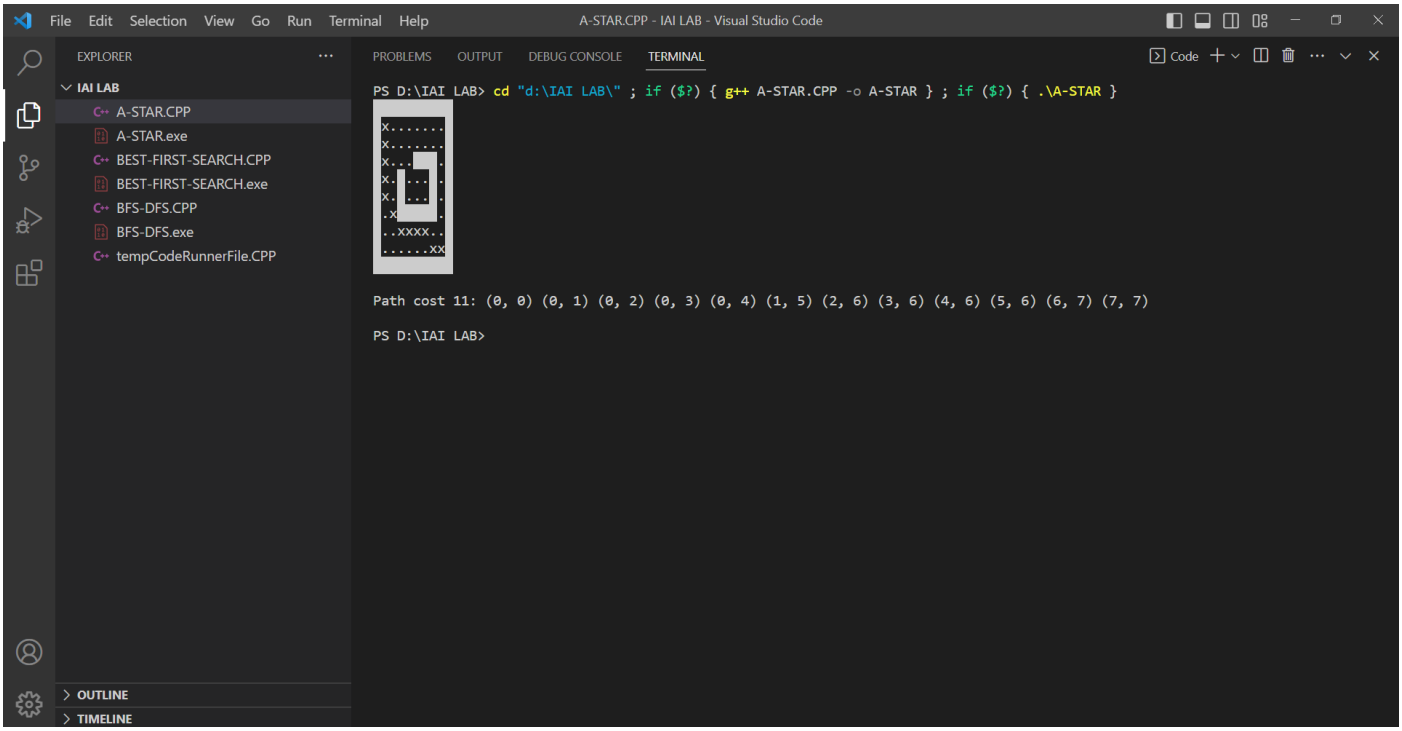
```
#include <list>
#include <algorithm>
#include <iostream>

class point {
public:
   point( int a = 0, int b = 0 ) { x = a; y = b; }
   bool operator ==( const point& o ) { return o.x == x && o.y == y; }
   point operator +( const point& o ) { return point( o.x + x, o.y + y ); }
   int x, y;
};

class map {
public:
   map() {
      char t[8][8] = {
         {0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0},
         {0, 0, 0, 0, 1, 1, 1, 0}, {0, 0, 1, 0, 0, 0, 1, 0},
         {0, 0, 1, 0, 0, 0, 1, 0}, {0, 0, 1, 1, 1, 1, 1, 0},
         {0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0}
      };
      w = h = 8;
      for( int r = 0; r < h; r++ )
         for( int s = 0; s < w; s++ )
            m[s][r] = t[r][s];
   }
   int operator() ( int x, int y ) { return m[x][y]; }
   char m[8][8];
   int w, h;
};

class node {
public:
   bool operator == (const node& o ) { return pos == o.pos; }
   bool operator == (const point& o ) { return pos == o; }
   bool operator < (const node& o ) { return dist + cost < o.dist + o.cost; }
   point pos, parent;
   int dist, cost;
};

class aStar {
public:
   aStar() {
      neighbours[0] = point( -1, -1 ); neighbours[1] = point(  1, -1 );
      neighbours[2] = point( -1,  1 ); neighbours[3] = point(  1,  1 );
      neighbours[4] = point(  0, -1 ); neighbours[5] = point( -1,  0 );
      neighbours[6] = point(  0,  1 ); neighbours[7] = point(  1,  0 );
   }

   int calcDist( point& p ){
```

```
    // need a better heuristic
    int x = end.x - p.x, y = end.y - p.y;
    return( x * x + y * y );
}

bool isValid( point& p ) {
    return ( p.x >-1 && p.y > -1 && p.x < m.w && p.y < m.h );
}

bool existPoint( point& p, int cost ) {
    std::list<node>::iterator i;
    i = std::find( closed.begin(), closed.end(), p );
    if( i != closed.end() ) {
        if( ( *i ).cost + ( *i ).dist < cost ) return true;
        else { closed.erase( i ); return false; }
    }
    i = std::find( open.begin(), open.end(), p );
    if( i != open.end() ) {
        if( ( *i ).cost + ( *i ).dist < cost ) return true;
        else { open.erase( i ); return false; }
    }
    return false;
}

bool fillOpen( node& n ) {
    int stepCost, nc, dist;
    point neighbour;

    for( int x = 0; x < 8; x++ ) {
        // one can make diagonals have different cost
        stepCost = x < 4 ? 1 : 1;
        neighbour = n.pos + neighbours[x];
        if( neighbour == end ) return true;

        if( isValid( neighbour ) && m( neighbour.x, neighbour.y ) != 1 ) {
            nc = stepCost + n.cost;
            dist = calcDist( neighbour );
            if( !existPoint( neighbour, nc + dist ) ) {
                node m;
                m.cost = nc; m.dist = dist;
                m.pos = neighbour;
                m.parent = n.pos;
                open.push_back( m );
            }
        }
    }
    return false;
}

bool search( point& s, point& e, map& mp ) {
    node n; end = e; start = s; m = mp;
```

```
        n.cost = 0; n.pos = s; n.parent = 0; n.dist = calcDist( s );
        open.push_back( n );
        while( !open.empty() ) {
            //open.sort();
            node n = open.front();
            open.pop_front();
            closed.push_back( n );
            if( fillOpen( n ) ) return true;
        }
        return false;
    }

    int path( std::list<point>& path ) {
        path.push_front( end );
        int cost = 1 + closed.back().cost;
        path.push_front( closed.back().pos );
        point parent = closed.back().parent;

        for( std::list<node>::reverse_iterator i = closed.rbegin(); i != closed.rend(); i++ ) {
            if( ( *i ).pos == parent && !( ( *i ).pos == start ) ) {
                path.push_front( ( *i ).pos );
                parent = ( *i ).parent;
            }
        }
        path.push_front( start );
        return cost;
    }

    map m; point end, start;
    point neighbours[8];
    std::list<node> open;
    std::list<node> closed;
};

int main( int argc, char* argv[] ) {
    map m;
    point s, e( 7, 7 );
    aStar as;

    if( as.search( s, e, m ) ) {
        std::list<point> path;
        int c = as.path( path );
        for( int y = -1; y < 9; y++ ) {
            for( int x = -1; x < 9; x++ ) {
                if( x < 0 || y < 0 || x > 7 || y > 7 || m( x, y ) == 1 )
                    std::cout << char(0xdb);
                else {
                    if( std::find( path.begin(), path.end(), point( x, y ) )!= path.end() )
                        std::cout << "x";
                    else std::cout << ".";
                }
```

```
            }
            std::cout << "\n";
        }

        std::cout << "\nPath cost " << c << ": ";
        for( std::list<point>::iterator i = path.begin(); i != path.end(); i++ ) {
            std::cout<< "(" << ( *i ).x << ", " << ( *i ).y << ") ";
        }
    }
    std::cout << "\n\n";
    return 0;
}
```

