Get on the Bus ------ Simple or Not?

New Mexico

Supercomputing Challenge

Final Report

April 1, 2009

Team 45

Homeschool/LAHS

Team Member:

Isaac Koh

ifckoh@gmail.com

Teacher:

Aik-Siong Koh

askoh@askoh.com

Mentor:

Aik-Siong Koh

Summary:

This year for my Supercomputing Challenge problem, I wanted to create a program for the Atomic City Transit bus system that would let a user input his/her starting point, destination, and desired time of arrival, and the program would tell the user when and where to catch the bus, and when to switch buses if necessary. I had no programming experience prior to this Challenge and started out learning C#. I went through the tutorials that Microsoft provided and then proceeded to building my program. At first it looked like it was a lot more than I could chew, but gradually with my teacher's help the program took shape. We had several iterations and by the Midterm presentations, we could produce the bus trips (circuits) and the times that they ran on. After the presentations we turned to solving the case where the start point and destination were on the same Route. We first looked for the Route that had both the start and end on it. Then we looked for the Trip that had the selected time in it. That set of code was good until we noticed that some Trips pass certain stops multiple times during the course of the circulation. That made the program say that we would arrive at our destination earlier than we caught the bus, which is physically impossible. To solve this problem we strung together two consecutive trips. Then we found the destination, and traced backwards until we met the starting point. After that, we turned to the problem of switching buses. We had the code look for the Routes that had the start and the Routes that had the end. Then we found the intersection of the stops between each start and each end Route. Using the end Route we traced backwards from the end to the nearest intersection. Then we treated the intersection time as the end time and had the program trace back to the starting point along the start Route. Thus we were able to return all the information from start to end including the bus change. This was our last major obstacle and our program is functioning very nicely now. In the future I will put my program on a website, and then look into making a similar program for Santa Fe.

Resources:

Microsoft Visual C# 2008 Express Edition

Google Search

Atomic City Transit Schedule

Compaq Presario V3000 Laptop

Problem Investigated:

I wanted to make a program for the Los Alamos Atomic City Transit bus system that lets the user input his/her starting place, destination, and the time they want to arrive. Then the program will return when and where to catch the bus, when and where to switch if necessary and when the user will arrive.

Methods:

1. Create the Route class to store the bus route name, list of TripPatterns, and a list of all Trips for the day.
2. Create Trip class to store list of stops, list of arrival times, list of waiting times, and name. A Trip is one circuit that starts and ends at the Transit Center.
3. Create TripPattern class as sub-class of Trip to store the schedule repetition time (minutes), start and end time of schedule, and a list of Trips.
4. Input the Atomic City Transit schedule into TripPatterns and have them generate the Trips for all the Routes thoughout the day. (Appendix A)

Example:

```
route1.name = "Route 1";

route1.cTripPattern = new List<TripPattern>();
aTripPattern = new TripPattern();
aTripPattern.cDest = new List<string>();
aTripPattern.name = "Route 1";
aTripPattern.cDest.Add("Transit Center");
aTripPattern.cDest.Add("Diamond & Trinity");
aTripPattern.cDest.Add("Trinity & Oppenheimer");
aTripPattern.cDest.Add("Trinity & 15th");
aTripPattern.cDest.Add("4th & Central");
aTripPattern.cDest.Add("Central & 15th");
aTripPattern.cDest.Add("Central & Canyon");
aTripPattern.cDest.Add("Canyon & Diamond");
aTripPattern.cDest.Add("Transit Center");

int[] c1ArrivalTime = {  8 ,  10 ,  11 ,  12 ,  14 ,  16 ,  18 ,  21 ,  23  };
aTripPattern.cArrivalTime = c1ArrivalTime.ToList();
aTripPattern.repeatMinutes = 20; // repeats every 20 minutes
aTripPattern.firstRepeatTime = 5 * 60 + 40; // 5:40 AM in minutes
aTripPattern.lastRepeatTime = 19 * 60; // 7:00 PM in minutes
aTripPattern.setcTrip();
route1.cTripPattern.Add(aTripPattern);
```

TripPatterns will generate Trips like the one below.
Example Trip:

cDest	cArrivalTime
("Transit Center");	7:08 AM
("Diamond & Trinity");	7:10 AM
("Trinity & Oppenheimer");	7:11 AM
("Trinity & 15th");	7:12 AM
("4th & Central");	7:14 AM
("Central & 15th");	7:16 AM
("Central & Canyon");	7:18 AM
("Canyon & Diamond");	7:21 AM
("Transit Center");	7:23 AM

5. Write algorithm as explained in the section below. The source code is in Appendix B.

Problems and How I Solved Them:

The programming was initially difficult. I was not familiar with the language and my project required many operations that had not been covered in the tutorials, but as the project continued I got the hang of it.

Inputing times for the Downtown route was rather straight forward. But we found that all the other routes had three different sub-schedules within the day: all day, AM peak, and PM peak. This led to us creating TripPatterns that generated individual Trips (circuits) and treated each of the above sub-schedules as a different TripPattern. Therefore each route had several TripPatterns, and each contained several Trips that each represented one circulation. A Trip has the stops and the times of arrival.

The White Rock route has a 14 minute stop in the middle of it. We then added another property in Trip called WaitTime that we could set and left it at zero for all the other stops that didn't need it.

We had simplified our code so much that when it started returning the time to get on and off, we were told to get on after reaching our destination. This happened because we were finding the solution all within one Trip. Also, some Trips visited the same stops more that once. To fix this we strung two consecutive Trips together and traced from the end of the combined Trips backwards until we found the endpoint. Then we continued backwards until we found the endpoint again or the start point. If we found the endpoint again we used that instance instead of

the previous one, and continued backwards to find the start point. When we find the start point, we have the solution and return the following:

Start Time, Starting Point, Bus Route

End Time, Destination

The above worked when consecutive Trips did not overlap, but during the peak hours consecutive Trips often overlapped. Therefore we had to search for the nearest non-overlapping Trip before joining the two Trips.

The above algorithm works when a start point and end point are on the same Route. When they are on separate Routes we adjusted the algorithm as follows: We find the Routes that contain the end and the Routes that contain the start. Then we find the common stops of each combination of each start Route and end Route. Then starting from the end point we trace backwards along the end Route to the nearest common stop and treat that as the new starting point. Since the new starting point and the original end point are now on the same Route we can use the previous algorithm to find the best times for this connecting Trip. Now we make the connection stop the new destination and find the best time between it and the actual starting point. Hence we have the solution for the following:

