COE116L/C2–DATA STRUCTURES AND ALGORITHM LABORATORY

4th Quarter SY 2015-2016

**Traffic Simulation**

By

**Danielle Nicole Jawod**

**Eden Veronica S. Villar**

**Maria Exequiella M. Domingo**

**Ralph Emerson D. de Belen**

Analyn Yumang

Professor

Mapua Institute of Technology

March 2016

**ACKNOWLEDGEMENT**

Programming and finalizing this program was a tough task. Yet, the programmers were able to finish the program through the help of several people who committed themselves to make this project possible. The programmers would like to express and extend their deepest gratitude to the following:

Their parents and friends for the continuous assistance and support they have given throughout the progress in making the program;

Ms. Analyn Yumang, for her deep devotion and enthusiasm in teaching and guiding the process towards Intersection Traffic Simulation;

Mr. Jose Lazaro Jr., for introducing algorithms and other topics that are used to complete the program;

Lastly, the Lord, Heavenly King and the Creator of all things for being the guidance that gave unity and collaboration to the researchers and for giving His undying love and blessings to make this research possible.

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| TITLE PAGE | i |
| ACKNOWLEDGMENT | ii |
| TABLE OF CONTENTS | iii |
| ABSTRACT | 1 |
| Machine Problem Background and Introduction | 2 |
| Overview | 2 |
| Statement of the Problem | 2 |
| Objectives | 3 |
| Significance | 3 |
| Scope and Delimitations | 3 |
| Related Literature |  |
| Algorithms/Program Flow Chart | 5 |
| Class Diagram | 8 |
| Program Listing with Comments | 9 |
| Sample Test Runs with Screen Shots | 19 |
| BIBLIOGRAPHY | 21 |

**ABSTRACT**

Two-way Traffic Simulation. Simulation of transportation system is an application of computer software to better help plan, design, and operate transportation systems. Various local transportation agencies, academic institution, and consulting firms use simulation to aid their management of transportation network. The purpose of this machine problem is to demonstrate a two-way intersection road by using data structure algorithm. Data structures algorithm is used to manage the data and memory properly. The data structure that this machine problem used is queues. Queues is used to record the time of the traffic light. There might be lost in analytical or numerical treatment and can produce attractive visual demonstrations of present and future scenarios in studying simulation in transportation.

Keywords: queues, traffic simulation, data structure, algorithm, two-way traffic

**MACHINE PROBLEM BACKGROUND AND INTRODUCTION**

Intersection Traffic Simulation, also called the simulation of transport systems is the modelling of transportation systems such as junctions and intersections with the application of a computer software that helps in the planning, designing and operation of such transportation systems. This simulation is one of the most important areas of discipline in Traffic Engineering and Transportation Planning nowadays.

**OVERVIEW**

The group came up with an idea to use one-way intersection to better illustrate the logic and use of the traffic simulation. The car’s interval is set to 2 seconds. In every successful program run, the traffic light in the left road goes on green first then it will be followed by the stoplight in the down road. Once the traffic light on the down road turns red, the traffic light on the left road will go green again. Thus, the program does not end, or the cars’ movements and the traffic lights’ timers will continue to run, unless the user exits the console. Only one traffic light can go green at a certain time, the other traffic light is on red. Once a traffic light goes on green, the cars on the specified road will increase in number adding up to the queue otherwise, it will decrease in number decrementing the queue.

**STATEMENT OF THE PROBLEM**

Managing the traffic is common problem in our industry.  The possibilities of intersection of vehicles in the road are almost endless. A simulation and statistical analysis can help illustrate and simplify the complex process.

**OBJECTIVES**

The main objective of this program is to illustrate how the traffic simulation works using a one-way intersection. This program aims to show what big help a traffic simulation software can give in order for transportation systems to be planned, designed and operated nicely.