OUTPUT: -


File Edit Selection View Go Run Terminal Help
A-STAR.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
A-STAR.CPP
A-STAR.exe
BEST-FIRST-SEARCH.CPP
BEST-FIRST-SEARCH.exe
BFS-DFS.CPP
BFS-DFS.exe
tempCodeRunnerFile.CPP
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
PS D:\IAI LAB> cd "d:\IAI LAB\" ; if ($?) { g++ A-STAR.CPP -o A-STAR } ; if ($?) { .\A-STAR }
x.....
x.....
x...#.
x.#...
x.#...
.x#...
..xxxx
.....xx
Path cost 11: (0, 0) (0, 1) (0, 2) (0, 3) (0, 4) (1, 5) (2, 6) (3, 6) (4, 6) (5, 6) (6, 7) (7, 7)
PS D:\IAI LAB>

3. Write a program to implement Water Jug Problem.

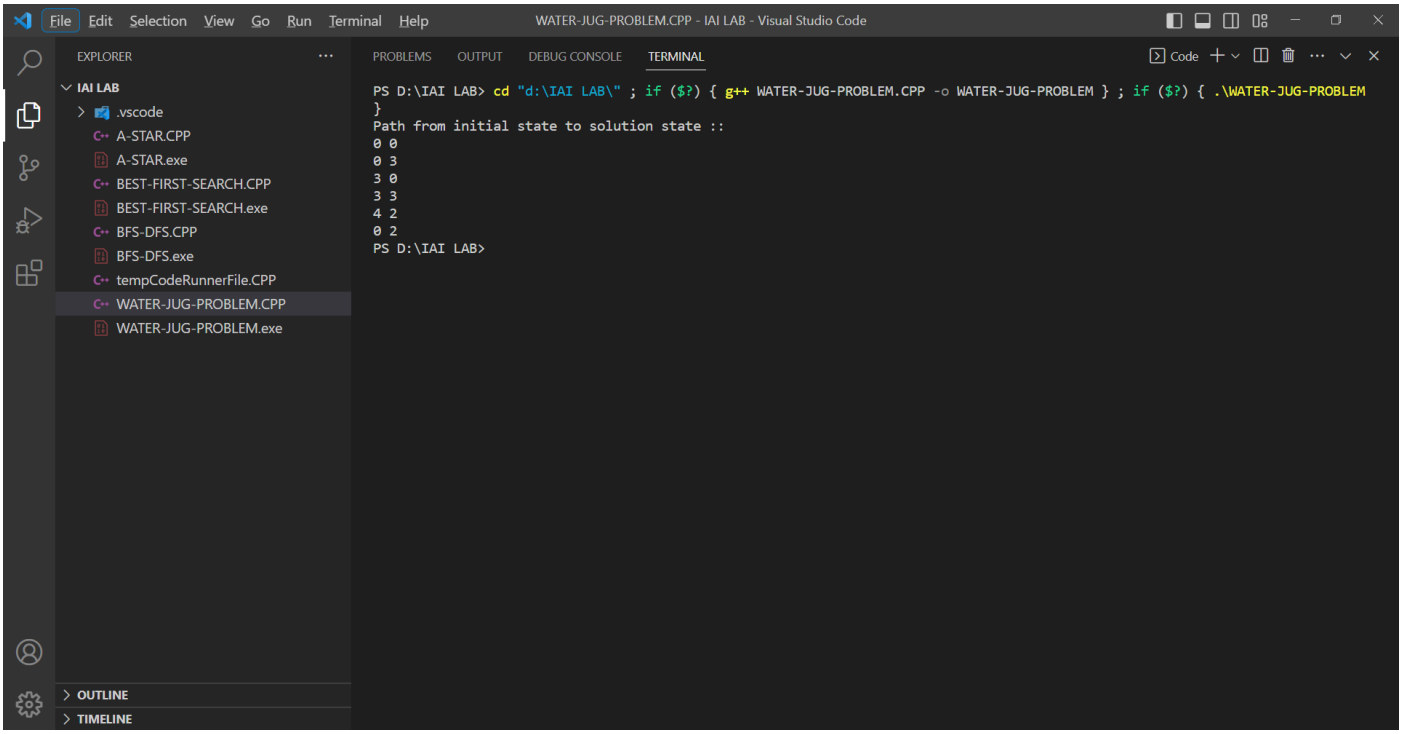
CODE: -

```
#include <bits/stdc++.h>
using namespace std;
typedef pair<int, int> pii;
void printpath(map<pii, pii> mp, pii u)
{
   if (u.first == 0 && u.second == 0) {
      cout << 0 << " " << 0 << endl;
      return;
   }
   printpath(mp, mp[u]);
   cout << u.first << " " << u.second << endl;
}
void BFS(int a, int b, int target)
{
   map<pii, int> m;
   bool isSolvable = false;
   map<pii, pii> mp;

   queue<pii> q;

   q.push(make_pair(0, 0));
   while (!q.empty()) {

      auto u = q.front();

      q.pop();
      if (m[u] == 1)
         continue;

      if ((u.first > a || u.second > b || u.first < 0
         || u.second < 0))
         continue;

      m[{ u.first, u.second }] = 1;

      if (u.first == target || u.second == target) {
         isSolvable = true;

         printpath(mp, u);
         if (u.first == target) {
            if (u.second != 0)
               cout << u.first << " " << 0 << endl;
         }
         else {
            if (u.first != 0)
               cout << 0 << " " << u.second << endl;
         }
```

```
    return;
}

if (m[{ u.first, b }] != 1) {
    q.push({ u.first, b });
    mp[{ u.first, b }] = u;
}

if (m[{ a, u.second }] != 1) {
    q.push({ a, u.second });
    mp[{ a, u.second }] = u;
}

int d = b - u.second;
if (u.first >= d) {
    int c = u.first - d;
    if (m[{ c, b }] != 1) {
        q.push({ c, b });
        mp[{ c, b }] = u;
    }
}
else {
    int c = u.first + u.second;
    if (m[{ 0, c }] != 1) {
        q.push({ 0, c });
        mp[{ 0, c }] = u;
    }
}

d = a - u.first;
if (u.second >= d) {
    int c = u.second - d;
    if (m[{ a, c }] != 1) {
        q.push({ a, c });
        mp[{ a, c }] = u;
    }
}
else {
    int c = u.first + u.second;
    if (m[{ c, 0 }] != 1) {
        q.push({ c, 0 });
        mp[{ c, 0 }] = u;
    }
}

if (m[{ u.first, 0 }] != 1) {
    q.push({ u.first, 0 });
    mp[{ u.first, 0 }] = u;
}

if (m[{ 0, u.second }] != 1) {
```

```
            q.push({ 0, u.second });
            mp[{ 0, u.second }] = u;
        }
    }
    if (!isSolvable)
        cout << "No solution";
}

int main()
{
    int Jug1 = 4, Jug2 = 3, target = 2;
    cout << "Path from initial state "
            "to solution state ::\n";
    BFS(Jug1, Jug2, target);
    return 0;
}
```

OUTPUT: -


File Edit Selection View Go Run Terminal Help
WATER-JUG-PROBLEM.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
.vscode
A-STAR.CPP
A-STAR.exe
BEST-FIRST-SEARCH.CPP
BEST-FIRST-SEARCH.exe
BFS-DFS.CPP
BFS-DFS.exe
tempCodeRunnerFile.CPP
WATER-JUG-PROBLEM.CPP
WATER-JUG-PROBLEM.exe
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
PS D:\IAI LAB> cd "d:\IAI LAB\" ; if ($?) { g++ WATER-JUG-PROBLEM.CPP -o WATER-JUG-PROBLEM } ; if ($?) { .\WATER-JUG-PROBLEM
}
Path from initial state to solution state ::
0 0
0 3
3 0
3 3
4 2
0 2
PS D:\IAI LAB>

