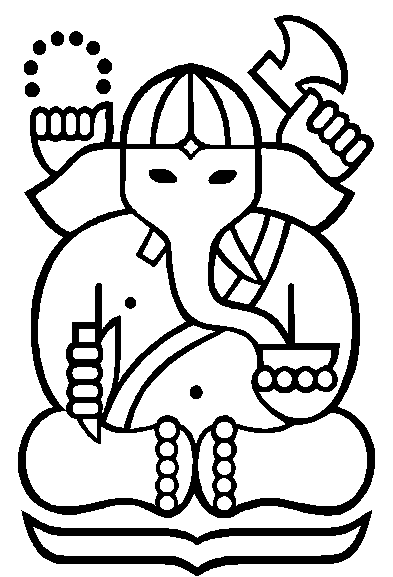
Penyelesaian Persoalan TSP dengan Algoritma

*Branch and Bound*



oleh

**Felix Limanta / 13515065**

**Laporan**

Disusun untuk memenuhi persyaratan nilai mata kuliah

IF2211 – Strategi Algoritma

**Program Studi Teknik Informatika**

**Institut Teknologi Bandung**

**Bandung**

**2017**

# Deskripsi Masalah

This paper. Theoretically, ...

The issue of ...

This paper is organized as follows. The next section presents ... Then, Section 3 discusses

the ... Section 4 analyzes the ... Concluding remarks are offered in Section 5.

# *Source Code*

## SquareMatrix

|  |
| --- |
| **// File : square\_matrix.h**  **// Author : Felix Limanta**  **// Date : March 30, 2017**  #include <cstdlib>  #include <iostream>  #include <limits>  **using** **namespace** std**;**  #ifndef SQUARE\_MATRIX\_H  #define SQUARE\_MATRIX\_H  **const** **double** INF **=** numeric\_limits**<double>::**infinity**();**  **class** SquareMatrix **{**  **private:**  **static** **const** **int** DEF\_SIZE**;**  **double\*\*** data**;**  **int** size**;**  **public:**  **class** SquareMatrixRow **{**  **private:**  **double\*** row**;**  **public:**  SquareMatrixRow**(double\*** row**):** row**(**row**)** **{** **}**  **double** **operator[](**size\_t index**)** **const** **{**  **return** row**[**index**];**  **}**  **};**  SquareMatrix**(int** size **=** DEF\_SIZE**);**  SquareMatrix**(const** SquareMatrix**&** m**);**  **~**SquareMatrix**();**  SquareMatrix**&** **operator=(const** SquareMatrix**&** m**);**  SquareMatrixRow **operator[](**size\_t index**)** **const** **{**  **return** SquareMatrixRow**(**data**[**index**]);**  **}**  **friend** ostream**&** **operator<<(**ostream**&** os**,** **const** SquareMatrix**&** m**);**  **void** setData**(**size\_t i**,** size\_t j**,** **double** data**);**  **double\*\*** getData**()** **const;**  **double** getData**(**size\_t i**,** size\_t j**)** **const;**  **int** getSize**()** **const;**  **};**  #endif |
| **// File : square\_matrix.cpp**  **// Author : Felix Limanta**  **// Date : March 30, 2017**  #include "square\_matrix.h"  #include <iostream>  **using** **namespace** std**;**  **const** **int** SquareMatrix**::**DEF\_SIZE **=** 20**;**  SquareMatrix**::**SquareMatrix**(int** size**):** size**(**size**)** **{**  data **=** **new** **double\*[**size**];**  **for(int** i **=** 0**;** i **<** size**;** **++**i**)** **{**  data**[**i**]** **=** **new** **double[**size**];**  **for** **(int** j **=** 0**;** j **<** size**;** **++**j**)**  data**[**i**][**j**]** **=** 0**;**  **}**  **}**  SquareMatrix**::**SquareMatrix**(const** SquareMatrix**&** m**):** size**(**m**.**size**)** **{**  data **=** **new** **double\*[**size**];**  **for(int** i **=** 0**;** i **<** size**;** **++**i**)** **{**  data**[**i**]** **=** **new** **double[**size**];**  **for** **(int** j **=** 0**;** j **<** size**;** **++**j**)**  data**[**i**][**j**]** **=** m**.**data**[**i**][**j**];**  **}**  **}**  SquareMatrix**::~**SquareMatrix**()** **{**  **for** **(int** i **=** 0**;** i **<** size**;** **++**i**)**  **delete** **[]** data**[**i**];**  **delete** **[]** data**;**  **}**  SquareMatrix**&** SquareMatrix**::operator=(const** SquareMatrix**&** m**)** **{**  **if** **(this** **!=** **&**m**)** **{**  **// dtor**  **for** **(int** i **=** 0**;** i **<** size**;** **++**i**)**  **delete** **[]** data**[**i**];**  **delete** **[]** data**;**    **// cctor**  size **=** m**.**size**;**  data **=** **new** **double\*[**size**];**  **for(int** i **=** 0**;** i **<** size**;** **++**i**)** **{**  data**[**i**]** **=** **new** **double[**size**];**  **for** **(int** j **=** 0**;** j **<** size**;** **++**j**)**  data**[**i**][**j**]** **=** m**.**data**[**i**][**j**];**  **}**  **}**  **return** **\*this;**  **}**  ostream**&** **operator<<(**ostream**&** os**,** **const** SquareMatrix**&** m**)** **{**  **for** **(int** i **=** 0**;** i **<** m**.**size**;** **++**i**)** **{**  **for** **(int** j **=** 0**;** j **<** m**.**size**;** **++**j**)** **{**  os **<<** m**.**data**[**i**][**j**]** **<<** '\t'**;**  **}**  os **<<** '\n'**;**  **}**  **return** os**;**  **}**  **void** SquareMatrix**::**setData**(**size\_t i**,** size\_t j**,** **double** data**)** **{**  **this->**data**[**i**][**j**]** **=** data**;**  **}**  **double\*\*** SquareMatrix**::**getData**()** **const** **{**  **return** data**;**  **}**  **double** SquareMatrix**::**getData**(**size\_t i**,** size\_t j**)** **const** **{**  **return** data**[**i**][**j**];**  **}**  **int** SquareMatrix**::**getSize**()** **const** **{**  **return** size**;**  **}** |