Start Time, Starting Point, Bus Route

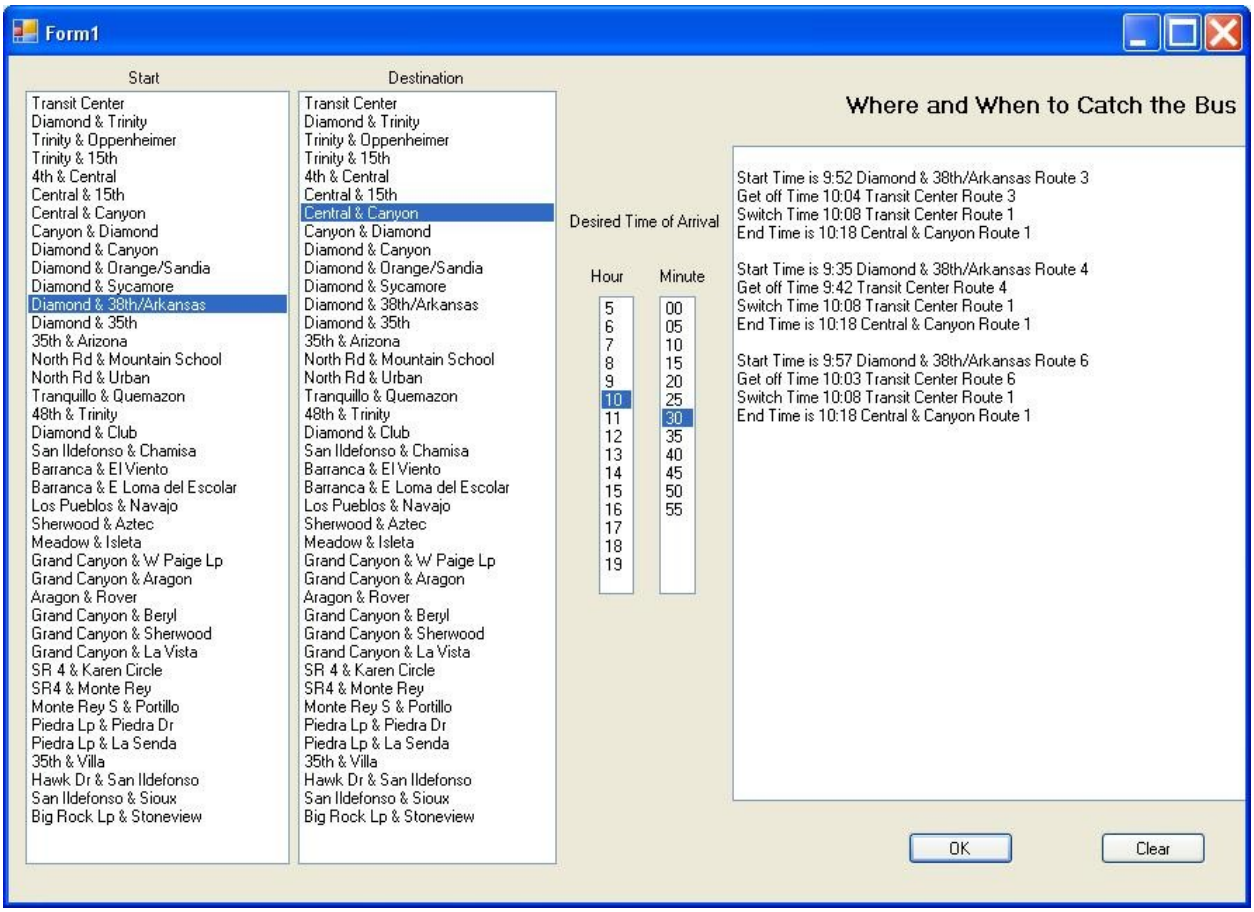
Arrival at Intersection Time, Intersection Point

Connection Time, Connection Bus Route

End Time, Destination

Results:

For example:



Conclusions:

It is possible to create a program to make using a bus system easier. I have learned a lot from this project, and will learn more as I polish it. It is satisfying to know that this program can actually help people. Looking ahead I will put this program on a website, and make it accessible from the web browser. Then I will look into making a similar program for Santa Fe.

Code:

See Appendix B

Achievements:

There is no other program for the Atomic City Transit that will tell you when and where to catch the bus. This saves you from having to go through the schedule every time.

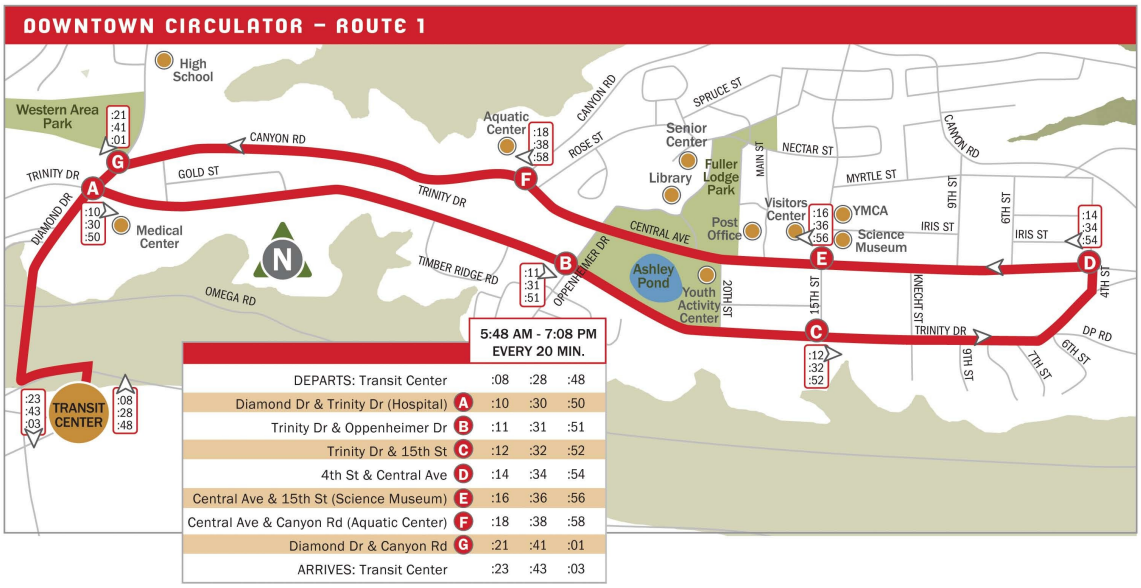
Chicago has a program that tells you when the bus will arrive at a certain place, but you still have to plan out your own route.
http://www.ctabustracker.com/bustime/home.jsp

Acknowledgements:

I would like to thank my dad for being my teacher, sponsor, and for helping me do this project.
I would also like to thank the Supercomputing Challenge organizers for the encouragement and feedback throughout this Challenge.

Appendix A

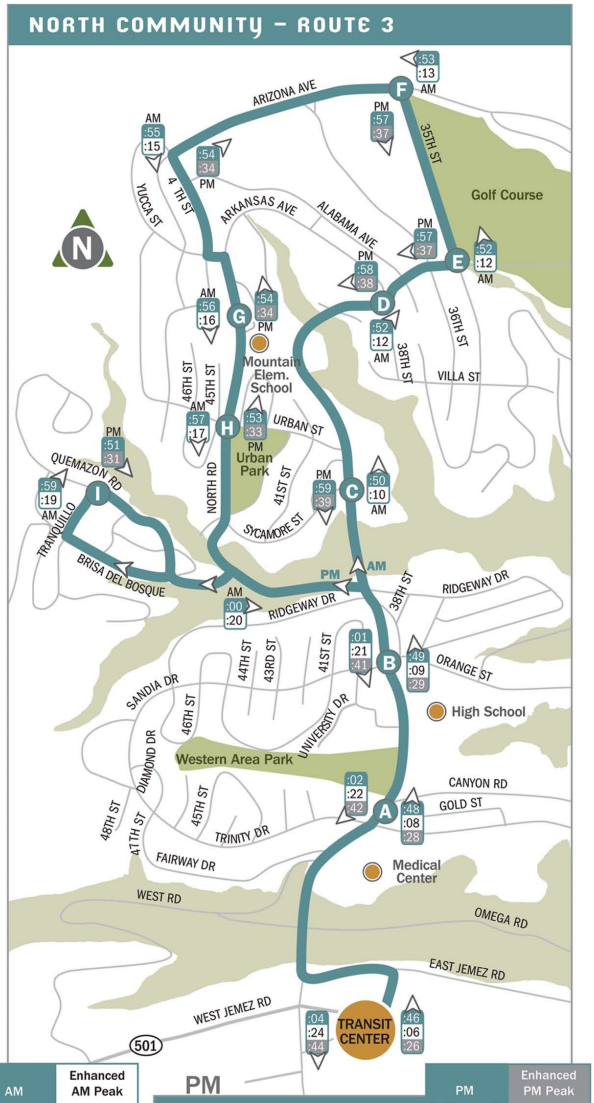
The letters stand for major points on a route. The numbers stand for minutes, and they repeat every hour. Look at A on Route 1. The bus passes there at 5:50 AM, 6:10 AM, 6:30 AM, 6:50 AM, 7:10 AM and so on until 7:10 PM. The same goes for all the other stops and routes.