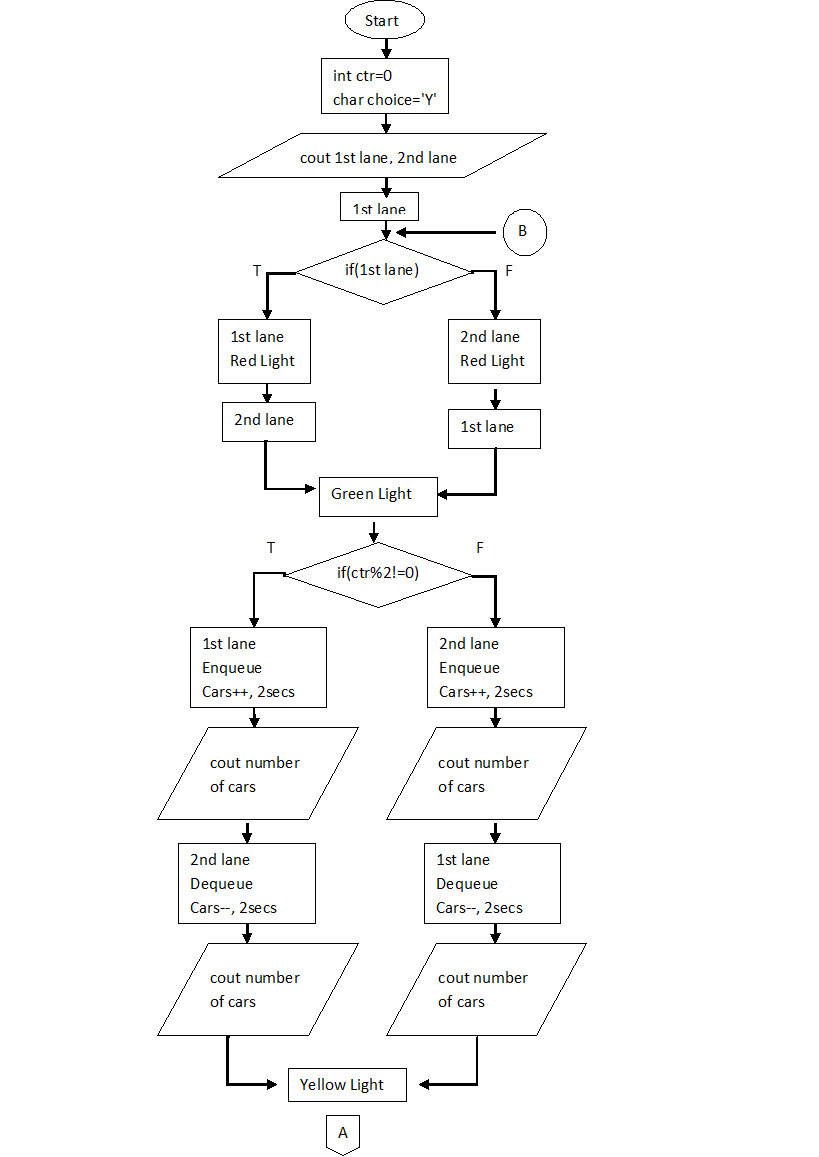
**SIGNIFICANCE**

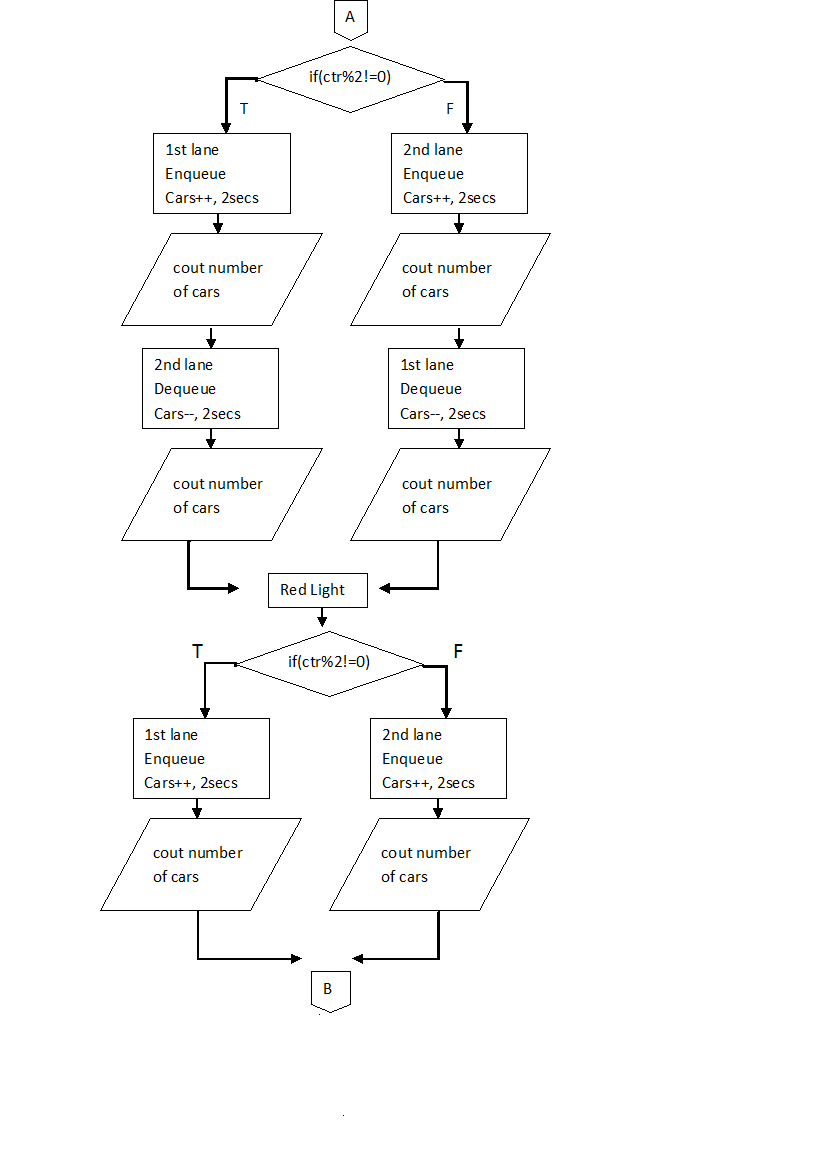
Traffic simulation in transportation is important because it can study models too complicated for analytical treatment. It can also be used for experimental studies for transportation systems. It can study detailed relations that might be lost in analytical treatment and can produce visual demonstrations of possible present and future scenarios.