## Node

|  |
| --- |
| **// File : node.h**  **// Author : Felix Limanta**  **// Date : March 30, 2017**  #include "square\_matrix.h"  #include <vector>  #ifndef NODE\_H  #define NODE\_H  **class** Node **{**  **protected:**  **int** id**;**  Node**\*** parent**;**  Node**\*\*** child**;**  vector**<int>** path**;**  **bool\*** visited**;**  SquareMatrix matrix**;**  **double** cost**;**    **public:**  Node**(int** id**,** **const** SquareMatrix**&** m**);**  Node**(int** id**,** **const** SquareMatrix**&** m**,** Node**\*** parent**);**  **virtual** **~**Node**();**    **int** getID**()** **const;**  Node**\*** getParent**()** **const;**  Node**\*\*** getChild**()** **const;**  vector**<int>&** getPath**();**  **bool\*** getVisited**()** **const;**  SquareMatrix**&** getMatrix**();**  **double** getCost**()** **const;**    **virtual** **void** calculateCost**()** **=** 0**;**  **virtual** **void** expandNode**()** **=** 0**;**  **bool** isSolution**();**  **};**  #endif |
| **// File : node.cpp**  **// Author : Felix Limanta**  **// Date : March 30, 2017**  #include "node.h"  Node**::**Node**(int** id**,** **const** SquareMatrix**&** m**):** id**(**id**),** parent**(NULL),** cost**(**0**)** **{**  matrix **=** m**;**  child **=** **new** Node**\*[**m**.**getSize**()];**    path**.**push\_back**(**id**);**  visited **=** **new** **bool[**m**.**getSize**()];**  **for** **(int** i **=** 0**;** i **<** m**.**getSize**();** **++**i**)** **{**  visited**[**i**]** **=** **false;**  **}**  visited**[**id**]** **=** **true;**  **}**  Node**::**Node**(int** id**,** **const** SquareMatrix**&** m**,** Node**\*** parent**):** id**(**id**),** parent**(**parent**),** cost**(**0**)** **{**  matrix **=** m**;**  child **=** **new** Node**\*[**m**.**getSize**()];**    path **=** parent**->**path**;**  path**.**push\_back**(**id**);**  visited **=** **new** **bool[**m**.**getSize**()];**  **for** **(int** i **=** 0**;** i **<** m**.**getSize**();** **++**i**)** **{**  visited**[**i**]** **=** parent**->**visited**[**i**];**  **}**  visited**[**id**]** **=** **true;**  **}**  Node**::~**Node**()** **{**  **for** **(int** i **=** 0**;** i **<** path**.**size**()** **-** 1**;** **++**i**)**  **delete** child**[**i**];**  **delete** **[]** child**;**  **}**  **int** Node**::**getID**()** **const** **{**  **return** id**;**  **}**  Node**\*** Node**::**getParent**()** **const** **{**  **return** parent**;**  **}**  Node**\*\*** Node**::**getChild**()** **const** **{**  **return** child**;**  **}**  vector**<int>&** Node**::**getPath**()** **{**  **return** path**;**  **}**  **bool\*** Node**::**getVisited**()** **const** **{**  **return** visited**;**  **}**  SquareMatrix**&** Node**::**getMatrix**()** **{**  **return** matrix**;**  **}**  **double** Node**::**getCost**()** **const** **{**  **return** cost**;**  **}**  **bool** Node**::**isSolution**()** **{**  **return** **(**path**.**size**()** **==** matrix**.**getSize**());**  **}** |

## ReducedCostMatrixNode

|  |
| --- |
| **// File : reduced\_cost\_matrix.h**  **// Author : Felix Limanta**  **// Date : March 30, 2017**  #include "node.h"  #ifndef REDUCED\_COST\_MATRIX\_NODE\_H  #define REDUCED\_COST\_MATRIX\_NODE\_H  **class** ReducedCostMatrixNode**:** **public** Node **{**  **private:**  **int** subtracted\_value**;**    **public:**  ReducedCostMatrixNode**(int** id**,** **const** SquareMatrix**&** m**);**  ReducedCostMatrixNode**(int** id**,** **const** SquareMatrix**&** m**,** ReducedCostMatrixNode**\*** parent**);**    **void** setRowsAndColumnsToInfinite**();**  **void** reduceMatrix**();**  **void** calculateCost**();**  **void** expandNode**();**  **};**  #endif |
| **// File : reduced\_cost\_matrix.cpp**  **// Author : Felix Limanta**  **// Date : March 30, 2017**  #include "reduced\_cost\_matrix\_node.h"  #include <algorithm>  ReducedCostMatrixNode**::**ReducedCostMatrixNode**(int** id**,** **const** SquareMatrix**&** m**):** Node**(**id**,**m**),** subtracted\_value**(**0**)** **{**  reduceMatrix**();**  calculateCost**();**  **}**  ReducedCostMatrixNode**::**ReducedCostMatrixNode**(int** id**,** **const** SquareMatrix**&** m**,** ReducedCostMatrixNode**\*** parent**):** Node**(**id**,**m**,**parent**),** subtracted\_value**(**0**)** **{**  setRowsAndColumnsToInfinite**();**  reduceMatrix**();**  calculateCost**();**  **}**  **void** ReducedCostMatrixNode**::**setRowsAndColumnsToInfinite**()** **{**  **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)** **{**  matrix**.**setData**(**parent**->**getID**(),** i**,** INF**);**  matrix**.**setData**(**i**,** id**,** INF**);**  **}**  matrix**.**setData**(**id**,** 0**,** INF**);**  **}**  **void** ReducedCostMatrixNode**::**reduceMatrix**()** **{**  **// Subtract row**  **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)** **{**  **// Find minimum**  **double** min **=** INF**;**  **for** **(int** j **=** 0**;** j **<** matrix**.**getSize**();** **++**j**)** **{**  **if** **(**matrix**[**i**][**j**]** **!=** INF **&&** matrix**[**i**][**j**]** **<** min**)** **{**  min **=** matrix**[**i**][**j**];**  **}**  **}**    **// Subtract row**  **if** **(**min **!=** INF**)** **{**  **for** **(int** j **=** 0**;** j **<** matrix**.**getSize**();** **++**j**)** **{**  **if** **(**matrix**[**i**][**j**]** **!=** INF**)** **{**  matrix**.**setData**(**i**,** j**,** matrix**[**i**][**j**]** **-** min**);**  **}**  **}**  subtracted\_value **+=** min**;**  **}**  **}**    **// Subtract column**  **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)** **{**  **// Find minimum**  **double** min **=** INF**;**  **for** **(int** j **=** 0**;** j **<** matrix**.**getSize**();** **++**j**)** **{**  **if** **(**matrix**[**j**][**i**]** **!=** INF **&&** matrix**[**j**][**i**]** **<** min**)** **{**  min **=** matrix**[**j**][**i**];**  **}**  **}**    **// Subtract column**  **if** **(**min **!=** INF**)** **{**  **for** **(int** j **=** 0**;** j **<** matrix**.**getSize**();** **++**j**)** **{**  **if** **(**matrix**[**j**][**i**]** **!=** INF**)** **{**  matrix**.**setData**(**j**,** i**,** matrix**[**j**][**i**]** **-** min**);**  **}**  **}**  subtracted\_value **+=** min**;**  **}**  **}**  **}**  **void** ReducedCostMatrixNode**::**calculateCost**()** **{**  cost **=** subtracted\_value**;**  **if** **(**parent **!=** **NULL)** **{**  cost **+=** parent**->**getCost**()** **+** parent**->**getMatrix**()[**parent**->**getID**()][**id**];**  **}**  **}**  **void** ReducedCostMatrixNode**::**expandNode**()** **{**  **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)** **{**  **if** **(**visited**[**i**])** **{**  child**[**i**]** **=** **NULL;**  **}** **else** **{**  child**[**i**]** **=** **new** ReducedCostMatrixNode**(**i**,**matrix**,this);**  **}**  **}**  **}** |