ATOMIC CITY TRANSIT

Route 2 does not exist.

NORTH COMMUNITY – ROUTE 3



AM

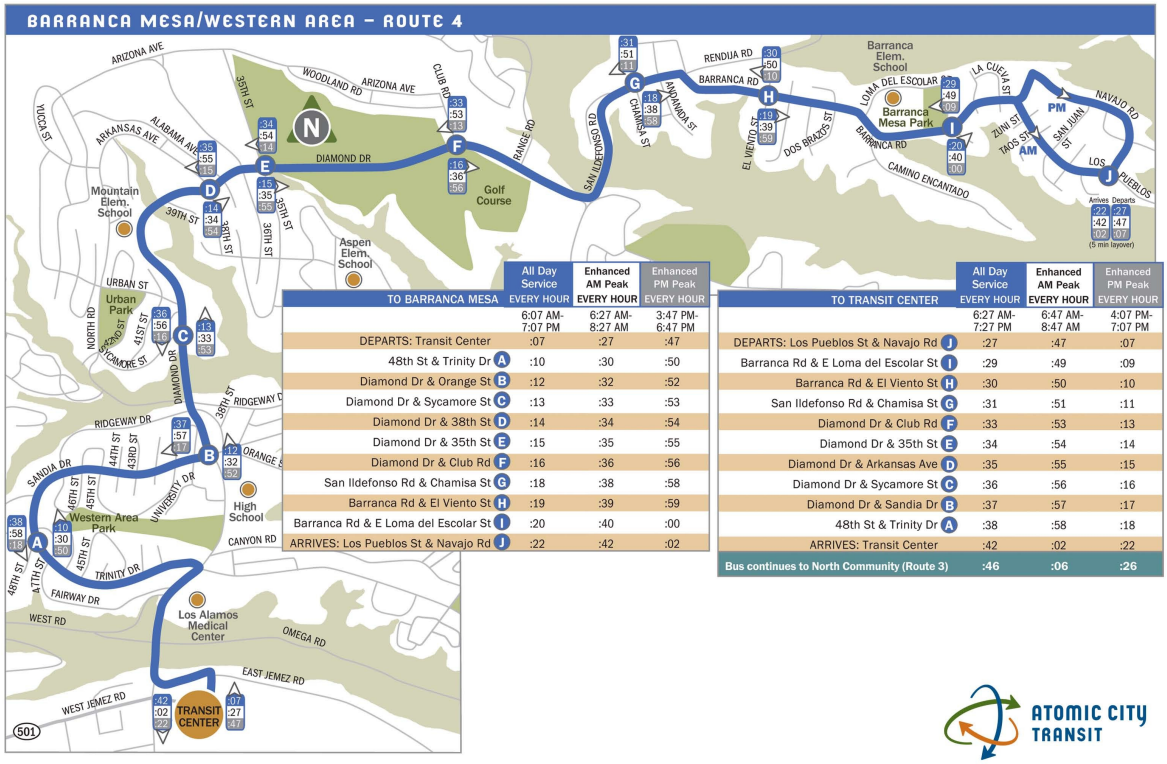
	AM EVERY HOUR	Enhanced AM Peak EVERY HOUR
	5:46 AM-11:46 AM	6:06 AM-8:06 AM
DEPARTS: Transit Center	:46	:06
Diamond Dr & Canyon (A)	:48	:08
Diamond Dr & Orange St (B)	:49	:09
Diamond Dr & Sycamore St (C)	:50	:10
Diamond Dr & 38th St (D)	:52	:12
35th St & Diamond Dr (E)	:52	:12
35th St & Arizona Ave (F)	:53	:13
North Rd & Mountain School (G)	:56	:16
North Rd & Urban St (H)	:57	:17
Tranquillo St & Quemazon Rd (I)	:59	:19
Diamond Dr & Sandia Dr (B)	:01	:21
Diamond Dr & Canyon Rd (A)	:02	:22
ARRIVES: Transit Center	:04	:24
Bus continues to Barranca (Route 4)	**:07**	**:27**

PM

	PM EVERY HOUR	Enhanced PM Peak EVERY HOUR
	12:46 PM-6:46 PM	3:26 PM-7:26 PM
DEPARTS: Transit Center	:46	:26
Diamond Dr & Canyon Rd (A)	:48	:28
Diamond Dr & Orange St (B)	:49	:29
Tranquillo St & Quemazon Rd (I)	:51	:31
North Rd & Urban St (H)	:53	:33
North Rd & Mountain School (G)	:54	:34
35th St & Arizona Ave (F)	:57	:37
35th St & Diamond Dr (E)	:57	:37
Diamond Dr & Arkansas Ave (D)	:58	:38
Diamond Dr & Sycamore St (C)	:59	:39
Diamond Dr & Sandia Dr (B)	:01	:41
Diamond Dr & Canyon Rd (A)	:02	:42
ARRIVES: Transit Center	:04	:44
Bus continues to Barranca (Route 4)	**:07**	**:47**