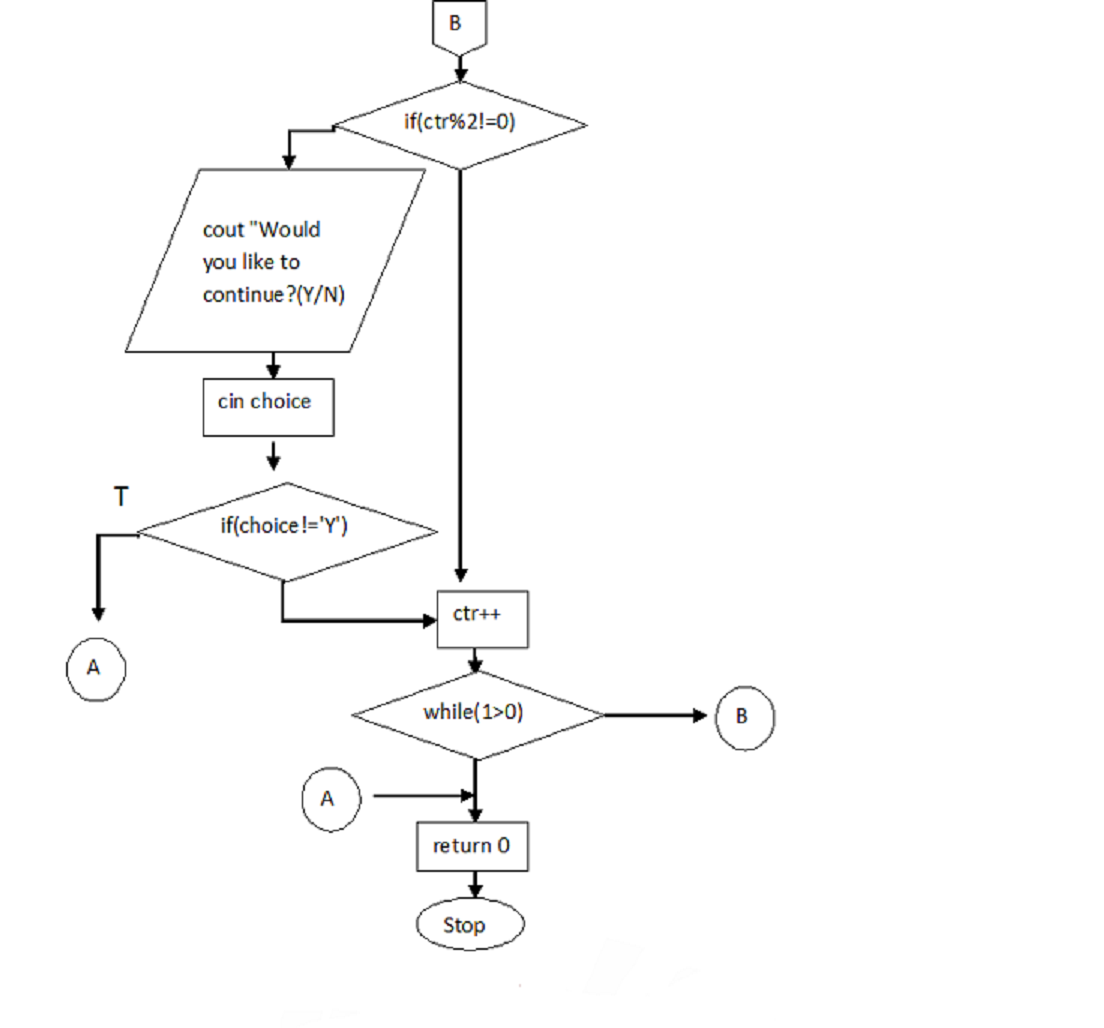
**SCOPE AND DELIMITATIONS**

This program is made to know and simulate how the traffic system works and its own rules and regulations. It uses Queue Data Structure to keep track the number of cars in a particular road. This program is very useful for those who are studying driving especially for those who are trying to acquire their own driver’s license. However, this program only covers one-way road. Only one road will give the cars a go signals at a time where they are freely to choose other roads.

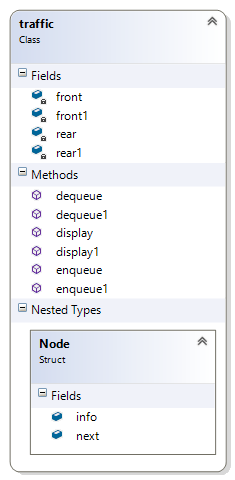
**ALGORITHMS/PROGRAM FLOW CHART**

****





**CLASS DIAGRAM**



**PROGRAM LISTING WITH COMMENTS**

traffic.h

#include<iostream>

using namespace std;

class traffic

{

private:

struct Node

{

int info;

Node \* next;

};

Node \* front = NULL;

Node \* front1 = NULL;

Node \* rear = NULL;

Node \*rear1 = NULL;

public:

void enqueue(int x, int y) // add cars

{

struct Node \* temp = new Node;

if (y == 0)

front = NULL;

temp->info = x;

temp->next = NULL;

if (front == NULL)

{

front = temp;

}

else

{

rear->next = temp;

}

rear = temp;

}

void enqueue1(int x, int y)

{