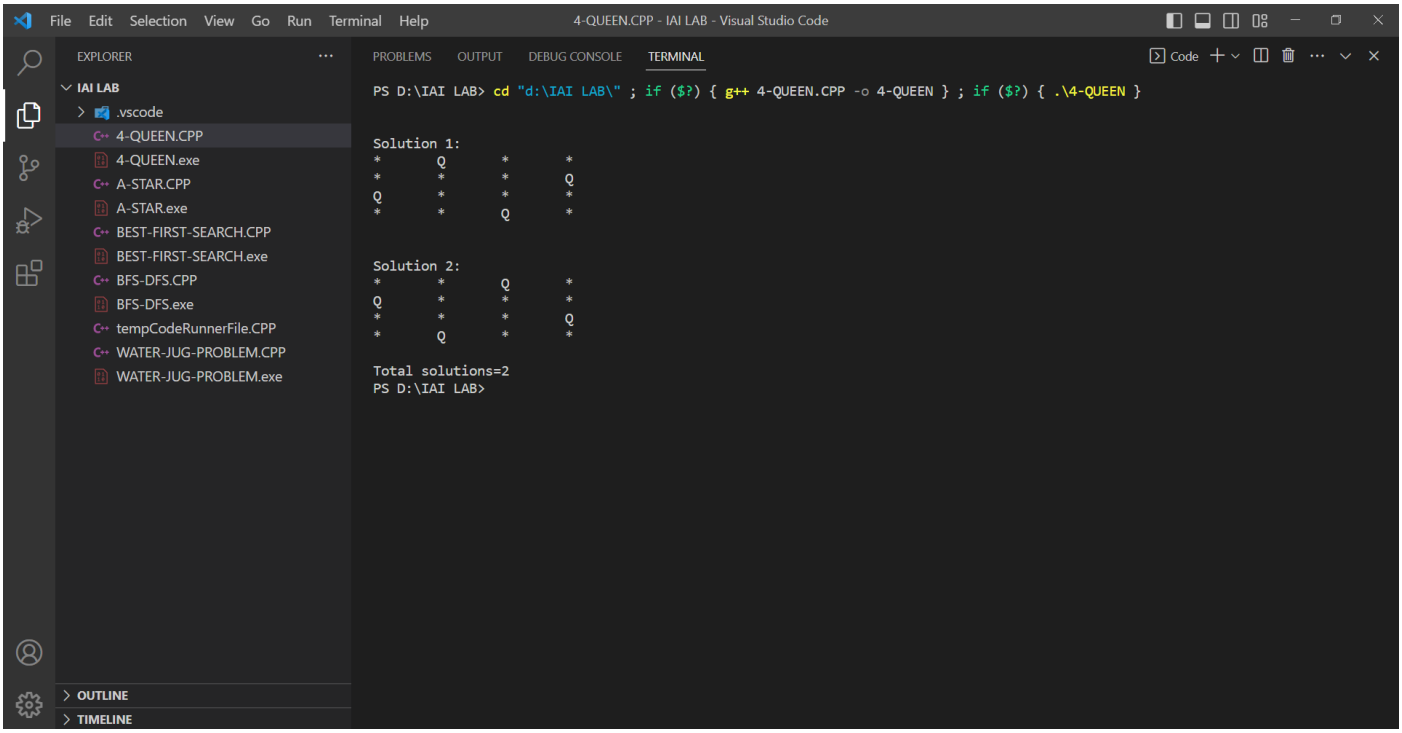
4. Write a program to implement 4-Queen Problem.

CODE: -

```
#include <bits/stdc++.h>
using namespace std;

int a[30], cnt;

int place(int pos)
{
   int i;
   for (i = 1; i < pos; i++) {
      if ((a[i] == a[pos])
         || ((abs(a[i] - a[pos]) == abs(i - pos))))
         return 0;
   }
   return 1;
}

void print_sol(int N)
{
   int i, j;
   cnt++;
   cout << "\n\nSolution " << cnt << ":\n";
   for (i = 1; i <= N; i++) {
      for (j = 1; j <= N; j++) {
         if (a[i] == j)
            cout << "Q\t";
         else
            cout << "*\t";
      }
      cout << endl;
   }
}

void queen(int n)
{
   cnt = 0;
   int k = 1;
   a[k] = 0;
   while (k != 0) {
      a[k] = a[k] + 1;
      while ((a[k] <= n) && !place(k))
         a[k]++;
      if (a[k] <= n) {
         if (k == n)
            print_sol(n);
         else {
            k++;
            a[k] = 0;
         }
      }
      else
```

```
            k--;
        }
    }

int main()
{
    int N = 4;

    queen(N);
    cout << "\nTotal solutions=" << cnt;
    return 0;
}
```

OUTPUT: -


File Edit Selection View Go Run Terminal Help
4-QUEEN.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
.vscode
4-QUEEN.CPP
4-QUEEN.exe
A-STAR.CPP
A-STAR.exe
BEST-FIRST-SEARCH.CPP
BEST-FIRST-SEARCH.exe
BFS-DFS.CPP
BFS-DFS.exe
tempCodeRunnerFile.CPP
WATER-JUG-PROBLEM.CPP
WATER-JUG-PROBLEM.exe
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
PS D:\IAI LAB> cd "d:\IAI LAB\" ; if ($?) { g++ 4-QUEEN.CPP -o 4-QUEEN } ; if ($?) { .\4-QUEEN }
Solution 1:
* Q * *
* * * Q
Q * * *
* * Q *
Solution 2:
* * Q *
Q * * *
* * * Q
* Q * *
Total solutions=2
PS D:\IAI LAB>

5. Write a program to implement AO* algorithm.

CODE: -

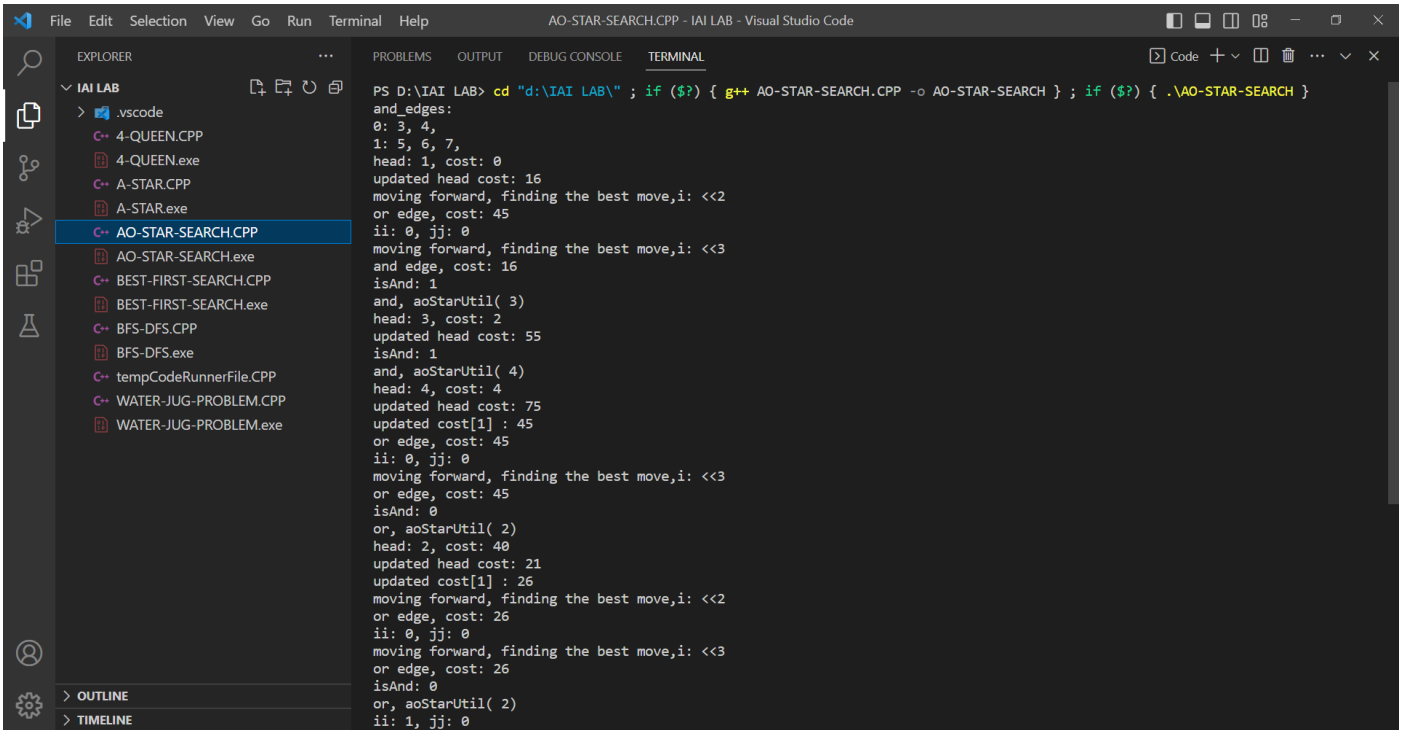
```
#include<bits/stdc++.h>
#define MAX 1000
#define EDGE 5
using namespace std;
bool or_map[MAX],and_map[MAX];

vector<int > or_edges;
vector< vector<int> > and_edges;

vector<int> adj[MAX];

int cost[MAX];
bool isSolved[MAX],visited[MAX];
int min(int a,int b){return a<b?a:b;}
void init(){
   adj[1].push_back(2);adj[1].push_back(3);adj[1].push_back(4);
   adj[2].push_back(5);adj[2].push_back(6);adj[2].push_back(7);
   adj[3].push_back(8);adj[3].push_back(9);
   adj[4].push_back(10);adj[4].push_back(11);
   adj[5].push_back(12);adj[5].push_back(13);
   adj[6].push_back(14);adj[6].push_back(15);
   adj[7].push_back(16);adj[7].push_back(17);
   adj[8].push_back(18);adj[9].push_back(19);adj[10].push_back(20);adj[11].push_back(21);

   cost[1]=0;cost[2]=40;cost[3]=2;cost[4]=4;cost[5]=1;cost[6]=2;cost[7]=3;cost[8]=50;
   cost[9]=60;cost[10]=70;cost[11]=80;cost[12]=4;cost[13]=5;cost[14]=8;cost[15]=9;
   cost[16]=6;cost[17]=7;cost[18]=cost[19]=cost[20]=cost[21]=90;
   for(int i=0;i<=21;i++){
      visited[i]=false;
      isSolved[i]=false;
      and_map[i]=or_map[i]=false;
      if(adj[i].size()==0) isSolved[i]=true;
   }
   vector<int> v;v.push_back(3);v.push_back(4);
   and_map[3]=and_map[4]=true;
   and_edges.push_back(v); v.clear();

   v.push_back(5);v.push_back(6);v.push_back(7);
   and_map[5]=and_map[6]=and_map[7]=true;
   and_edges.push_back(v); v.clear();

   for(int i=1;i<=21;i++)
      if(and_map[i]==false) or_map[i]=true;

   cout<<"and_edges: "<<endl;
   for(int i=0;i<and_edges.size();i++){
      cout<<i<<": ";
      for(int j=0;j<and_edges[i].size();j++)
         cout<<and_edges[i][j]<<", ";
      cout<<endl;
   }
```