## CompleteTourWeightNode

|  |
| --- |
| **// File : complete\_tour\_weight\_node.h**  **// Author : Felix Limanta**  **// Date : April 1, 2017**  #include "node.h"  #include <utility>  #ifndef COMPLETE\_TOUR\_WEIGHT\_NODE\_H  #define COMPLETE\_TOUR\_WEIGHT\_NODE\_H  **class** CompleteTourWeightNode**:** **public** Node **{**  **private:**  pair**<int,int>\*** edges**;**  **bool\*\*** path\_m**;**    **public:**  CompleteTourWeightNode**(int** id**,** **const** SquareMatrix**&** m**);**  CompleteTourWeightNode**(int** id**,** **const** SquareMatrix**&** m**,** CompleteTourWeightNode**\*** parent**);**  **~**CompleteTourWeightNode**();**    **void** findMinimumEdges**();**  **void** calculateCost**();**  **void** expandNode**();**  **};**  #endif |
| **// File : complete\_tour\_weight\_node.cpp**  **// Author : Felix Limanta**  **// Date : April 1, 2017**  #include "complete\_tour\_weight\_node.h"  #include <iostream>  CompleteTourWeightNode**::**CompleteTourWeightNode**(int** id**,** **const** SquareMatrix**&** m**):** Node**(**id**,**m**)** **{**  edges **=** **new** pair**<int,int>[**m**.**getSize**()];**  findMinimumEdges**();**  calculateCost**();**  **}**  CompleteTourWeightNode**::**CompleteTourWeightNode**(int** id**,** **const** SquareMatrix**&** m**,** CompleteTourWeightNode**\*** parent**):** Node**(**id**,**m**,**parent**)** **{**  edges **=** **new** pair**<int,int>[**m**.**getSize**()];**  findMinimumEdges**();**  calculateCost**();**  **}**  CompleteTourWeightNode**::~**CompleteTourWeightNode**()** **{**  **delete** **[]** edges**;**  **}**  **void** CompleteTourWeightNode**::**findMinimumEdges**()** **{**  pair**<int,double>** min1**,** min2**;**    **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)** **{**  min1 **=** make\_pair**(**0**,**INF**);**  min2 **=** min1**;**  **for** **(int** j **=** 0**;** j **<** matrix**.**getSize**();** **++**j**)** **{**  **if** **(**matrix**[**i**][**j**]** **<=** min1**.**second**)** **{**  min2 **=** min1**;**  min1 **=** make\_pair**(**j**,**matrix**[**i**][**j**]);**  **}** **else** **if** **(**matrix**[**i**][**j**]** **<=** min2**.**second**)** **{**  min2 **=** make\_pair**(**j**,**matrix**[**i**][**j**]);**  **}**  **}**  edges**[**i**]** **=** make\_pair**(**min1**.**first**,** min2**.**first**);**  **}**  **for** **(int** i **=** 0**;** i **<** path**.**size**()** **-** 1**;** **++**i**)** **{**  **if** **(**edges**[**path**[**i**]].**first **!=** path**[**i**+**1**])**  edges**[**path**[**i**]].**second **=** path**[**i**+**1**];**  **if** **(**edges**[**path**[**i**+**1**]].**first **!=** path**[**i**])**  edges**[**path**[**i**+**1**]].**second **=** path**[**i**];**  **}**  **}**  **void** CompleteTourWeightNode**::**calculateCost**()** **{**  cost **=** 0**;**  **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)** **{**  cost **+=** matrix**[**i**][**edges**[**i**].**first**];**  cost **+=** matrix**[**i**][**edges**[**i**].**second**];**  **}**  cost **/=** 2**;**  **}**  **void** CompleteTourWeightNode**::**expandNode**()** **{**  **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)** **{**  **if** **(**visited**[**i**])** **{**  child**[**i**]** **=** **NULL;**  **}** **else** **{**  child**[**i**]** **=** **new** CompleteTourWeightNode**(**i**,**matrix**,this);**  **}**  **}**  **}** |