struct Node \* temp = new Node;

if (y == 0)

front1 = NULL;

temp->info = x;

temp->next = NULL;

if (front1 == NULL)

{

front1 = temp;

}

else

{

rear1->next = temp;

}

rear1 = temp;

}

int dequeue(int x, int y) // subtract cars

{

struct Node \* temp = front;

if (front == NULL)

return y;

else if (x == 0)

{

while (temp != NULL)

{

y = temp->info + y;

temp = temp->next;

}

y--;

return y;

}

else

{

if (y != 0)

y--;

return y;

}

}

int dequeue1(int x, int y)

{

struct Node \* temp = front1;

if (front1 == NULL)

{

return y;

}

else if (x == 0)

{

while (temp != NULL)

{

y = temp->info + y;

temp = temp->next;

}

y--;

return y;

}

else

{

if (y != 0)

y--;

return y;

}

}

int display()

{

int x = 0;

struct Node \*temp = new Node;

temp = front;

if (front == NULL)

return x;

else

{

while (temp != NULL)

{

x = temp->info + x;

temp = temp->next;

}

return x;

}

}

int display1()

{

int x = 0;

struct Node \*temp = new Node;

temp = front1;

if (front1 == NULL)

return x;

else

{

while (temp != NULL)

{

x = temp->info + x;

temp = temp->next;

}

return x;

}

}

};

traffic.cpp

// traffic4.cpp : Defines the entry point for the console application.