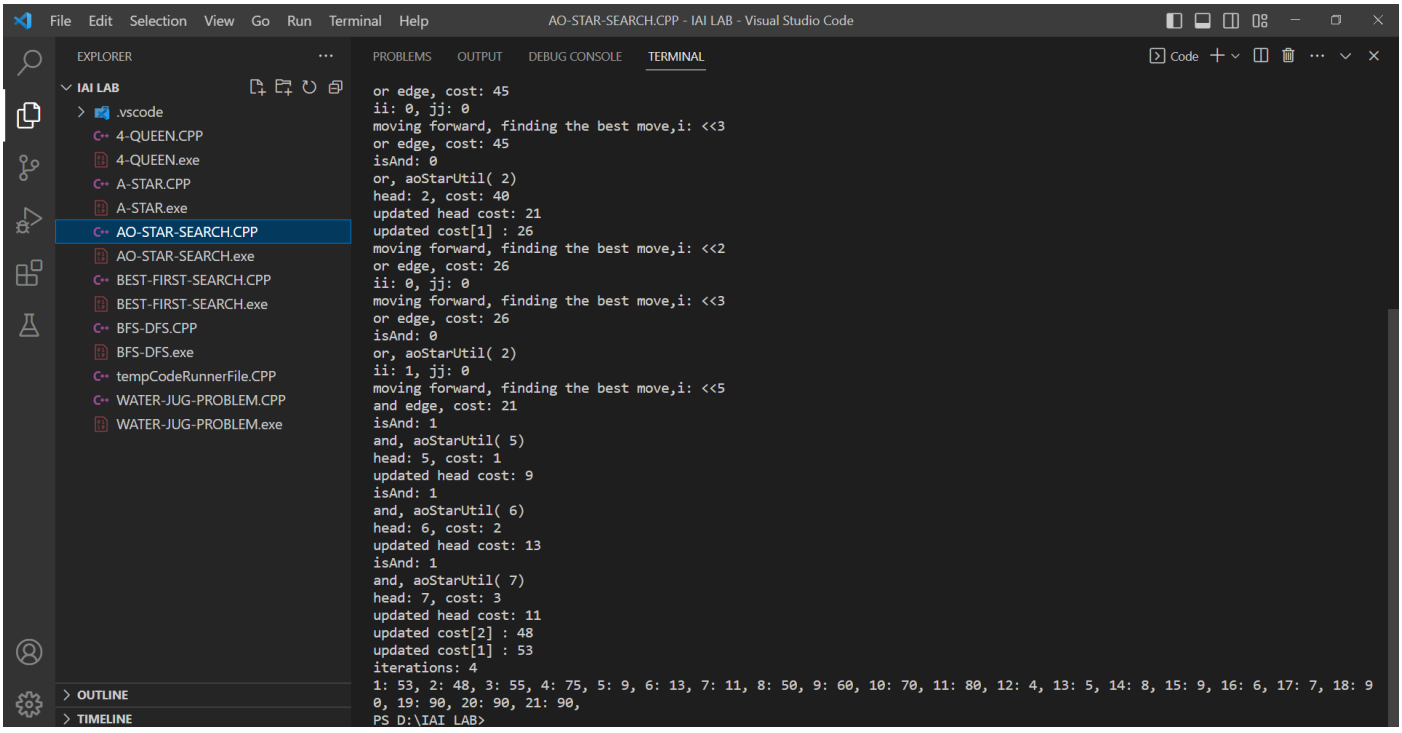
```
}

void aoStarUtil(int head){
   if(visited[head]==false){

      cout<<"head: "<<head<<", cost: "<<cost[head]<<endl;
      visited[head]=true;
      int temp_cost=MAX;bool flag=true;
      int ii=-1,jj=-1;
      map<int,int> temp_map;
      for(int i=0;i<adj[head].size();i++){
         if(temp_map[adj[head][i]]) continue;
         if( and_map[adj[head][i]] ){

            bool temp_solved=true;
            for( ii=0;ii<and_edges.size() ;ii++){
               for( jj=0;jj<and_edges[ii].size() ;jj++){
                  if(and_edges[ii][jj]==adj[head][i]){
                     flag=false;break;
                  }
               }
               if(jj<and_edges[ii].size()){
                  int cc=0;
                  for(int k=0;k<and_edges[ii].size();k++){
                     cc+=cost[and_edges[ii][k]]+EDGE;
                     temp_map[and_edges[ii][k]]=1;
                     temp_solved=temp_solved && isSolved[and_edges[ii][k] ];
                  }
                  temp_cost=min(temp_cost,cc);
                  break;
               }
            }
            if(temp_solved) isSolved[head]=true;
         }else {
            temp_cost=min(temp_cost,cost[adj[head][i] ]+EDGE );
            temp_map[adj[head][i]]=true;
            if(isSolved[adj[head][i] ]) isSolved[head]=true;
         }
      }
      if(temp_cost<=MAX)
         cost[head]=temp_cost;
      cout<<"updated head cost: "<<cost[head]<<endl;
   } else {
      bool isAnd=false;
      int bestCost=MAX,bestMove=-1,bestAndIndex=-1;
      map<int,int> temp_map1;
      for(int i=0;i<adj[head].size();i++){
         if(temp_map1[adj[head][i]]) continue;
         if(or_map[adj[head][i]]){
            if(bestCost>cost[adj[head][i] ]+EDGE){
               bestCost=cost[adj[head][i] ]+EDGE;
               bestMove=i;isAnd=false;
               temp_map1[adj[head][i]]=1;

            }
```

```
    } else {
        int ii=0,jj=0;int c=0;
        for( ii=0;ii<and_edges.size();ii++){
            for( jj=0;jj<and_edges[ii].size();jj++){
                if(and_edges[ii][jj]==adj[head][i]) break;
            }
            if(jj<and_edges[ii].size()){
                for(int k=0;k<and_edges[ii].size();k++){
                    c+=cost[and_edges[ii][k]]+EDGE;
                    temp_map1[and_edges[ii][k]]=1;
                }
                cout<<"ii: "<<ii<<", jj: "<<jj<<endl;
                break;
            }
        }

        if(bestCost>c && c!=0){
            bestCost=c;bestAndIndex=ii;
            bestMove=i;isAnd=true;
        }
    }
    cout<<"moving forward, finding the best move,i: <<"<<adj[head][i]<<"\n";
    if(isAnd){
        cout<<"and edge, cost: "<<bestCost<<endl;
    }else cout<<"or edge, cost: "<<bestCost<<endl;
}
if(isAnd){
    for(int k=0;k<and_edges[bestAndIndex].size();k++){
        cout<<"isAnd: "<<isAnd<<endl;
        cout<<"and, aoStarUtil( "<<and_edges[bestAndIndex][k]<<")"<<endl;
        aoStarUtil(and_edges[bestAndIndex][k]);
    }
}
else{
    cout<<"isAnd: "<<isAnd<<endl;
    cout<<"or, aoStarUtil( "<<adj[head][bestMove]<<")"<<endl;
    aoStarUtil(adj[head][bestMove]);
}
int temp_cost=MAX;map<int,int> temp_map;
for(int i=0;i<adj[head].size();i++){
    if(temp_map[adj[head][i]]) continue;
    if(or_map[adj[head][i] ] ){
        if(isSolved[adj[head][i] ]) isSolved[head]=true;
        temp_cost=min(temp_cost,cost[adj[head][i]]+EDGE);
        temp_map[adj[head][i]]=true;
    }
    else if(and_map[adj[head][i]]){

        int ii=0,jj=0;

        for(ii=0;ii<and_edges.size();ii++){
            for(jj=0;jj<and_edges[ii].size();jj++){
                if(and_edges[ii][jj]==adj[head][i]){
                    break;
                }
```

```
                }
                if(jj<and_edges[ii].size()){
                    int f=true;int cc=0;
                    for(int k=0;k<and_edges[ii].size();k++){
                        f=f&&(isSolved[and_edges[ii][k]]);
                        cc+=cost[and_edges[ii][k]]+EDGE;
                        temp_map[and_edges[ii][k]]=true;
                    }
                    temp_cost=min(temp_cost,cc);
                    if(f) isSolved[head]=true;
                    break;
                }
            }
        }
    }
    if(temp_cost<=MAX)
        cost[head]=temp_cost;
    cout<<"updated cost["<<head<<"] : "<<cost[head]<<endl;
  }
}

void aoStar(int head){
  int iterations=0;
  while(!isSolved[head] && iterations<MAX){
    aoStarUtil(head);
    iterations++;
  }
  cout<<"iterations: "<<iterations<<endl;
  for(int i=1;i<=21;i++){
    cout<<i<<": "<<cost[i]<<", ";
  }cout<<endl;
}

int main(){
  init();
  aoStar(1);

  return 0;
}
```