ATOMIC CITY TRANSIT

BARRANCA MESA/WESTERN AREA – ROUTE 4



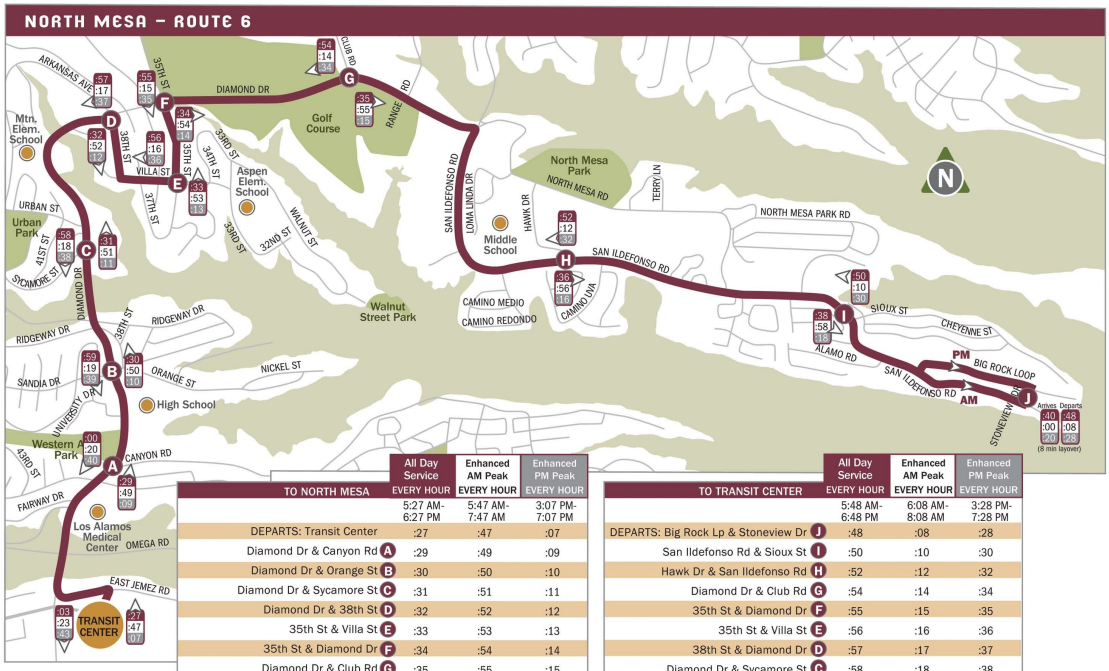
TO BARRANCA MESA	All Day Service EVERY HOUR	Enhanced AM Peak EVERY HOUR	Enhanced PM Peak EVERY HOUR
	6:07 AM-7:07 PM	6:27 AM-8:27 AM	3:47 PM-6:47 PM
DEPARTS: Transit Center	:07	:27	:47
48th St & Trinity Dr (A)	:10	:30	:50
Diamond Dr & Orange St (B)	:12	:32	:52
Diamond Dr & Sycamore St (C)	:13	:33	:53
Diamond Dr & 38th St (D)	:14	:34	:54
Diamond Dr & 35th St (E)	:15	:35	:55
Diamond Dr & Club Rd (F)	:16	:36	:56
San Ildefonso Rd & Chamisa St (G)	:18	:38	:58
Barranca Rd & El Viento St (H)	:19	:39	:59
Barranca Rd & E Loma del Escolar St (I)	:20	:40	:00
ARRIVES: Los Pueblos St & Navajo Rd (J)	:22	:42	:02

TO TRANSIT CENTER	All Day Service EVERY HOUR	Enhanced AM Peak EVERY HOUR	Enhanced PM Peak EVERY HOUR
	6:27 AM-7:27 PM	6:47 AM-8:47 AM	4:07 PM-7:07 PM
DEPARTS: Los Pueblos St & Navajo Rd (J)	:27	:47	:07
Barranca Rd & E Loma del Escolar St (I)	:29	:49	:09
Barranca Rd & El Viento St (H)	:30	:50	:10
San Ildefonso Rd & Chamisa St (G)	:31	:51	:11
Diamond Dr & Club Rd (F)	:33	:53	:13
Diamond Dr & 35th St (E)	:34	:54	:14
Diamond Dr & Arkansas Ave (D)	:35	:55	:15
Diamond Dr & Sycamore St (C)	:36	:56	:16
Diamond Dr & Sandia Dr (B)	:37	:57	:17
48th St & Trinity Dr (A)	:38	:58	:18
ARRIVES: Transit Center	:42	:02	:22
Bus continues to North Community (Route 3)	**:46**	**:06**	**:26**

ATOMIC CITY TRANSIT

ATOMIC CITY TRANSIT

NORTH MESA – ROUTE 6



TO NORTH MESA	All Day Service EVERY HOUR	Enhanced AM Peak EVERY HOUR	Enhanced PM Peak EVERY HOUR
	5:27 AM-6:27 PM	5:47 AM-7:47 AM	3:07 PM-7:07 PM
DEPARTS: Transit Center	:27	:47	:07
Diamond Dr & Canyon Rd (A)	:29	:49	:09
Diamond Dr & Orange St (B)	:30	:50	:10
Diamond Dr & Sycamore St (C)	:31	:51	:11
Diamond Dr & 38th St (D)	:32	:52	:12
35th St & Villa St (E)	:33	:53	:13
35th St & Diamond Dr (F)	:34	:54	:14
Diamond Dr & Club Rd (G)	:35	:55	:15
Hawk Dr & San Ildefonso Rd (H)	:36	:56	:16
San Ildefonso Rd & Sioux St (I)	:38	:58	:18
ARRIVES: Big Rock Lp & Stoneview Dr (J)	:40	:00	:20

TO TRANSIT CENTER	All Day Service EVERY HOUR	Enhanced AM Peak EVERY HOUR	Enhanced PM Peak EVERY HOUR
	5:48 AM-6:48 PM	6:08 AM-8:08 AM	3:28 PM-7:28 PM
DEPARTS: Big Rock Lp & Stoneview Dr (J)	:48	:08	:28
San Ildefonso Rd & Sioux St (I)	:50	:10	:30
Hawk Dr & San Ildefonso Rd (H)	:52	:12	:32
Diamond Dr & Club Rd (G)	:54	:14	:34
35th St & Diamond Dr (F)	:55	:15	:35
35th St & Villa St (E)	:56	:16	:36
38th St & Diamond Dr (D)	:57	:17	:37
Diamond Dr & Sycamore St (C)	:58	:18	:38
Diamond Dr & Sandia Dr (B)	:59	:19	:39
Diamond Dr & Canyon Rd (A)	:00	:20	:40
ARRIVES: Transit Center	:03	:23	:43
Bus continues to White Rock (Route 5)	**:07**	**:27**	**:47**

North Mesa and White Rock routes are connected so you can stay on the bus and travel from North Mesa to White Rock.

ATOMIC CITY TRANSIT

Appendix B

Form 1:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace test6
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      public List<Route> allRoutes = new List<Route>();
      public Route route1 = new Route();
      public Route route3 = new Route();
      public Route route4 = new Route();
      public Route route5 = new Route();
      public Route route6 = new Route();
      public TripPattern pattern1 = new TripPattern();
      public List<string> cDest = new List<string>();
      public TripPattern aTripPattern;
      public TripPattern bTripPattern;
      public TripPattern cTripPattern;
      public TripPattern dTripPattern;
      public TripPattern eTripPattern;
      public TripPattern fTripPattern;
      public TripPattern gTripPattern;
      public TripPattern hTripPattern;
      public TripPattern iTripPattern;
      public TripPattern jTripPattern;
      public TripPattern kTripPattern;
      public TripPattern lTripPattern;
      public TripPattern mTripPattern;
      public TripPattern nTripPattern;
      public TripPattern oTripPattern;
      //public List routesWithEnd;

      private void Form1_Load(object sender, EventArgs e)
      {


         #region Route1

         route1.name = "Route 1";
```

```
route1.cTripPattern = new List<TripPattern>();
aTripPattern = new TripPattern();
aTripPattern.cDest = new List<string>();
aTripPattern.name = "Route 1";
aTripPattern.cDest.Add("Transit Center");
aTripPattern.cDest.Add("Diamond & Trinity");
aTripPattern.cDest.Add("Trinity & Oppenheimer");
aTripPattern.cDest.Add("Trinity & 15th");
aTripPattern.cDest.Add("4th & Central");
aTripPattern.cDest.Add("Central & 15th");
aTripPattern.cDest.Add("Central & Canyon");
aTripPattern.cDest.Add("Canyon & Diamond");
aTripPattern.cDest.Add("Transit Center");

int[] c1ArrivalTime = {  8 ,  10 ,  11 ,  12 ,  14 ,  16 ,  18 ,  21 ,  23  };
aTripPattern.cArrivalTime = c1ArrivalTime.ToList();
aTripPattern.repeatMinutes = 20;
aTripPattern.firstRepeatTime = 5 * 60 + 40;
aTripPattern.lastRepeatTime = 19 * 60;
aTripPattern.setcTrip();
route1.cTripPattern.Add(aTripPattern);