//

/\*#include "stdafx.h"

#include <iostream>

#include <stdio.h>

#include <time.h>

using namespace std;

void fiftyseconds()

{

char choice = 'N';

int x = 0, y = 0;

time\_t start;

time\_t current;

time(&start);

do {

time(&current);

} while (difftime(current, start) < 10.0);

}

void twoseconds()

{

time\_t start;

time\_t current;

time(&start);

do { time(&current);

} while (difftime(current, start) < 2.0);

}

void redlight()

{

cout<<"Lane1 - RED; Lane2 - RED;";

printf("\tLane3. - GREEN; Lane4 - GREEN\n");

fiftyseconds();

printf("Lane1 - RED; Lane2 - RED; \tLane3. - YELLOW; Lane4 - YELLOW\n");

twoseconds();

printf("Lane1 - RED; Lane2 - RED; \tLane3. - RED; Lane4 - RED\n");

twoseconds();

printf("Lane1 - GREEN; Lane2 - GREEN; \tLane3. - RED; Lane4 - RED\n");

fiftyseconds();

printf("Lane1 - YELLOW; Lane2 - YELLOW; \tLane3. - RED; Lane4 - RED\n");

twoseconds();

printf("Lane1 - RED; Lane2 - RED; \tLane3. - RED; Lane4 - RED\n");

twoseconds();

}

int main(){

printf("Stoplight 1.0\n\n");

while (1)

redlight();

return 0;

}\*/

#include "stdafx.h"

#include "traffic4.h"

# include <iostream>

#include<Windows.h>

using namespace std;

void gotoxy(int, int);

int main()

{

traffic k;

system("cls");

char cg[14] = "TRAFFIC LIGHT";

char car[20] = "Car Interval: 2 sec";

char lane[14] = "One way Lanes";

char cont[34] = "Would you like to continue?(Y/N):";

char choice = 'Y';

for (int i = 0; i<14; i++)

{

gotoxy(5, 5 + i);

cout << cg[i];

Sleep(100);

}

for (int i = 0; i<20; i++)

{

gotoxy(5 + i, 0);

cout << car[i];

Sleep(100);

}

for (int i = 0; i<14; i++)

{

gotoxy(5 + i, 1);

cout << lane[i];

Sleep(100);

}

char c[14] = "|||||||||||||";

char d[60] = "-----------------------------------------------------------";

char e[70] = "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_";

char f[20] = "lllllllllllllllllll";

gotoxy(10, 8);

cout << "First Lane";

gotoxy(41, 17);

cout << "Second Lane";

gotoxy(10, 12);

cout << d;

gotoxy(10, 13);

cout << d;

gotoxy(10, 14);

cout << d;

gotoxy(10, 5);

cout << d;

gotoxy(10, 6);

cout << d;

gotoxy(10, 7);

cout << d;

for (int i = 0; i<70; i++)

{

gotoxy(9 + i, 9);

cout << e[i];

Sleep(100);

}

for (int i = 0; i<14; i++)

{

gotoxy(33, 3 + i);

cout << c[i];

}

for (int i = 0; i<14; i++)

{

gotoxy(34, 3 + i);

cout << c[i];

}

for (int i = 0; i<14; i++)

{

gotoxy(35, 3 + i);

cout << c[i];

}

for (int i = 0; i<14; i++)

{

gotoxy(41, 3 + i);

cout << c[i];

}

for (int i = 0; i<14; i++)

{

gotoxy(42, 3 + i);

cout << c[i];

}

for (int i = 0; i<14; i++)

{

gotoxy(43, 3 + i);

cout << c[i];

}

for (int i = 0; i<20; i++)

{

gotoxy(38, 2 + i);

cout << f[i];

Sleep(100);

}

int sig; sig = 41;

int sig1 = 18;

int z = 0;

int ctr = 0;

do

{

int x = 0;

int y = 0;

if (x == 0)

{

gotoxy(41, 18);

cout << "red";

}

if (sig == 10)

{

sig = 41;

sig1 = 18;

}

else

{

sig = 10;

sig1 = 9;

}

gotoxy(sig, sig1);

cout << "red ";

Sleep(1500);

gotoxy(sig, sig1);

cout << "green ";

for (int i = 0; i < 5; i++)

{

if (ctr % 2 != 0)

{

k.enqueue(1, x);

z = k.display();

gotoxy(10, 10);

cout << "Cars on First Lane: " << z;

y = k.dequeue1(x, y);

gotoxy(41, 19);

cout << "Cars on Second Lane: " << y;

Sleep(2000);

x++;