## TSP

|  |
| --- |
| **// File : tsp.cpp**  **// Author : Felix Limanta**  **// Date : March 31, 2017**  #include "square\_matrix.h"  #include "node.h"  #include "complete\_tour\_weight\_node.h"  #include "reduced\_cost\_matrix\_node.h"  #include <algorithm>  **using** **namespace** std**;**  #ifndef TSP\_H  #define TSP\_H  #define REDUCED\_COST\_MATRIX 0  #define COMPLETE\_TOUR\_WEIGHT 1  **struct** Leaf **{**  **double** cost**;**  Node**\*** node\_addr**;**    Leaf**(double** cost**,** Node**\*** node\_addr**):** cost**(**cost**),** node\_addr**(**node\_addr**)** **{};**  **};**  **struct** LeafComparator **{**  **bool** **operator()(const** Leaf**&** lhs**,** **const** Leaf**&** rhs**)** **{**  **return** lhs**.**cost **>** rhs**.**cost**;**  **}**  **};**  **class** TSP **{**  **private:**  SquareMatrix matrix**;**  Node**\*** root**;**  vector**<**Leaf**>** leaves**;**  vector**<int>** final\_path**;**  **double** final\_cost**;**    **public:**  TSP**(int** method**,** **const** SquareMatrix**&** m**);**  **~**TSP**();**    vector**<int>&** getPath**();**  **double** getCost**();**    **void** findPath**();**  **void** expandNode**(**Node**&** n**);**  **};**  #endif |
| **// File : tsp.cpp**  **// Author : Felix Limanta**  **// Date : March 31, 2017**  #include "tsp.h"  TSP**::**TSP **(int** method**,** **const** SquareMatrix**&** m**)** **{**  matrix **=** m**;**  **if** **(**method **==** REDUCED\_COST\_MATRIX**)** **{**  root **=** **new** ReducedCostMatrixNode**(**0**,** m**);**  **}** **else** **if** **(**method **==** COMPLETE\_TOUR\_WEIGHT**)** **{**  root **=** **new** CompleteTourWeightNode**(**0**,** m**);**  **}**  leaves**.**push\_back**(**Leaf**(**root**->**getCost**(),**root**));**  make\_heap**(**leaves**.**begin**(),**leaves**.**end**(),**LeafComparator**());**  **}**  TSP**::~**TSP**()** **{**  **delete** root**;**  **}**  vector**<int>&** TSP**::**getPath**()** **{**  **return** final\_path**;**  **}**  **double** TSP**::**getCost**()** **{**  **return** final\_cost**;**  **}**  **void** TSP**::**findPath**()** **{**  Node**\*** curr\_node**;**  **do** **{**  curr\_node **=** leaves**.**front**().**node\_addr**;**  pop\_heap**(**leaves**.**begin**(),**leaves**.**end**(),**LeafComparator**());**  leaves**.**pop\_back**();**      **while** **(!**curr\_node**->**isSolution**())** **{**  expandNode**(\***curr\_node**);**    curr\_node **=** leaves**.**front**().**node\_addr**;**  pop\_heap**(**leaves**.**begin**(),**leaves**.**end**(),**LeafComparator**());**  leaves**.**pop\_back**();**  **}**  **}** **while** **(**matrix**[**curr\_node**->**getID**()][**0**]** **==** INF**);**    final\_path **=** curr\_node**->**getPath**();**  final\_cost **=** curr\_node**->**getCost**();**  **}**  **void** TSP**::**expandNode**(**Node**&** n**)** **{**  n**.**expandNode**();**    Node**\*\*** child **=** n**.**getChild**();**  **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)** **{**  **if** **(**child**[**i**]** **!=** **NULL)**  leaves**.**push\_back**(**Leaf**(**child**[**i**]->**getCost**(),** child**[**i**]));**  push\_heap**(**leaves**.**begin**(),**leaves**.**end**(),**LeafComparator**());**  **}**  **}** |

## MatrixReader

|  |
| --- |
| **// File : matrix\_reader.h**  **// Author : Felix Limanta**  **// Date : March 30, 2017**  #include "square\_matrix.h"  #include <string>  **using** **namespace** std**;**  #ifndef MATRIX\_READER\_H  #define MATRIX\_READER\_H  **class** MatrixReader **{**  **private:**  SquareMatrix**\*** matrix**;**    **public:**  MatrixReader**(const** string**&** fname**);**  **~**MatrixReader**();**    SquareMatrix getMatrix**()** **const;**  **};**  #endif |
| **// File : matrix\_reader.cpp**  **// Author : Felix Limanta**  **// Date : March 30, 2017**  #include "matrix\_reader.h"  #include <fstream>  #include <sstream>  MatrixReader**::**MatrixReader**(const** string**&** fname**)** **{**  ifstream inf**;**  inf**.**open**(**fname**,** ios**::**in**);**  string instr**;**    **// Read number of nodes**  **int** size**;**  getline**(**inf**,** instr**);**  istringstream**(**instr**)** **>>** size**;**    **// Read matrix**  matrix **=** **new** SquareMatrix**(**size**);**  **for** **(int** i **=** 0**;** i **<** size**;** **++**i**)** **{**  getline**(**inf**,** instr**);**  istringstream row\_in**(**instr**);**  **for** **(int** j **=** 0**;** j **<** size**;** **++**j**)** **{**  **double** n**;**  row\_in **>>** n**;**  **if** **(**i **==** j **||** n **<=** 0**)**  n **=** INF**;**  matrix**->**setData**(**i**,** j**,** n**);**  **}**  **}**    inf**.**close**();**  **}**  MatrixReader**::~**MatrixReader**()** **{**  **delete** matrix**;**  **}**  SquareMatrix MatrixReader**::**getMatrix**()** **const** **{**  **return** **\***matrix**;**  **}** |