#endregion

#region Route3

route3.name = "Route 3";
route3.cTripPattern = new List<TripPattern>();
bTripPattern = new TripPattern();
bTripPattern.cDest = new List<string>();
bTripPattern.name = "Route 3 AM";
bTripPattern.cDest.Add("Transit Center");
bTripPattern.cDest.Add("Diamond & Canyon");
bTripPattern.cDest.Add("Diamond & Orange/Sandia");
bTripPattern.cDest.Add("Diamond & Sycamore");
bTripPattern.cDest.Add("Diamond & 38th/Arkansas");
bTripPattern.cDest.Add("Diamond & 35th");
bTripPattern.cDest.Add("35th & Arizona");
bTripPattern.cDest.Add("North Rd & Mountain School");
bTripPattern.cDest.Add("North Rd & Urban");
bTripPattern.cDest.Add("Tranquillo & Quemazon");
bTripPattern.cDest.Add("Diamond & Orange/Sandia");
bTripPattern.cDest.Add("Diamond & Canyon");
bTripPattern.cDest.Add("Transit Center");

int[] c3ArrivalTime = { 46, 48, 49, 50, 52, 52, 53, 56, 57, 59, 61, 62, 64 };
int[] cWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 42 };
bTripPattern.cArrivalTime = c3ArrivalTime.ToList();
bTripPattern.cWaitTime = cWaitTime.ToList();
bTripPattern.repeatMinutes = 60;
bTripPattern.firstRepeatTime = 5 * 60;
bTripPattern.lastRepeatTime = 11 * 60;
bTripPattern.setcTrip();
route3.cTripPattern.Add(bTripPattern);

cTripPattern = new TripPattern();
```

```
cTripPattern.cDest = new List<string>();
cTripPattern.name = "Route 3 AM";
cTripPattern.cDest.Add("Transit Center");
cTripPattern.cDest.Add("Diamond & Canyon");
cTripPattern.cDest.Add("Diamond & Orange/Sandia");
cTripPattern.cDest.Add("Diamond & Sycamore");
cTripPattern.cDest.Add("Diamond & 38th/Arkansas");
cTripPattern.cDest.Add("Diamond & 35th");
cTripPattern.cDest.Add("35th & Arizona");
cTripPattern.cDest.Add("North Rd & Mountain School");
cTripPattern.cDest.Add("North Rd & Urban");
cTripPattern.cDest.Add("Tranquillo & Quemazon");
cTripPattern.cDest.Add("Diamond & Orange/Sandia");
cTripPattern.cDest.Add("Diamond & Canyon");
cTripPattern.cDest.Add("Transit Center");

int[] c3aArrivalTime = { 06, 08, 09, 10, 12, 12, 13, 16, 17, 19, 21, 22, 24 };
int[] c3aWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 42 };
cTripPattern.cArrivalTime = c3aArrivalTime.ToList();
cTripPattern.cWaitTime = c3aWaitTime.ToList();
cTripPattern.repeatMinutes = 60;
cTripPattern.firstRepeatTime = 6 * 60;
cTripPattern.lastRepeatTime = 8 * 60;
cTripPattern.setcTrip();
route3.cTripPattern.Add(cTripPattern);

dTripPattern = new TripPattern();
dTripPattern.cDest = new List<string>();
dTripPattern.name = "Route 3 PM";
dTripPattern.cDest.Add("Transit Center");
dTripPattern.cDest.Add("Diamond & Canyon");
dTripPattern.cDest.Add("Diamond & Orange/Sandia");
dTripPattern.cDest.Add("Tranquillo & Quemazon");
dTripPattern.cDest.Add("North Rd & Urban");
dTripPattern.cDest.Add("North Rd & Mountain School");
dTripPattern.cDest.Add("35th & Arizona");
dTripPattern.cDest.Add("Diamond & 35th");
dTripPattern.cDest.Add("Diamond & 38th/Arkansas");
dTripPattern.cDest.Add("Diamond & Sycamore");
dTripPattern.cDest.Add("Diamond & Orange/Sandia");
dTripPattern.cDest.Add("Diamond & Canyon");
dTripPattern.cDest.Add("Transit Center");

int[] c3pArrivalTime = { 46, 48, 49, 51, 53, 54, 57, 57, 58, 59, 61, 62, 64 };
int[] c3pWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 42 };
dTripPattern.cArrivalTime = c3pArrivalTime.ToList();
dTripPattern.cWaitTime = c3pWaitTime.ToList();
dTripPattern.repeatMinutes = 60;
dTripPattern.firstRepeatTime = 12 * 60;
dTripPattern.lastRepeatTime = 18 * 60;
dTripPattern.setcTrip();
route3.cTripPattern.Add(dTripPattern);

eTripPattern = new TripPattern();
eTripPattern.cDest = new List<string>();
eTripPattern.name = "Route 3 PM";
```

```
eTripPattern.cDest.Add("Transit Center");
eTripPattern.cDest.Add("Diamond & Canyon");
eTripPattern.cDest.Add("Diamond & Orange/Sandia");
eTripPattern.cDest.Add("Tranquillo & Quemazon");
eTripPattern.cDest.Add("North Rd & Urban");
eTripPattern.cDest.Add("North Rd & Mountain School");
eTripPattern.cDest.Add("35th & Arizona");
eTripPattern.cDest.Add("Diamond & 35th");
eTripPattern.cDest.Add("Diamond & 38th/Arkansas");
eTripPattern.cDest.Add("Diamond & Sycamore");
eTripPattern.cDest.Add("Diamond & Orange/Sandia");
eTripPattern.cDest.Add("Diamond & Canyon");
eTripPattern.cDest.Add("Transit Center");

int[] c3ppArrivalTime = { 26, 28, 29, 31, 33, 34, 37, 37, 38, 39, 41, 42, 44 };
int[] c3ppWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 42 };
eTripPattern.cArrivalTime = c3ppArrivalTime.ToList();
eTripPattern.cWaitTime = c3ppWaitTime.ToList();
eTripPattern.repeatMinutes = 60;
eTripPattern.firstRepeatTime = 15 * 60;
eTripPattern.lastRepeatTime = 19 * 60;
eTripPattern.setcTrip();
route3.cTripPattern.Add(eTripPattern);