OUTPUT: -


File Edit Selection View Go Run Terminal Help
AO-STAR-SEARCH.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
.vscode
4-QUEEN.CPP
4-QUEEN.exe
A-STAR.CPP
A-STAR.exe
AO-STAR-SEARCH.CPP
AO-STAR-SEARCH.exe
BEST-FIRST-SEARCH.CPP
BEST-FIRST-SEARCH.exe
BFS-DFS.CPP
BFS-DFS.exe
tempCodeRunnerFile.CPP
WATER-JUG-PROBLEM.CPP
WATER-JUG-PROBLEM.exe
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
PS D:\IAI LAB> cd "d:\IAI LAB\" ; if ($?) { g++ AO-STAR-SEARCH.CPP -o AO-STAR-SEARCH } ; if ($?) { .\AO-STAR-SEARCH }
and_edges:
0: 3, 4,
1: 5, 6, 7,
head: 1, cost: 0
updated head cost: 16
moving forward, finding the best move,i: <<2
or edge, cost: 45
ii: 0, jj: 0
moving forward, finding the best move,i: <<3
and edge, cost: 16
isAnd: 1
and, aoStarUtil(3)
head: 3, cost: 2
updated head cost: 55
isAnd: 1
and, aoStarUtil(4)
head: 4, cost: 4
updated head cost: 75
updated cost[1] : 45
or edge, cost: 45
ii: 0, jj: 0
moving forward, finding the best move,i: <<3
or edge, cost: 45
isAnd: 0
or, aoStarUtil(2)
head: 2, cost: 40
updated head cost: 21
updated cost[1] : 26
moving forward, finding the best move,i: <<2
or edge, cost: 26
ii: 0, jj: 0
moving forward, finding the best move,i: <<3
or edge, cost: 26
isAnd: 0
or, aoStarUtil(2)
ii: 1, jj: 0