## GraphVizGenerator

|  |
| --- |
| **// File : graphviz\_gnerator.h**  **// Author : Felix Limanta**  **// Date : March 31, 2017**  #include "square\_matrix.h"  #include <string>  #include <vector>  #ifndef GRAPHVIZ\_GENERATOR\_H  #define GRAPHVIZ\_GENERATOR\_H  **class** GraphVizGenerator **{**  **private:**  **static** **const** string DEF\_FNAME**;**  SquareMatrix matrix**;**  **bool\*\*** path\_m**;**  **bool** directed**;**  string fname**;**  **double** min\_len**;**  **double** max\_len**;**  **double** avg\_len**;**  **int** n\_edge**;**  **public:**  GraphVizGenerator**(const** SquareMatrix**&** m**,** **const** vector**<int>&** path**,**  string fname **=** DEF\_FNAME**,** **bool** directed **=** **true);**  **~**GraphVizGenerator**();**  **void** generateDirectedGraph**();**  **void** generateUndirectedGraph**();**  **void** generatePNG**();**  **void** generateSVG**();**  **};**  #endif |
| **// File : graphviz\_gnerator.cpp**  **// Author : Felix Limanta**  **// Date : March 31, 2017**  #include "graphviz\_generator.h"  #include <cmath>  #include <fstream>  #include <iostream>  #include <sstream>  string remove\_extension**(const** std**::**string**&** filename**)** **{**  size\_t lastdot **=** filename**.**find\_last\_of**(**"."**);**  **if** **(**lastdot **==** std**::**string**::**npos**)**  **return** filename**;**  **return** filename**.**substr**(**0**,** lastdot**);**  **}**  **const** string GraphVizGenerator**::**DEF\_FNAME **=** "graph"**;**  GraphVizGenerator**::**GraphVizGenerator**(const** SquareMatrix**&** m**,** **const** vector**<int>&** path**,**  string fname**,** **bool** directed**):** directed**(**directed**)** **{**  **this->**fname **=** remove\_extension**(**fname**);**  matrix **=** m**;**  path\_m **=** **new** **bool\*[**m**.**getSize**()];**  **for** **(int** i **=** 0**;** i **<** m**.**getSize**();** **++**i**)** **{**  path\_m**[**i**]** **=** **new** **bool[**m**.**getSize**()];**  **for** **(int** j **=** 0**;** j **<** m**.**getSize**();** **++**j**)** **{**  path\_m**[**i**][**j**]** **=** **false;**  **}**  **}**  **for** **(int** i **=** 0**;** i **<** path**.**size**()** **-** 1**;** **++**i**)** **{**  path\_m**[**path**[**i**]][**path**[**i **+** 1**]]** **=** **true;**  **}**  path\_m**[**path**.**back**()][**0**]** **=** **true;**  avg\_len **=** 0**;**  n\_edge **=** 0**;**  **for** **(int** i **=** 0**;** i **<** m**.**getSize**();** **++**i**)** **{**  **for** **(int** j **=** 0**;** j **<** m**.**getSize**();** **++**j**)** **{**  **if** **(**matrix**[**i**][**j**]** **!=** INF**)** **{**  avg\_len **+=** matrix**[**i**][**j**];**  n\_edge**++;**  **}**  **}**  **}**  avg\_len **/=** n\_edge**;**  **}**  GraphVizGenerator**::~**GraphVizGenerator**()** **{**  **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)**  **delete** **[]** path\_m**[**i**];**  **delete** **[]** path\_m**;**  **}**  **void** GraphVizGenerator**::**generateDirectedGraph**()** **{**  ofstream outf**;**  outf**.**open**(**fname **+** ".gv"**,** ios**::**out**);**  outf **<<** "digraph {\n"**;**  outf **<<** "\tgraph [layout=circo, overlap=scale, splines=true, mindist=2.5];\n"**;**  outf **<<** "\tnode [shape=circle];\n"**;**  outf **<<** "\t0 [peripheries=2];"**;**  **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)** **{**  **for** **(int** j **=** 0**;** j **<** matrix**.**getSize**();** **++**j**)** **{**  **if** **(**matrix**[**i**][**j**]** **!=** INF**)** **{**  outf **<<** '\t' **<<** i **<<** " -> " **<<** j **<<** " [len="  **<<** matrix**[**i**][**j**]/**avg\_len **\*** pow**(**matrix**.**getSize**(),**1.5**)**  **<<** ", label=\"" **<<** matrix**[**i**][**j**];**  **if** **(**path\_m**[**i**][**j**])** **{**  outf **<<** "\", color=red, penwidth=3.0, fontcolor=red];\n"**;**  **}** **else** **{**  outf **<<** "\"];\n"**;**  **}**  **}**  **}**  **}**  outf **<<** "}"**;**  outf**.**close**();**  cout **<<** "GraphViz file generated to " **<<** fname **<<** ".gv\n"**;**  **}**  **void** GraphVizGenerator**::**generateUndirectedGraph**()** **{**  ofstream outf**;**  outf**.**open**(**fname **+** ".gv"**,** ios**::**out**);**  outf **<<** "graph {\n"**;**  outf **<<** "\tgraph [layout=circo, overlap=scale, splines=true, mindist=2.5];\n"**;**  outf **<<** "\tnode [shape=circle];\n"**;**  outf **<<** "\t0 [peripheries=2];"**;**  **for** **(int** i **=** 0**;** i **<** matrix**.**getSize**();** **++**i**)** **{**  **for** **(int** j **=** 0**;** j **<** i**;** **++**j**)** **{**  **if** **(**matrix**[**i**][**j**]** **!=** INF**)** **{**  outf **<<** '\t' **<<** i **<<** " -- " **<<** j **<<** " [len="  **<<** matrix**[**i**][**j**]/**avg\_len **\*** pow**(**matrix**.**getSize**(),**1.5**)**  **<<** ", label=\"" **<<** matrix**[**i**][**j**];**  **if** **(**path\_m**[**i**][**j**]** **||** path\_m**[**j**][**i**])** **{**  outf **<<** "\", color=red, penwidth=3.0, fontcolor=red];\n"**;**  **}** **else** **{**  outf **<<** "\"];\n"**;**  **}**  **}**  **}**  **}**  outf **<<** "}"**;**  outf**.**close**();**  cout **<<** "GraphViz file generated to " **<<** fname **<<** ".gv\n"**;**  **}**  **void** GraphVizGenerator**::**generatePNG**()** **{**  ostringstream command**;**  command **<<** "neato -Tpng " **<<** fname **<<** ".gv -o " **<<** fname **<<** ".png"**;**  system**(**command**.**str**().**c\_str**());**  cout **<<** "Graph image generated to " **<<** fname **<<** ".png\n"**;**  **}**  **void** GraphVizGenerator**::**generateSVG**()** **{**  ostringstream command**;**  command **<<** "neato -Tsvg " **<<** fname **<<** ".gv -o " **<<** fname **<<** ".svg"**;**  system**(**command**.**str**().**c\_str**());**  cout **<<** "Graph image generated to " **<<** fname **<<** ".svg\n"**;**  **}** |