#endregion

#region Route4

route4.name = "Route 4";
route4.cTripPattern = new List<TripPattern>();
fTripPattern = new TripPattern();
fTripPattern.cDest = new List<string>();
fTripPattern.name = "Route 4";
fTripPattern.cDest.Add("Transit Center");
fTripPattern.cDest.Add("48th & Trinity");
fTripPattern.cDest.Add("Diamond & Orange/Sandia");
fTripPattern.cDest.Add("Diamond & Sycamore");
fTripPattern.cDest.Add("Diamond & 38th/Arkansas");
fTripPattern.cDest.Add("Diamond & 35th");
fTripPattern.cDest.Add("Diamond & Club");
fTripPattern.cDest.Add("San Ildefonso & Chamisa");
fTripPattern.cDest.Add("Barranca & El Viento");
fTripPattern.cDest.Add("Barranca & E Loma del Escolar");
fTripPattern.cDest.Add("Los Pueblos & Navajo");
fTripPattern.cDest.Add("Barranca & E Loma del Escolar");
fTripPattern.cDest.Add("Barranca & El Viento");
fTripPattern.cDest.Add("San Ildefonso & Chamisa");
fTripPattern.cDest.Add("Diamond & Club");
fTripPattern.cDest.Add("Diamond & 35th");
fTripPattern.cDest.Add("Diamond & 38th/Arkansas");
fTripPattern.cDest.Add("Diamond & Sycamore");
fTripPattern.cDest.Add("Diamond & Orange/Sandia");
fTripPattern.cDest.Add("48th & Trinity");
fTripPattern.cDest.Add("Transit Center");

int[] c4ArrivalTime = { 07, 10, 12, 13, 14, 15, 16, 18, 19, 20, 22, 29, 30, 31, 33, 34, 35, 36, 37, 38, 42 };
```

```
int[] c4WaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 45 };
fTripPattern.cArrivalTime = c4ArrivalTime.ToList();
fTripPattern.cWaitTime = c4WaitTime.ToList();
fTripPattern.repeatMinutes = 60;
fTripPattern.firstRepeatTime = 6 * 60;
fTripPattern.lastRepeatTime = 19 * 60;
fTripPattern.setcTrip();
route4.cTripPattern.Add(fTripPattern);

gTripPattern = new TripPattern();
gTripPattern.cDest = new List<string>();
gTripPattern.name = "Route 4 AM";
gTripPattern.cDest.Add("Transit Center");
gTripPattern.cDest.Add("48th & Trinity");
gTripPattern.cDest.Add("Diamond & Orange/Sandia");
gTripPattern.cDest.Add("Diamond & Sycamore");
gTripPattern.cDest.Add("Diamond & 38th/Arkansas");
gTripPattern.cDest.Add("Diamond & 35th");
gTripPattern.cDest.Add("Diamond & Club");
gTripPattern.cDest.Add("San Ildefonso & Chamisa");
gTripPattern.cDest.Add("Barranca & El Viento");
gTripPattern.cDest.Add("Barranca & E Loma del Escolar");
gTripPattern.cDest.Add("Los Pueblos & Navajo");
gTripPattern.cDest.Add("Barranca & E Loma del Escolar");
gTripPattern.cDest.Add("Barranca & El Viento");
gTripPattern.cDest.Add("San Ildefonso & Chamisa");
gTripPattern.cDest.Add("Diamond & Club");
gTripPattern.cDest.Add("Diamond & 35th");
gTripPattern.cDest.Add("Diamond & 38th/Arkansas");
gTripPattern.cDest.Add("Diamond & Sycamore");
gTripPattern.cDest.Add("Diamond & Orange/Sandia");
gTripPattern.cDest.Add("48th & Trinity");
gTripPattern.cDest.Add("Transit Center");

int[] c4aArrivalTime = { 27, 30, 32, 33, 34, 35, 36, 38, 39, 40, 42, 49, 50, 51, 53, 54, 55, 56, 57, 58, 62 };
int[] c4aWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 45 };
gTripPattern.cArrivalTime = c4aArrivalTime.ToList();
gTripPattern.cWaitTime = c4aWaitTime.ToList();
gTripPattern.repeatMinutes = 60;
gTripPattern.firstRepeatTime = 6 * 60;
gTripPattern.lastRepeatTime = 8 * 60;
gTripPattern.setcTrip();
route4.cTripPattern.Add(gTripPattern);

hTripPattern = new TripPattern();
hTripPattern.cDest = new List<string>();
hTripPattern.name = "Route 4 PM";
hTripPattern.cDest.Add("Transit Center");
hTripPattern.cDest.Add("48th & Trinity");
hTripPattern.cDest.Add("Diamond & Orange/Sandia");
hTripPattern.cDest.Add("Diamond & Sycamore");
hTripPattern.cDest.Add("Diamond & 38th/Arkansas");
hTripPattern.cDest.Add("Diamond & 35th");
hTripPattern.cDest.Add("Diamond & Club");
hTripPattern.cDest.Add("San Ildefonso & Chamisa");
hTripPattern.cDest.Add("Barranca & El Viento");
```

```
hTripPattern.cDest.Add("Barranca & E Loma del Escolar");
hTripPattern.cDest.Add("Los Pueblos & Navajo");
hTripPattern.cDest.Add("Barranca & E Loma del Escolar");
hTripPattern.cDest.Add("Barranca & El Viento");
hTripPattern.cDest.Add("San Ildefonso & Chamisa");
hTripPattern.cDest.Add("Diamond & Club");
hTripPattern.cDest.Add("Diamond & 35th");
hTripPattern.cDest.Add("Diamond & 38th/Arkansas");
hTripPattern.cDest.Add("Diamond & Sycamore");
hTripPattern.cDest.Add("Diamond & Orange/Sandia");
hTripPattern.cDest.Add("48th & Trinity");
hTripPattern.cDest.Add("Transit Center");

int[] c4pArrivalTime = { 47, 50, 52, 53, 54, 55, 56, 58, 59, 60, 62, 69, 70, 71, 73, 74, 75, 76, 77, 78, 82 };
int[] c4pWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 45 };
hTripPattern.cArrivalTime = c4pArrivalTime.ToList();
hTripPattern.cWaitTime = c4pWaitTime.ToList();
hTripPattern.repeatMinutes = 60;
hTripPattern.firstRepeatTime = 15 * 60;
hTripPattern.lastRepeatTime = 18 * 60;
hTripPattern.setcTrip();
route4.cTripPattern.Add(hTripPattern);

#endregion

#region Route5

route5.name = "Route 5";
route5.cTripPattern = new List<TripPattern>();
iTripPattern = new TripPattern();
iTripPattern.cDest = new List<string>();
iTripPattern.name = "Route 5 AM";
iTripPattern.cDest.Add("Transit Center");
iTripPattern.cDest.Add("Sherwood & Aztec");
iTripPattern.cDest.Add("Meadow & Isleta");
iTripPattern.cDest.Add("Grand Canyon & W Paige Lp");
iTripPattern.cDest.Add("Grand Canyon & Aragon");
iTripPattern.cDest.Add("Aragon & Rover");
iTripPattern.cDest.Add("Grand Canyon & Beryl");
iTripPattern.cDest.Add("Grand Canyon & Sherwood");
iTripPattern.cDest.Add("Grand Canyon & La Vista");
iTripPattern.cDest.Add("SR 4 & Karen Circle");
iTripPattern.cDest.Add("SR4 & Monte Rey");
iTripPattern.cDest.Add("Monte Rey S & Portillo");
iTripPattern.cDest.Add("Piedra Lp & Piedra Dr");
iTripPattern.cDest.Add("Piedra Lp & La Senda");
iTripPattern.cDest.Add("SR 4 & Karen Circle");
iTripPattern.cDest.Add("Grand Canyon & La Vista");
iTripPattern.cDest.Add("Grand Canyon & Sherwood");
iTripPattern.cDest.Add("Grand Canyon & Beryl");
iTripPattern.cDest.Add("Aragon & Rover");
iTripPattern.cDest.Add("Grand Canyon & Aragon");
iTripPattern.cDest.Add("Grand Canyon & W Paige Lp");
iTripPattern.cDest.Add("Meadow & Isleta");
iTripPattern.cDest.Add("Sherwood & Aztec");
iTripPattern.cDest.Add("Transit Center");
```