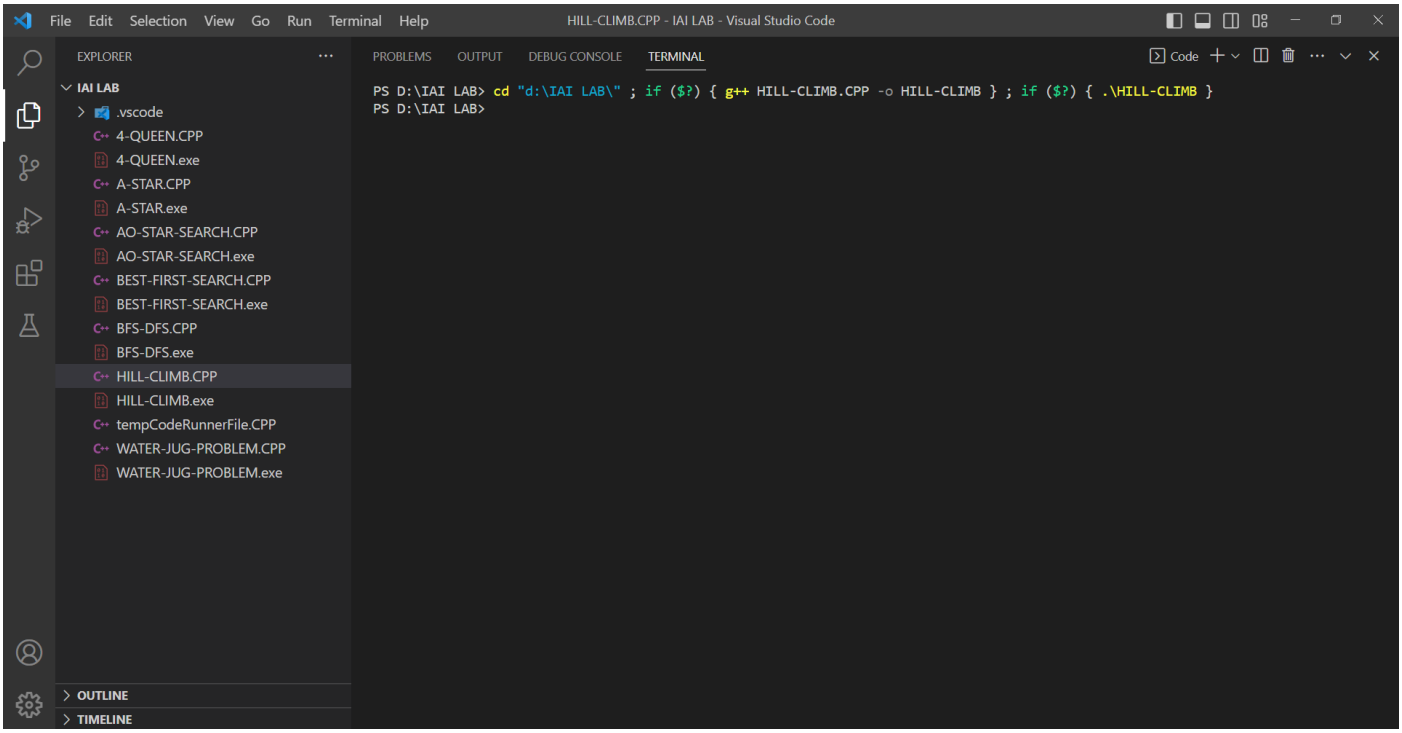

File Edit Selection View Go Run Terminal Help
AO-STAR-SEARCH.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
.vscode
4-QUEEN.CPP
4-QUEEN.exe
A-STAR.CPP
A-STAR.exe
AO-STAR-SEARCH.CPP
AO-STAR-SEARCH.exe
BEST-FIRST-SEARCH.CPP
BEST-FIRST-SEARCH.exe
BFS-DFS.CPP
BFS-DFS.exe
tempCodeRunnerFile.CPP
WATER-JUG-PROBLEM.CPP
WATER-JUG-PROBLEM.exe
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
or edge, cost: 45
ii: 0, jj: 0
moving forward, finding the best move,i: <<3
or edge, cost: 45
isAnd: 0
or, aoStarUtil(2)
head: 2, cost: 40
updated head cost: 21
updated cost[1] : 26
moving forward, finding the best move,i: <<2
or edge, cost: 26
ii: 0, jj: 0
moving forward, finding the best move,i: <<3
or edge, cost: 26
isAnd: 0
or, aoStarUtil(2)
ii: 1, jj: 0
moving forward, finding the best move,i: <<5
and edge, cost: 21
isAnd: 1
and, aoStarUtil(5)
head: 5, cost: 1
updated head cost: 9
isAnd: 1
and, aoStarUtil(6)
head: 6, cost: 2
updated head cost: 13
isAnd: 1
and, aoStarUtil(7)
head: 7, cost: 3
updated head cost: 11
updated cost[2] : 48
updated cost[1] : 53
iterations: 4
1: 53, 2: 48, 3: 55, 4: 75, 5: 9, 6: 13, 7: 11, 8: 50, 9: 60, 10: 70, 11: 80, 12: 4, 13: 5, 14: 8, 15: 9, 16: 6, 17: 7, 18: 9
0, 19: 90, 20: 90, 21: 90,
PS D:\IAI LAB>

6. Write a program to implement hill climbing & steepest ascent hill climbing algorithm.

CODE: -

```
#include <algorithm>
#include <iostream>
#include <vector>

std::vector<int> generate_neighbors(int x)
{
}

int hill_climbing(int (*f)(int), int x0)
{
   int x = x0;
   while (true)
   {
      std::vector<int> neighbors = generate_neighbors(
         x);
      int best_neighbor = *std::max_element(
         neighbors.begin(), neighbors.end(),
         [f](int a, int b)
         {
            return f(a) < f(b);
         });

      if (f(best_neighbor) <= f(x))

         return x;
      x = best_neighbor;
   }
}

int main()
{
   int x0 = 1;
   int x = hill_climbing([](int x)
                         { return x * x; },
                         x0);
   std::cout << "Result: " << x << std::endl;
   return 0;
}
```

OUTPUT: -

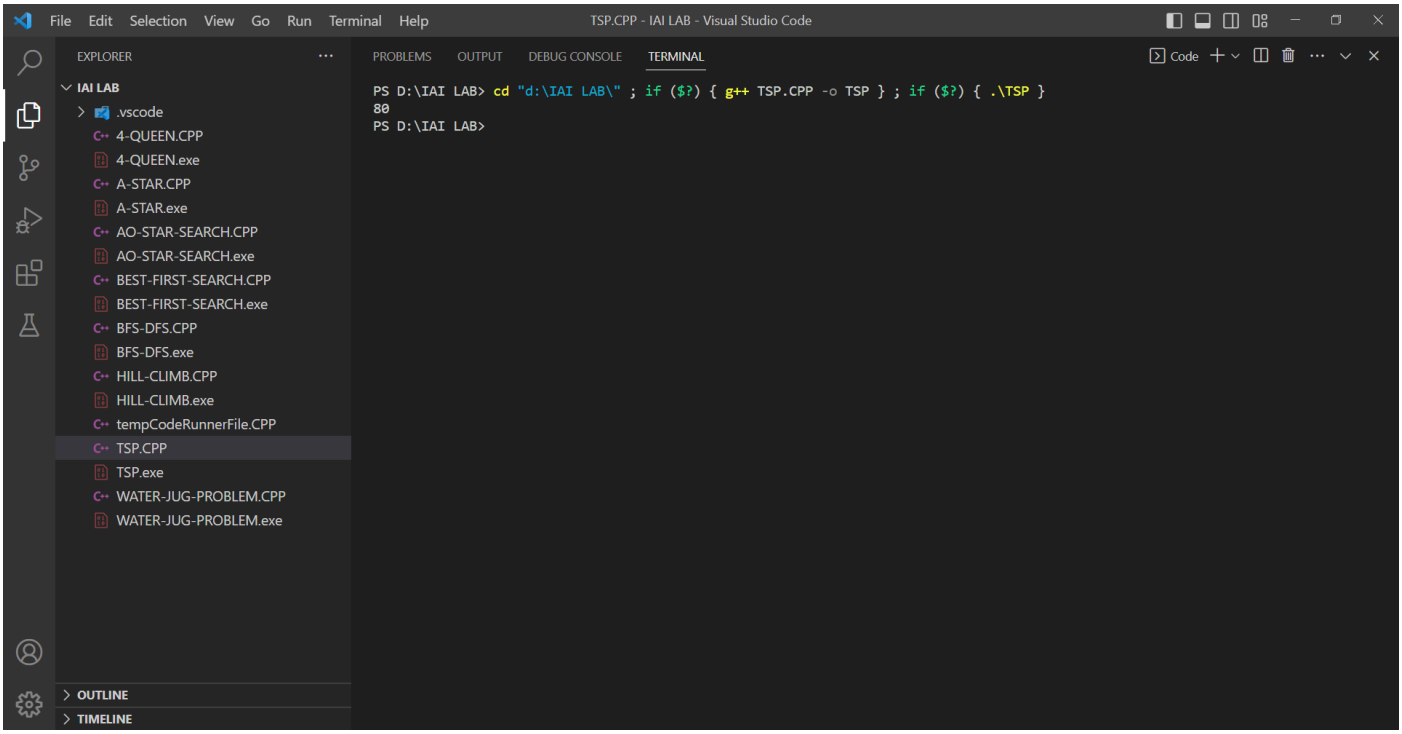

File Edit Selection View Go Run Terminal Help
HILL-CLIMB.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
.vscode
4-QUEEN.CPP
4-QUEEN.exe
A-STAR.CPP
A-STAR.exe
AO-STAR-SEARCH.CPP
AO-STAR-SEARCH.exe
BEST-FIRST-SEARCH.CPP
BEST-FIRST-SEARCH.exe
BFS-DFS.CPP
BFS-DFS.exe
HILL-CLIMB.CPP
HILL-CLIMB.exe
tempCodeRunnerFile.CPP
WATER-JUG-PROBLEM.CPP
WATER-JUG-PROBLEM.exe
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
PS D:\IAI LAB> cd "d:\IAI LAB\" ; if ($?) { g++ HILL-CLIMB.CPP -o HILL-CLIMB } ; if ($?) { .\HILL-CLIMB }
PS D:\IAI LAB>

7. Write a program to implement Travelling Salesman Problem.

CODE: -

```
#include <bits/stdc++.h>
using namespace std;
#define V 4

int travllingSalesmanProblem(int graph[][V], int s)
{
   vector<int> vertex;
   for (int i = 0; i < V; i++)
      if (i != s)
         vertex.push_back(i);

   int min_path = INT_MAX;
   do
   {

      int current_pathweight = 0;

      int k = s;
      for (int i = 0; i < vertex.size(); i++)
      {
         current_pathweight += graph[k][vertex[i]];
         k = vertex[i];
      }
      current_pathweight += graph[k][s];

      min_path = min(min_path, current_pathweight);

   } while (
      next_permutation(vertex.begin(), vertex.end()));

   return min_path;
}

int main()
{
   int graph[][V] = {{0, 10, 15, 20},
                     {10, 0, 35, 25},
                     {15, 35, 0, 30},
                     {20, 25, 30, 0}};
   int s = 0;
   cout << travllingSalesmanProblem(graph, s) << endl;
   return 0;
}
```

OUTPUT: -


File Edit Selection View Go Run Terminal Help
TSP.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
.vscode
4-QUEEN.CPP
4-QUEEN.exe
A-STAR.CPP
A-STAR.exe
AO-STAR-SEARCH.CPP
AO-STAR-SEARCH.exe
BEST-FIRST-SEARCH.CPP
BEST-FIRST-SEARCH.exe
BFS-DFS.CPP
BFS-DFS.exe
HILL-CLIMB.CPP
HILL-CLIMB.exe
tempCodeRunnerFile.CPP
TSP.CPP
TSP.exe
WATER-JUG-PROBLEM.CPP
WATER-JUG-PROBLEM.exe
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
PS D:\IAI LAB> cd "d:\IAI LAB\" ; if ($?) { g++ TSP.CPP -o TSP } ; if ($?) { .\TSP }
80
PS D:\IAI LAB>