## Main

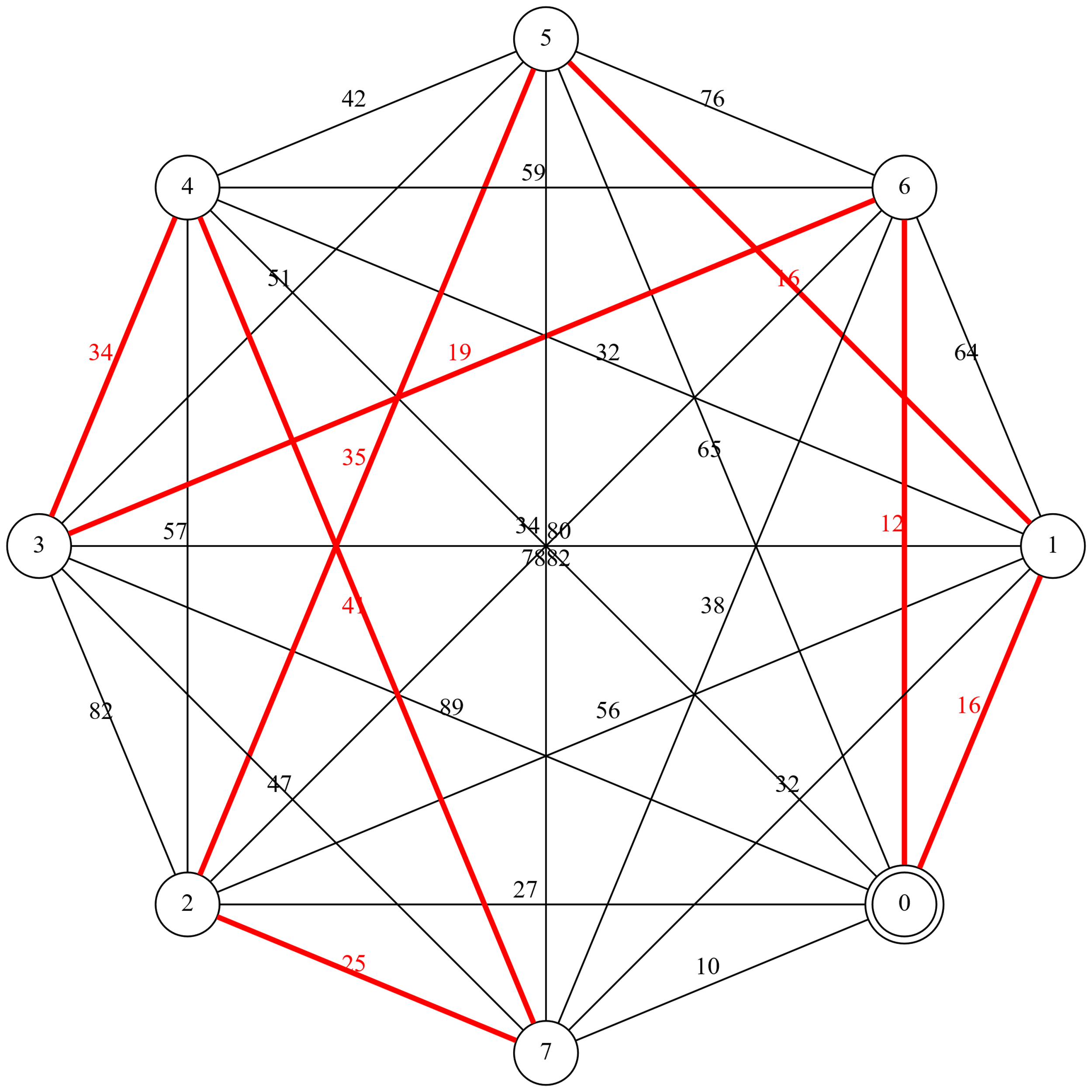
|  |
| --- |
| **// File : main.cpp**  **// Author : Felix Limanta**  **// Date : April 2, 2017**  #include "graphviz\_generator.h"  #include "matrix\_reader.h"  #include "square\_matrix.h"  #include "tsp.h"  #include <chrono>  #include <iostream>  **using** **namespace** std**;**  **int** main **(int** argc**,** **char\*\*** argv**)** **{**  **// Input**  string fname**;**  **int** method **=** REDUCED\_COST\_MATRIX**;**  **if** **(**argc **<=** 1**)** **{**  cout **<<** "Filename:\t"**;**  cin **>>** fname**;**  **}** **else** **{**  fname **=** argv**[**1**];**  **}**  **if** **(**argc **<=** 2**)** **{**  cout **<<** "0. Reduced cost matrix\n"  **<<** "1. Complete tour weight\n"  **<<** "Method:\t"**;**  cin **>>** method**;**  **}** **else** **{**  method **=** stoi**(**argv**[**2**]);**  **}**    SquareMatrix matrix **=** MatrixReader**(**fname**).**getMatrix**();**  cout **<<** "\nMatrix: \n" **<<** matrix**;**    **// Start execution time measurement**  **auto** start **=** chrono**::**system\_clock**::**now**();**    **// Process**  TSP**\*** tsp**;**  **if** **(**method **==** COMPLETE\_TOUR\_WEIGHT**)** **{**  tsp **=** **new** TSP**(**COMPLETE\_TOUR\_WEIGHT**,** matrix**);**  **}** **else** **{**  tsp **=** **new** TSP**(**REDUCED\_COST\_MATRIX**,** matrix**);**  **}**  tsp**->**findPath**();**  vector**<int>** path **=** tsp**->**getPath**();**  **double** cost **=** tsp**->**getCost**();**    **// Measure execution time**  **auto** end **=** chrono**::**system\_clock**::**now**();**  **auto** elapsed **=** chrono**::**duration\_cast**<**chrono**::**microseconds**>(**end **-** start**);**    **// Output results to terminal**  cout **<<** "\nPath:" **<<** '\t'**;**  **for** **(int** i **=** 0**;** i **<** path**.**size**();** **++**i**)**  cout **<<** path**[**i**]** **<<** " -> "**;**  cout **<<** "0\n"**;**  cout **<<** "Cost: " **<<** cost **<<** '\n'**;**  cout **<<** "Elapsed time: " **<<** elapsed**.**count**()** **<<** " us\n"**;**    **// Generate graph**  GraphVizGenerator**\*** gv**;**  **if** **(**method **==** COMPLETE\_TOUR\_WEIGHT**)** **{**  gv **=** **new** GraphVizGenerator**(**matrix**,**path**,**fname**,false);**  gv**->**generateUndirectedGraph**();**  **}** **else** **{**  gv **=** **new** GraphVizGenerator**(**matrix**,**path**,**fname**,true);**  gv**->**generateDirectedGraph**();**  **}**  gv**->**generatePNG**();**  gv**->**generateSVG**();**    **// Terminate**  **delete** tsp**;**  **delete** gv**;**  **return** 0**;**  **}** |