```
        int[] c5ArrivalTime = { 07, 19, 22, 24, 25, 26, 27, 28, 29, 30, 31, 46, 51, 56, 57, 58, 59, 60, 61, 62, 63, 65,
68, 81 };
        int[] c5WaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 45 };
        iTripPattern.cArrivalTime = c5ArrivalTime.ToList();
        iTripPattern.cWaitTime = c5WaitTime.ToList();
        iTripPattern.repeatMinutes = 60;
        iTripPattern.firstRepeatTime = 6 * 60;
        iTripPattern.lastRepeatTime = 11 * 60;
        iTripPattern.setcTrip();
        route5.cTripPattern.Add(iTripPattern);

        jTripPattern = new TripPattern();
        jTripPattern.cDest = new List<string>();
        jTripPattern.name = "Route 5 AM";
        jTripPattern.cDest.Add("Transit Center");
        jTripPattern.cDest.Add("Sherwood & Aztec");
        jTripPattern.cDest.Add("Meadow & Isleta");
        jTripPattern.cDest.Add("Grand Canyon & W Paige Lp");
        jTripPattern.cDest.Add("Grand Canyon & Aragon");
        jTripPattern.cDest.Add("Aragon & Rover");
        jTripPattern.cDest.Add("Grand Canyon & Beryl");
        jTripPattern.cDest.Add("Grand Canyon & Sherwood");
        jTripPattern.cDest.Add("Grand Canyon & La Vista");
        jTripPattern.cDest.Add("SR 4 & Karen Circle");
        jTripPattern.cDest.Add("SR4 & Monte Rey");
        jTripPattern.cDest.Add("Monte Rey S & Portillo");
        jTripPattern.cDest.Add("Piedra Lp & Piedra Dr");
        jTripPattern.cDest.Add("Piedra Lp & La Senda");
        jTripPattern.cDest.Add("SR 4 & Karen Circle");
        jTripPattern.cDest.Add("Grand Canyon & La Vista");
        jTripPattern.cDest.Add("Grand Canyon & Sherwood");
        jTripPattern.cDest.Add("Grand Canyon & Beryl");
        jTripPattern.cDest.Add("Aragon & Rover");
        jTripPattern.cDest.Add("Grand Canyon & Aragon");
        jTripPattern.cDest.Add("Grand Canyon & W Paige Lp");
        jTripPattern.cDest.Add("Meadow & Isleta");
        jTripPattern.cDest.Add("Sherwood & Aztec");
        jTripPattern.cDest.Add("Transit Center");


        int[] c5aArrivalTime = { 27, 39, 42, 44, 45, 46, 47, 48, 49, 50, 51, 66, 71, 76, 77, 78, 79, 80, 81, 82, 83, 85,
88, 101 };
        int[] c5aWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 45 };
        jTripPattern.cArrivalTime = c5aArrivalTime.ToList();
        jTripPattern.cWaitTime = c5aWaitTime.ToList();
        jTripPattern.repeatMinutes = 60;
        jTripPattern.firstRepeatTime = 6 * 60;
        jTripPattern.lastRepeatTime = 7 * 60;
        jTripPattern.setcTrip();
        route5.cTripPattern.Add(jTripPattern);

        kTripPattern = new TripPattern();
        kTripPattern.cDest = new List<string>();
        kTripPattern.name = "Route 5 PM";
```

```
        kTripPattern.cDest.Add("Transit Center");
        kTripPattern.cDest.Add("Sherwood & Aztec");
        kTripPattern.cDest.Add("Meadow & Isleta");
        kTripPattern.cDest.Add("Grand Canyon & W Paige Lp");
        kTripPattern.cDest.Add("Grand Canyon & Aragon");
        kTripPattern.cDest.Add("Aragon & Rover");
        kTripPattern.cDest.Add("Grand Canyon & Beryl");
        kTripPattern.cDest.Add("Grand Canyon & Sherwood");
        kTripPattern.cDest.Add("Grand Canyon & La Vista");
        kTripPattern.cDest.Add("SR 4 & Karen Circle");
        kTripPattern.cDest.Add("SR4 & Monte Rey");
        kTripPattern.cDest.Add("Monte Rey S & Portillo");
        kTripPattern.cDest.Add("Piedra Lp & Piedra Dr");
        kTripPattern.cDest.Add("Piedra Lp & La Senda");
        kTripPattern.cDest.Add("SR 4 & Karen Circle");
        kTripPattern.cDest.Add("Grand Canyon & La Vista");
        kTripPattern.cDest.Add("Grand Canyon & Sherwood");
        kTripPattern.cDest.Add("Grand Canyon & Beryl");
        kTripPattern.cDest.Add("Aragon & Rover");
        kTripPattern.cDest.Add("Grand Canyon & Aragon");
        kTripPattern.cDest.Add("Grand Canyon & W Paige Lp");
        kTripPattern.cDest.Add("Meadow & Isleta");
        kTripPattern.cDest.Add("Sherwood & Aztec");
        kTripPattern.cDest.Add("Transit Center");


        int[] c5pArrivalTime = { 07, 19, 22, 24, 25, 26, 27, 28, 29, 30, 31, 46, 51, 56, 57, 58, 59, 60, 61, 62, 63, 65,
68, 81 };
        int[] c5pWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 0, 0, 0, 0, 0, 0, 0, 0, 0, 45 };
        kTripPattern.cArrivalTime = c5pArrivalTime.ToList();
        kTripPattern.cWaitTime = c5pWaitTime.ToList();
        kTripPattern.repeatMinutes = 60;
        kTripPattern.firstRepeatTime = 12 * 60;
        kTripPattern.lastRepeatTime = 19 * 60;
        kTripPattern.setcTrip();
        route5.cTripPattern.Add(kTripPattern);

        lTripPattern = new TripPattern();
        lTripPattern.cDest = new List<string>();
        lTripPattern.name = "Route 5 PM";
        lTripPattern.cDest.Add("Transit Center");
        lTripPattern.cDest.Add("Sherwood & Aztec");
        lTripPattern.cDest.Add("Meadow & Isleta");
        lTripPattern.cDest.Add("Grand Canyon & W Paige Lp");
        lTripPattern.cDest.Add("Grand Canyon & Aragon");
        lTripPattern.cDest.Add("Aragon & Rover");
        lTripPattern.cDest.Add("Grand Canyon & Beryl");
        lTripPattern.cDest.Add("Grand Canyon & Sherwood");
        lTripPattern.cDest.Add("Grand Canyon & La Vista");
        lTripPattern.cDest.Add("SR 4 & Karen Circle");
        lTripPattern.cDest.Add("SR4 & Monte Rey");
        lTripPattern.cDest.Add("Monte Rey S & Portillo");
        lTripPattern.cDest.Add("Piedra Lp & Piedra Dr");
        lTripPattern.cDest.Add("Piedra Lp & La Senda");
        lTripPattern.cDest.Add("SR 4 & Karen Circle");
        lTripPattern.cDest.Add("Grand Canyon & La Vista");
```