8. Write a program to implement Genetic Algorithm for different types of gene representation.

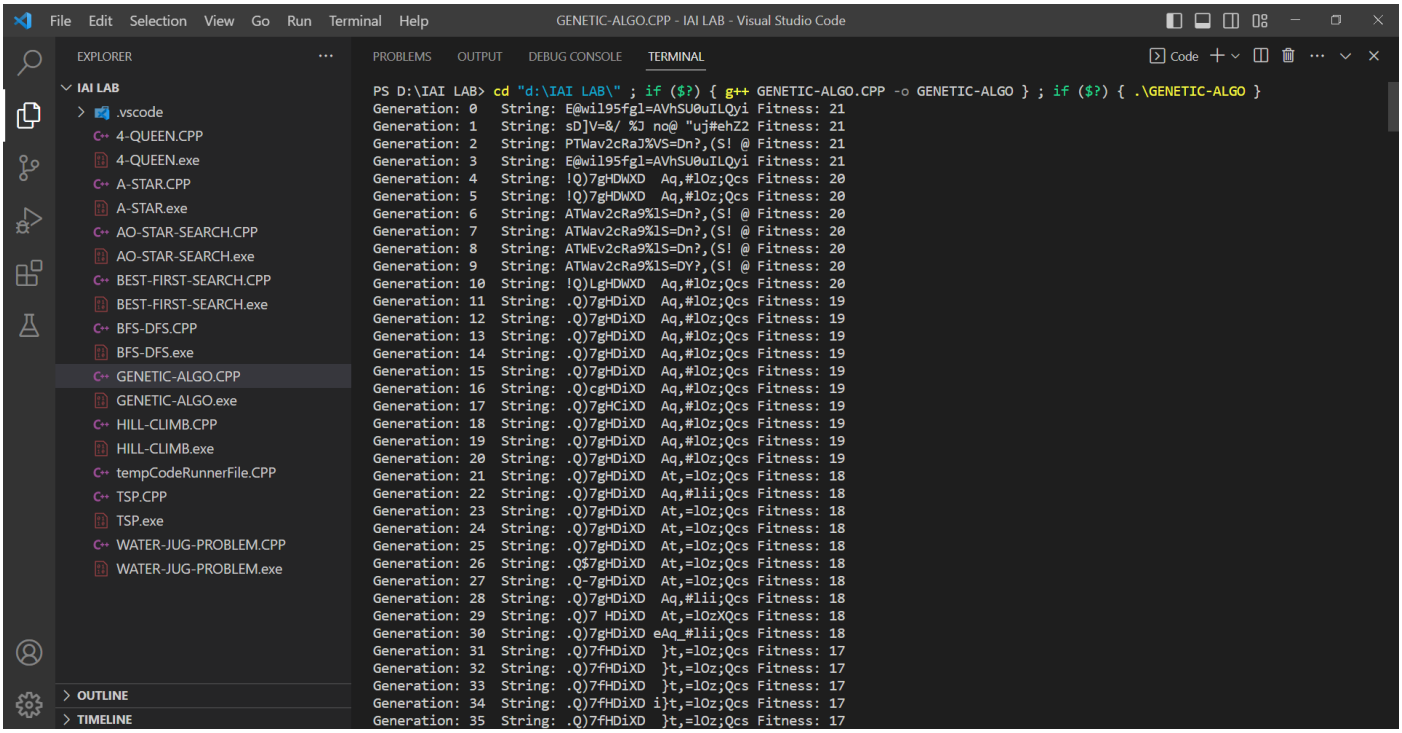
CODE: -

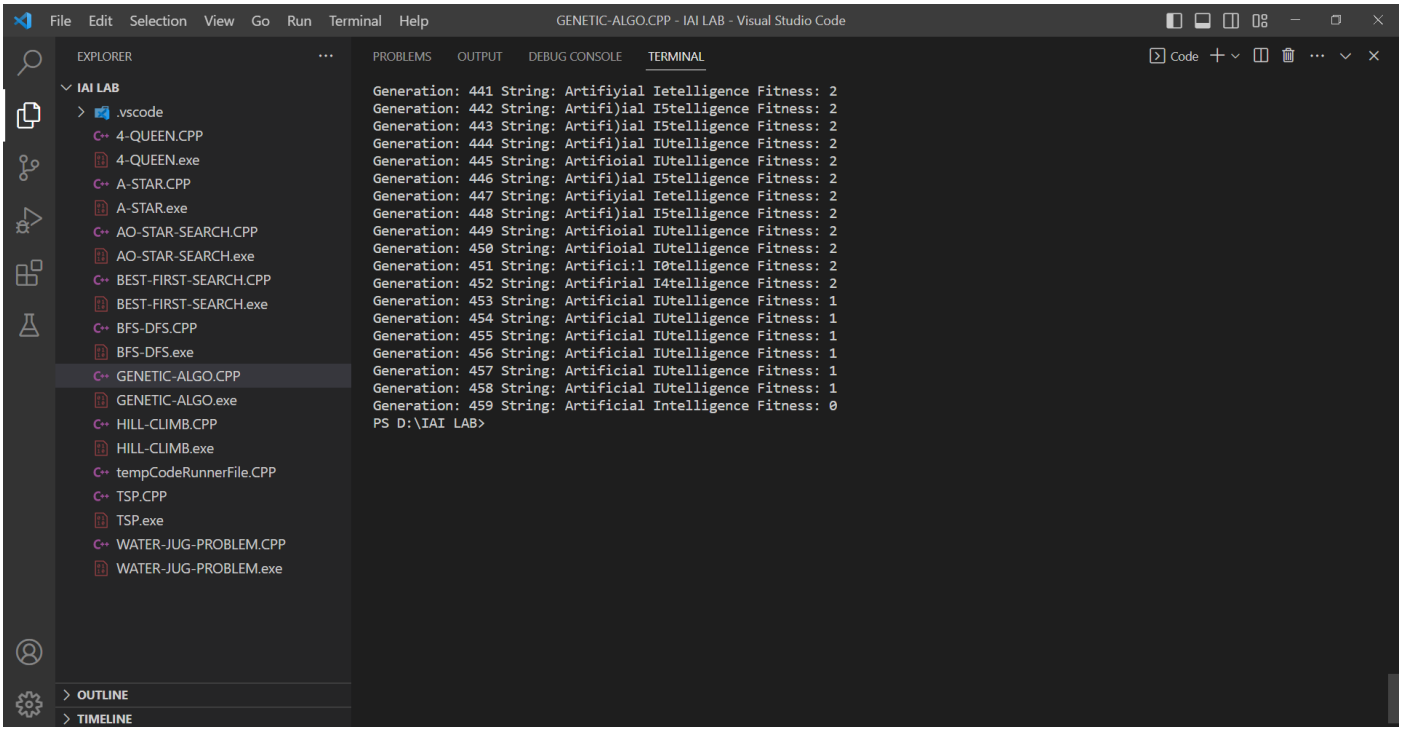
```
#include <bits/stdc++.h>
using namespace std;

#define POPULATION_SIZE 100

const string GENES = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOP"
                     "QRSTUVWXYZ 1234567890, .-;:_!\"#%&/()=?@${[]}";

const string TARGET = "Artificial Intelligence";

int random_num(int start, int end)
{
   int range = (end - start) + 1;
   int random_int = start + (rand() % range);
   return random_int;
}
char mutated_genes()
{
   int len = GENES.size();
   int r = random_num(0, len - 1);
   return GENES[r];
}
string create_gnome()
{
   int len = TARGET.size();
   string gnome = "";
   for (int i = 0; i < len; i++)
      gnome += mutated_genes();
   return gnome;
}
class Individual
{
public:
   string chromosome;
   int fitness;
   Individual(string chromosome);
   Individual mate(Individual parent2);
   int cal_fitness();
};

Individual::Individual(string chromosome)
{
   this->chromosome = chromosome;
   fitness = cal_fitness();
};
```

```
Individual Individual::mate(Individual par2)
{
    string child_chromosome = "";

    int len = chromosome.size();
    for (int i = 0; i < len; i++)
    {
        float p = random_num(0, 100) / 100;

        if (p < 0.45)
            child_chromosome += chromosome[i];

        else if (p < 0.90)
            child_chromosome += par2.chromosome[i];

        else
            child_chromosome += mutated_genes();
    }

    return Individual(child_chromosome);
};

int Individual::cal_fitness()
{
    int len = TARGET.size();
    int fitness = 0;
    for (int i = 0; i < len; i++)
    {
        if (chromosome[i] != TARGET[i])
            fitness++;
    }
    return fitness;
};

bool operator<(const Individual &ind1, const Individual &ind2)
{
    return ind1.fitness < ind2.fitness;
}

int main()
{
    srand((unsigned)(time(0)));

    int generation = 0;

    vector<Individual> population;
    bool found = false;

    for (int i = 0; i < POPULATION_SIZE; i++)
    {
        string gnome = create_gnome();
```

```
        population.push_back(Individual(gnome));
    }

    while (!found)
    {
        sort(population.begin(), population.end());

        if (population[0].fitness <= 0)
        {
            found = true;
            break;
        }
        vector<Individual> new_generation;

        int s = (10 * POPULATION_SIZE) / 100;
        for (int i = 0; i < s; i++)
            new_generation.push_back(population[i]);

        s = (90 * POPULATION_SIZE) / 100;
        for (int i = 0; i < s; i++)
        {
            int len = population.size();
            int r = random_num(0, 50);
            Individual parent1 = population[r];
            r = random_num(0, 50);
            Individual parent2 = population[r];
            Individual offspring = parent1.mate(parent2);
            new_generation.push_back(offspring);
        }
        population = new_generation;
        cout << "Generation: " << generation << "\t";
        cout << "String: " << population[0].chromosome << "\t";
        cout << "Fitness: " << population[0].fitness << "\n";

        generation++;
    }
    cout << "Generation: " << generation << "\t";
    cout << "String: " << population[0].chromosome << "\t";
    cout << "Fitness: " << population[0].fitness << "\n";
}
```