# Contoh Masukan dan Keluaran

## Bobot Tur Lengkap

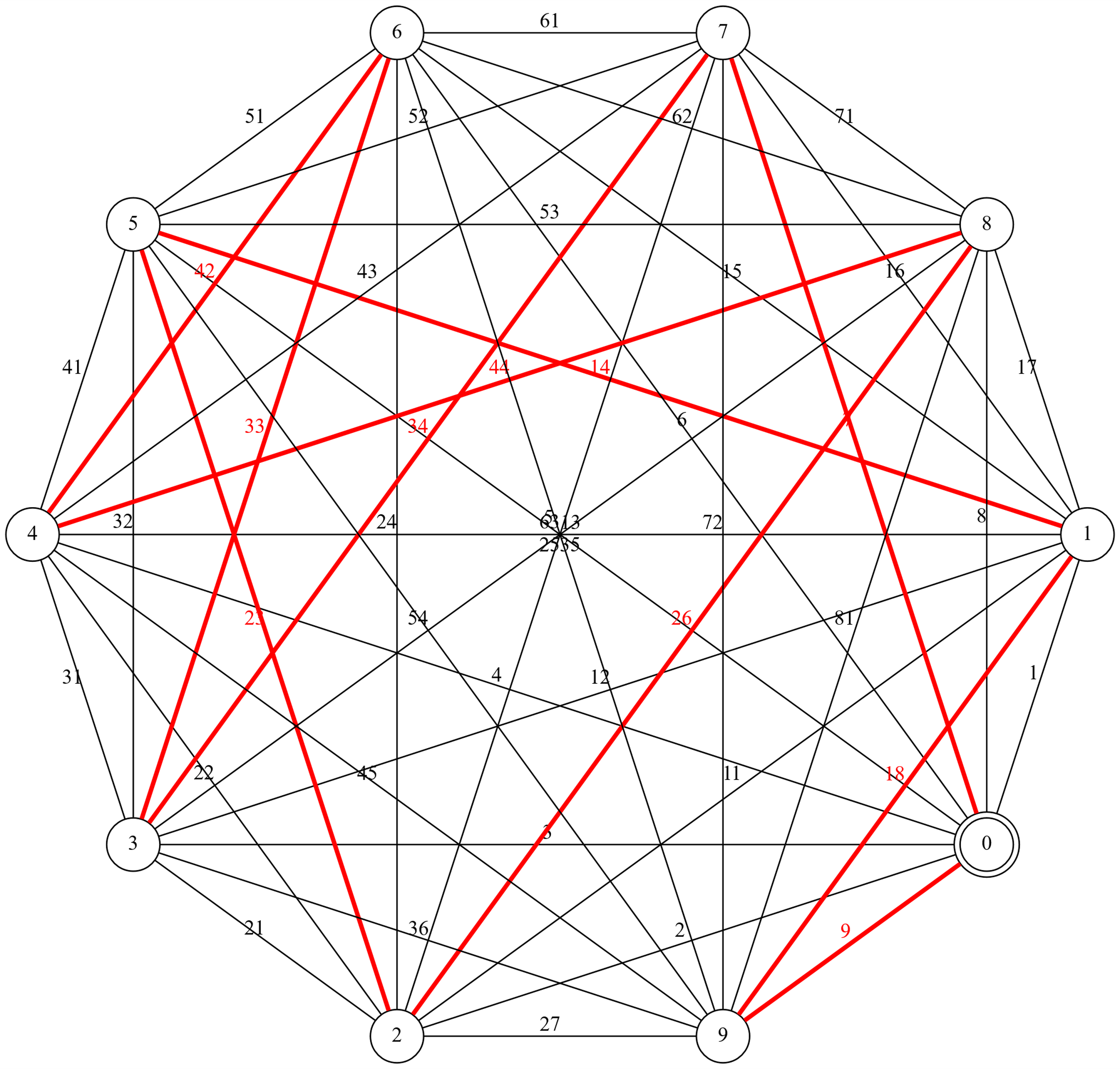
### *Test Case* 1:

|  |  |
| --- | --- |
| **Masukan** | **Keluaran** |
| 8  0 16 27 89 34 65 12 10  16 0 56 78 32 16 64 32  27 56 0 82 57 35 80 25  89 78 82 0 34 51 19 47  34 32 57 34 0 42 59 41  65 16 35 51 42 0 76 82  12 64 80 19 59 76 0 38  10 32 25 47 41 82 38 0 | Matrix:  inf 16 27 89 34 65 12 10  16 inf 56 78 32 16 64 32  27 56 inf 82 57 35 80 25  89 78 82 inf 34 51 19 47  34 32 57 34 inf 42 59 41  65 16 35 51 42 inf 76 82  12 64 80 19 59 76 inf 38  10 32 25 47 41 82 38 inf  Path: 0 -> 6 -> 3 -> 4 -> 7 -> 2 -> 5 -> 1 -> 0  Cost: 178.5  Elapsed time: 1000 us  GraphViz file generated to ../usr/test\_ctw\_2.gv  Graph image generated to ../usr/test\_ctw\_2.png  Graph image generated to ../usr/test\_ctw\_2.svg |