}

else

{

k.enqueue1(1, x);

z = k.display1();

gotoxy(41, 19);

cout << "Cars on Second Lane: " << z;

y = k.dequeue(x, y);

gotoxy(10, 10);

cout << "Cars on First Lane: " << y;

Sleep(2000);

x++;

}

}

gotoxy(sig, sig1);

cout << "yellow ";

if (ctr % 2 != 0)

{

k.enqueue(1, x);

z = k.display();

gotoxy(10, 10);

cout << "Cars on First Lane: " << z;

y = k.dequeue1(x, y);

gotoxy(41, 19);

cout << "Cars on Second Lane: " << y;

Sleep(2000);

x++;

}

else

{

k.enqueue1(1, x);

z = k.display1();

gotoxy(41, 19);

cout << "Cars on Second Lane: " << z;

y = k.dequeue(x, y);

gotoxy(10, 10);

cout << "Cars on First Lane: " << y;

Sleep(2000);

x++;

}

gotoxy(sig, sig1);

cout << "red ";

Sleep(1500);

if (ctr % 2 != 0)

{

k.enqueue(1, x);

z = k.display();

gotoxy(10, 10);

cout << "Cars on First Lane: " << z;

Sleep(2000);

x++;

}

else

{

k.enqueue1(1, x);

z = k.display1();

gotoxy(41, 19);

cout << "Cars on Second Lane: " << z;

Sleep(2000);

x++;

}

if (ctr % 2 != 0)

{

for (int i = 0; i<34; i++)

{

gotoxy(5 + i, 2);

cout << cont[i];

Sleep(100);

}

cin >> choice;

if (choice != 'Y')

return 0;

else

{

for (int i = 0; i < 35; i++)

{

gotoxy(5 + i, 2);

cout << " ";

}

}

}

ctr++;

} while (1>0);

return 0;

}

void gotoxy(int x, int y)

{

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

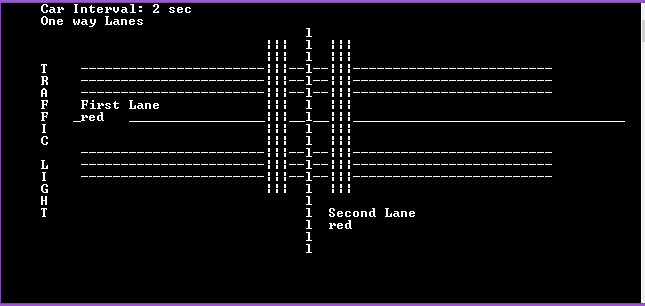
\_COORD pos; pos.X = x; pos.Y = y;

SetConsoleCursorPosition(hConsole, pos);

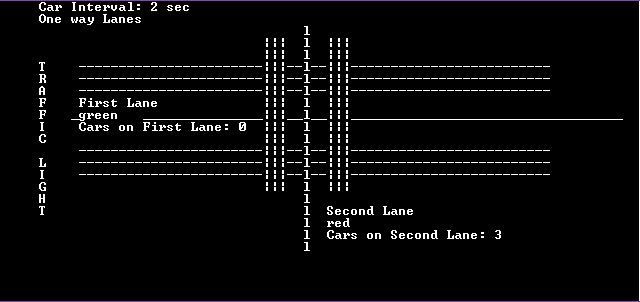
}

**Sample test runs with screen shots**

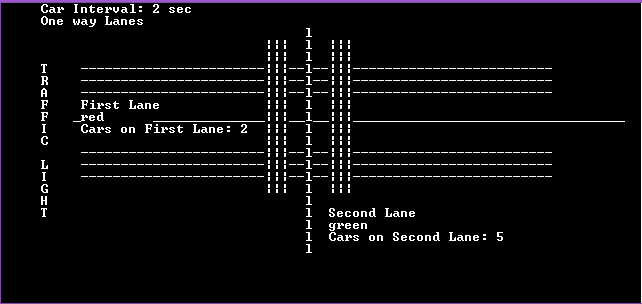
TEST RUNS WITH SCREEN SHOTS



Both traffic lights are initialized as red and the car interval is set to 2 seconds.



The number of cars will be counted on the lane with red light.



Once the light turns to green. Cars will move on the next lane and will then start counting the cars on that lane.

**BIBLIOGRAPHY**

Malik, D.S. (2010). Data Structures Using C++, 2nd Edition, Course Technology, Cengage Learning, Boston, USA.