OUTPUT: -


File Edit Selection View Go Run Terminal Help
GENETIC-ALGO.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
.vscode
4-QUEEN.CPP
4-QUEEN.exe
A-STAR.CPP
A-STAR.exe
AO-STAR-SEARCH.CPP
AO-STAR-SEARCH.exe
BEST-FIRST-SEARCH.CPP
BEST-FIRST-SEARCH.exe
BFS-DFS.CPP
BFS-DFS.exe
GENETIC-ALGO.CPP
GENETIC-ALGO.exe
HILL-CLIMB.CPP
HILL-CLIMB.exe
tempCodeRunnerFile.CPP
TSP.CPP
TSP.exe
WATER-JUG-PROBLEM.CPP
WATER-JUG-PROBLEM.exe
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
PS D:\IAI LAB> cd "d:\IAI LAB\" ; if ($?) { g++ GENETIC-ALGO.CPP -o GENETIC-ALGO } ; if ($?) { .\GENETIC-ALGO }
Generation: 0 String: E@wil95fgl=AVhSU0uILQyi Fitness: 21
Generation: 1 String: sD]V=&/ %J no@ "uj#ehZ2 Fitness: 21
Generation: 2 String: PTWav2cRaJ%VS=Dn?,(S! @ Fitness: 21
Generation: 3 String: E@wil95fgl=AVhSU0uILQyi Fitness: 21
Generation: 4 String: !Q)7gHDWXD Aq,#lOz;Qcs Fitness: 20
Generation: 5 String: !Q)7gHDWXD Aq,#lOz;Qcs Fitness: 20
Generation: 6 String: ATWav2cRa9%lS=Dn?,(S! @ Fitness: 20
Generation: 7 String: ATWav2cRa9%lS=Dn?,(S! @ Fitness: 20
Generation: 8 String: ATWEv2cRa9%lS=Dn?,(S! @ Fitness: 20
Generation: 9 String: ATWav2cRa9%lS=DY?,(S! @ Fitness: 20
Generation: 10 String: !Q)LgHDWXD Aq,#lOz;Qcs Fitness: 20
Generation: 11 String: .Q)7gHDiXD Aq,#lOz;Qcs Fitness: 19
Generation: 12 String: .Q)7gHDiXD Aq,#lOz;Qcs Fitness: 19
Generation: 13 String: .Q)7gHDiXD Aq,#lOz;Qcs Fitness: 19
Generation: 14 String: .Q)7gHDiXD Aq,#lOz;Qcs Fitness: 19
Generation: 15 String: .Q)7gHDiXD Aq,#lOz;Qcs Fitness: 19
Generation: 16 String: .Q)cgHDiXD Aq,#lOz;Qcs Fitness: 19
Generation: 17 String: .Q)7gHCiXD Aq,#lOz;Qcs Fitness: 19
Generation: 18 String: .Q)7gHDiXD Aq,#lOz;Qcs Fitness: 19
Generation: 19 String: .Q)7gHDiXD Aq,#lOz;Qcs Fitness: 19
Generation: 20 String: .Q)7gHDiXD Aq,#lOz;Qcs Fitness: 19
Generation: 21 String: .Q)7gHDiXD At,=lOz;Qcs Fitness: 18
Generation: 22 String: .Q)7gHDiXD Aq,#lii;Qcs Fitness: 18
Generation: 23 String: .Q)7gHDiXD At,=lOz;Qcs Fitness: 18
Generation: 24 String: .Q)7gHDiXD At,=lOz;Qcs Fitness: 18
Generation: 25 String: .Q)7gHDiXD At,=lOz;Qcs Fitness: 18
Generation: 26 String: .Q$7gHDiXD At,=lOz;Qcs Fitness: 18
Generation: 27 String: .Q-7gHDiXD At,=lOz;Qcs Fitness: 18
Generation: 28 String: .Q)7gHDiXD Aq,#lii;Qcs Fitness: 18
Generation: 29 String: .Q)7 HDiXD At,=lOzXQcs Fitness: 18
Generation: 30 String: .Q)7gHDiXD eAq_#lii;Qcs Fitness: 18
Generation: 31 String: .Q)7fHDiXD }t,=lOz;Qcs Fitness: 17
Generation: 32 String: .Q)7fHDiXD }t,=lOz;Qcs Fitness: 17
Generation: 33 String: .Q)7fHDiXD }t,=lOz;Qcs Fitness: 17
Generation: 34 String: .Q)7fHDiXD i}t,=lOz;Qcs Fitness: 17
Generation: 35 String: .Q)7fHDiXD }t,=lOz;Qcs Fitness: 17


File Edit Selection View Go Run Terminal Help
GENETIC-ALGO.CPP - IAI LAB - Visual Studio Code
EXPLORER
IAI LAB
.vscode
4-QUEEN.CPP
4-QUEEN.exe
A-STAR.CPP
A-STAR.exe
AO-STAR-SEARCH.CPP
AO-STAR-SEARCH.exe
BEST-FIRST-SEARCH.CPP
BEST-FIRST-SEARCH.exe
BFS-DFS.CPP
BFS-DFS.exe
GENETIC-ALGO.CPP
GENETIC-ALGO.exe
HILL-CLIMB.CPP
HILL-CLIMB.exe
tempCodeRunnerFile.CPP
TSP.CPP
TSP.exe
WATER-JUG-PROBLEM.CPP
WATER-JUG-PROBLEM.exe
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
Generation: 441 String: Artifiyial Ietelligence Fitness: 2
Generation: 442 String: Artifi)ial I5telligence Fitness: 2
Generation: 443 String: Artifi)ial I5telligence Fitness: 2
Generation: 444 String: Artifi)ial IUtelligence Fitness: 2
Generation: 445 String: Artifioial IUtelligence Fitness: 2
Generation: 446 String: Artifi)ial I5telligence Fitness: 2
Generation: 447 String: Artifiyial Ietelligence Fitness: 2
Generation: 448 String: Artifi)ial I5telligence Fitness: 2
Generation: 449 String: Artifioial IUtelligence Fitness: 2
Generation: 450 String: Artifioial IUtelligence Fitness: 2
Generation: 451 String: Artifici:l I0telligence Fitness: 2
Generation: 452 String: Artifirial I4telligence Fitness: 2
Generation: 453 String: Artificial IUtelligence Fitness: 1
Generation: 454 String: Artificial IUtelligence Fitness: 1
Generation: 455 String: Artificial IUtelligence Fitness: 1
Generation: 456 String: Artificial IUtelligence Fitness: 1
Generation: 457 String: Artificial IUtelligence Fitness: 1
Generation: 458 String: Artificial IUtelligence Fitness: 1
Generation: 459 String: Artificial Intelligence Fitness: 0
PS D:\IAI LAB>