### *Test Case* 2:

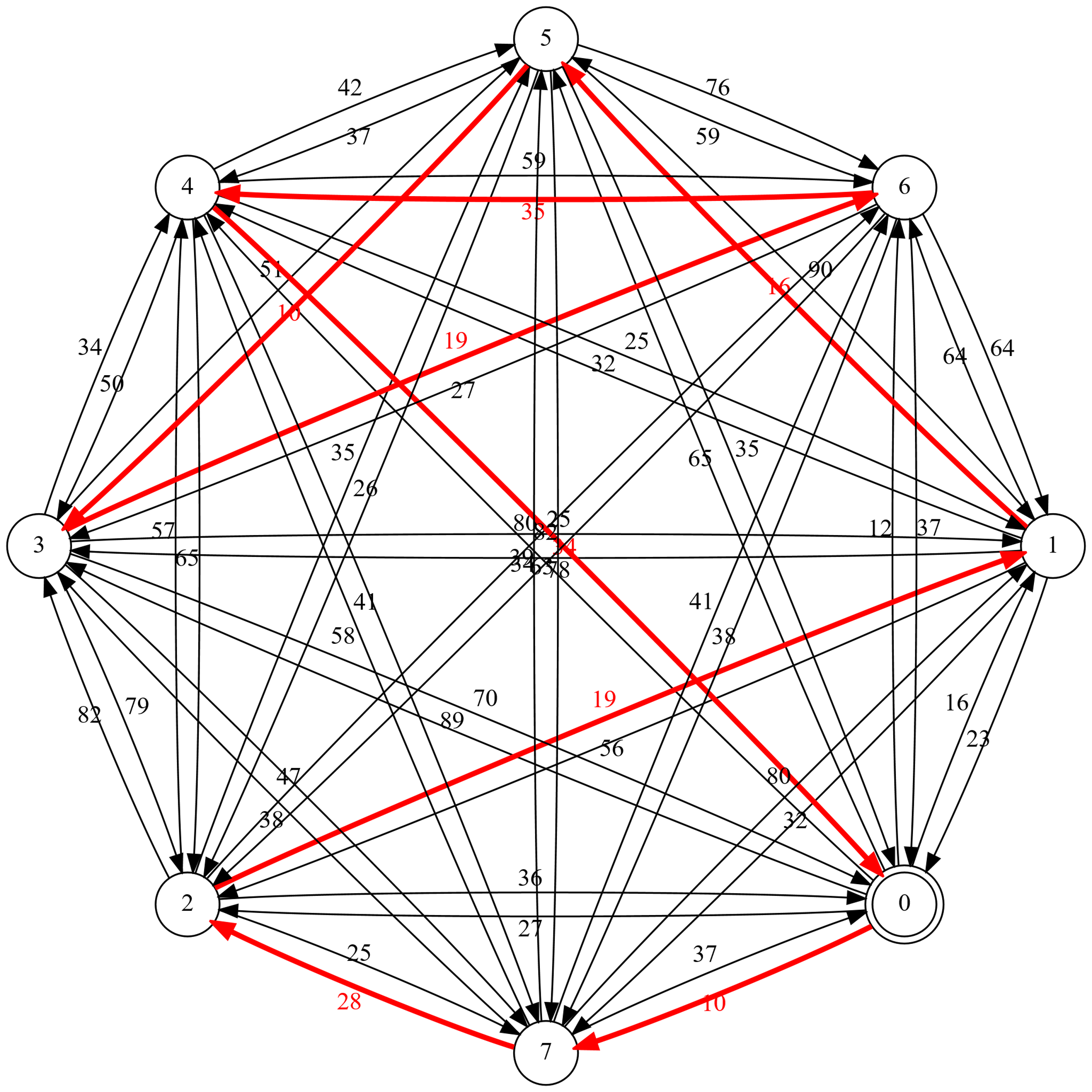
|  |  |
| --- | --- |
| **Masukan** | **Keluaran** |
| 10  0 1 2 3 4 5 6 7 8 9  1 0 11 12 13 14 15 16 17 18  2 11 0 21 22 23 24 25 26 27  3 12 21 0 31 32 33 34 35 36  4 13 22 31 0 41 42 43 44 45  5 14 23 32 41 0 51 52 53 54  6 15 24 33 42 51 0 61 62 63  7 16 25 34 43 52 61 0 71 72  8 17 26 35 44 53 62 71 0 81  9 18 27 36 45 54 63 72 81 0 | Matrix:  inf 1 2 3 4 5 6 7 8 9  1 inf 11 12 13 14 15 16 17 18  2 11 inf 21 22 23 24 25 26 27  3 12 21 inf 31 32 33 34 35 36  4 13 22 31 inf 41 42 43 44 45  5 14 23 32 41 inf 51 52 53 54  6 15 24 33 42 51 inf 61 62 63  7 16 25 34 43 52 61 inf 71 72  8 17 26 35 44 53 62 71 inf 81  9 18 27 36 45 54 63 72 81 inf  Path: 0 -> 7 -> 3 -> 6 -> 4 -> 8 -> 2 -> 5 -> 1 -> 9 -> 0  Cost: 152.5  Elapsed time: 1573121 us  GraphViz file generated to ../usr/test\_ctw\_3.gv  Graph image generated to ../usr/test\_ctw\_3.png  Graph image generated to ../usr/test\_ctw\_3.svg |



## *Reduced Cost Matrix*

### *Test Case* 1:

|  |  |
| --- | --- |
| **Masukan** | **Keluaran** |
| 8  0 16 27 89 34 65 12 10  23 0 56 78 32 16 64 32  36 19 0 82 57 35 80 25  70 25 79 0 34 51 19 47  34 25 65 50 0 42 59 41  35 90 26 10 37 0 76 82  37 64 63 27 35 59 0 38  37 80 28 38 58 39 41 0 | Matrix:  inf 16 27 89 34 65 12 10  23 inf 56 78 32 16 64 32  36 19 inf 82 57 35 80 25  70 25 79 inf 34 51 19 47  34 25 65 50 inf 42 59 41  35 90 26 10 37 inf 76 82  37 64 63 27 35 59 inf 38  37 80 28 38 58 39 41 inf  Path: 0 -> 7 -> 2 -> 1 -> 5 -> 3 -> 6 -> 4 -> 0  Cost: 171  Elapsed time: 0 us  GraphViz file generated to ../usr/test\_rcm\_2.gv  Graph image generated to ../usr/test\_rcm\_2.png  Graph image generated to ../usr/test\_rcm\_2.svg |



### *Test Case* 2:

|  |  |
| --- | --- |
| **Masukan** | **Keluaran** |
| 10  0 1 2 3 4 5 6 7 8 9  10 0 11 12 13 14 15 16 17 18  19 20 0 21 22 23 24 25 26 27  28 29 30 0 31 32 33 34 35 36  37 38 39 40 0 41 42 43 44 45  46 47 48 49 50 0 51 52 53 54  55 56 57 58 59 60 0 61 62 63  64 65 66 67 68 69 70 0 71 72  73 74 75 76 77 78 79 80 0 81  82 83 84 85 86 87 88 89 90 0 | Matrix:  inf 1 2 3 4 5 6 7 8 9  10 inf 11 12 13 14 15 16 17 18  19 20 inf 21 22 23 24 25 26 27  28 29 30 inf 31 32 33 34 35 36  37 38 39 40 inf 41 42 43 44 45  46 47 48 49 50 inf 51 52 53 54  55 56 57 58 59 60 inf 61 62 63  64 65 66 67 68 69 70 inf 71 72  73 74 75 76 77 78 79 80 inf 81  82 83 84 85 86 87 88 89 90 inf  Path: 0 -> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 0  Cost: 451  Elapsed time: 500 us  GraphViz file generated to ../usr/test\_rcm\_3.gv  Graph image generated to ../usr/test\_rcm\_3.png  Graph image generated to ../usr/test\_rcm\_3.svg |