```
        lTripPattern.cDest.Add("Grand Canyon & Sherwood");
        lTripPattern.cDest.Add("Grand Canyon & Beryl");
        lTripPattern.cDest.Add("Aragon & Rover");
        lTripPattern.cDest.Add("Grand Canyon & Aragon");
        lTripPattern.cDest.Add("Grand Canyon & W Paige Lp");
        lTripPattern.cDest.Add("Meadow & Isleta");
        lTripPattern.cDest.Add("Sherwood & Aztec");
        lTripPattern.cDest.Add("Transit Center");


        int[] c5ppArrivalTime = { 47, 59, 62, 64, 65, 66, 67, 68, 69, 70, 71, 86, 91, 96, 97, 98, 99, 100, 101, 102,
103, 105, 108, 121 };
        int[] c5ppWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 0, 0, 0, 0, 0, 0, 0, 0, 0, 45 };
        lTripPattern.cArrivalTime = c5ppArrivalTime.ToList();
        lTripPattern.cWaitTime = c5ppWaitTime.ToList();
        lTripPattern.repeatMinutes = 60;
        lTripPattern.firstRepeatTime = 15 * 60;
        lTripPattern.lastRepeatTime = 18 * 60;
        lTripPattern.setcTrip();
        route5.cTripPattern.Add(lTripPattern);

        #endregion

        #region Route6

        route6.name = "Route 6";
        route6.cTripPattern = new List<TripPattern>();
        mTripPattern = new TripPattern();
        mTripPattern.cDest = new List<string>();
        mTripPattern.name = "Route 6";
        mTripPattern.cDest.Add("Transit Center");
        mTripPattern.cDest.Add("Diamond & Canyon");
        mTripPattern.cDest.Add("Diamond & Orange/Sandia");
        mTripPattern.cDest.Add("Diamond & Sycamore");
        mTripPattern.cDest.Add("Diamond & 38th/Arkansas");
        mTripPattern.cDest.Add("35th & Villa");
        mTripPattern.cDest.Add("Diamond & 35th");
        mTripPattern.cDest.Add("Diamond & Club");
        mTripPattern.cDest.Add("Hawk Dr & San Ildefonso");
        mTripPattern.cDest.Add("San Ildefonso & Sioux");
        mTripPattern.cDest.Add("Big Rock Lp & Stoneview");
        mTripPattern.cDest.Add("San Ildefonso & Sioux");
        mTripPattern.cDest.Add("Hawk Dr & San Ildefonso");
        mTripPattern.cDest.Add("Diamond & Club");
        mTripPattern.cDest.Add("Diamond & 35th");
        mTripPattern.cDest.Add("35th & Villa");
        mTripPattern.cDest.Add("Diamond & 38th/Arkansas");
        mTripPattern.cDest.Add("Diamond & Sycamore");
        mTripPattern.cDest.Add("Diamond & Orange/Sandia");
        mTripPattern.cDest.Add("Diamond & Canyon");
        mTripPattern.cDest.Add("Transit Center");


        int[] c6ArrivalTime = { 27, 29, 30, 31, 32, 33, 34, 35, 36, 38, 40, 50, 52, 54, 55, 56, 57, 58, 59, 60, 63 };
        int[] c6WaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 24 };
        mTripPattern.cArrivalTime = c6ArrivalTime.ToList();
```

```
mTripPattern.cWaitTime = c6WaitTime.ToList();
mTripPattern.repeatMinutes = 60;
mTripPattern.firstRepeatTime = 5 * 60;
mTripPattern.lastRepeatTime = 18 * 60;
mTripPattern.setcTrip();
route6.cTripPattern.Add(mTripPattern);

nTripPattern = new TripPattern();
nTripPattern.cDest = new List<string>();
nTripPattern.name = "Route 6 AM";
nTripPattern.cDest.Add("Transit Center");
nTripPattern.cDest.Add("Diamond & Canyon");
nTripPattern.cDest.Add("Diamond & Orange/Sandia");
nTripPattern.cDest.Add("Diamond & Sycamore");
nTripPattern.cDest.Add("Diamond & 38th/Arkansas");
nTripPattern.cDest.Add("35th & Villa");
nTripPattern.cDest.Add("Diamond & 35th");
nTripPattern.cDest.Add("Diamond & Club");
nTripPattern.cDest.Add("Hawk Dr & San Ildefonso");
nTripPattern.cDest.Add("San Ildefonso & Sioux");
nTripPattern.cDest.Add("Big Rock Lp & Stoneview");
nTripPattern.cDest.Add("San Ildefonso & Sioux");
nTripPattern.cDest.Add("Hawk Dr & San Ildefonso");
nTripPattern.cDest.Add("Diamond & Club");
nTripPattern.cDest.Add("Diamond & 35th");
nTripPattern.cDest.Add("35th & Villa");
nTripPattern.cDest.Add("Diamond & 38th/Arkansas");
nTripPattern.cDest.Add("Diamond & Sycamore");
nTripPattern.cDest.Add("Diamond & Orange/Sandia");
nTripPattern.cDest.Add("Diamond & Canyon");
nTripPattern.cDest.Add("Transit Center");


int[] c6aArrivalTime = { 47, 49, 50, 51, 52, 53, 54, 55, 56, 58, 60, 70, 72, 74, 75, 76, 77, 78, 79, 80, 83 };
int[] c6aWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 24 };
nTripPattern.cArrivalTime = c6aArrivalTime.ToList();
nTripPattern.cWaitTime = c6aWaitTime.ToList();
nTripPattern.repeatMinutes = 60;
nTripPattern.firstRepeatTime = 5 * 60;
nTripPattern.lastRepeatTime = 7 * 60;
nTripPattern.setcTrip();
route6.cTripPattern.Add(nTripPattern);

oTripPattern = new TripPattern();
oTripPattern.cDest = new List<string>();
oTripPattern.name = "Route 6 PM";
oTripPattern.cDest.Add("Transit Center");
oTripPattern.cDest.Add("Diamond & Canyon");
oTripPattern.cDest.Add("Diamond & Orange/Sandia");
oTripPattern.cDest.Add("Diamond & Sycamore");
oTripPattern.cDest.Add("Diamond & 38th/Arkansas");
oTripPattern.cDest.Add("35th & Villa");
oTripPattern.cDest.Add("Diamond & 35th");
oTripPattern.cDest.Add("Diamond & Club");
oTripPattern.cDest.Add("Hawk Dr & San Ildefonso");
oTripPattern.cDest.Add("San Ildefonso & Sioux");
```

```
    oTripPattern.cDest.Add("Big Rock Lp & Stoneview");
    oTripPattern.cDest.Add("San Ildefonso & Sioux");
    oTripPattern.cDest.Add("Hawk Dr & San Ildefonso");
    oTripPattern.cDest.Add("Diamond & Club");
    oTripPattern.cDest.Add("Diamond & 35th");
    oTripPattern.cDest.Add("35th & Villa");
    oTripPattern.cDest.Add("Diamond & 38th/Arkansas");
    oTripPattern.cDest.Add("Diamond & Sycamore");
    oTripPattern.cDest.Add("Diamond & Orange/Sandia");
    oTripPattern.cDest.Add("Diamond & Canyon");
    oTripPattern.cDest.Add("Transit Center");


    int[] c6pArrivalTime = { 07, 09, 10, 11, 12, 13, 14, 15, 16, 18, 20, 30, 32, 34, 35, 36, 37, 38, 39, 40, 43 };
    int[] c6pWaitTime = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 24 };
    oTripPattern.cArrivalTime = c6pArrivalTime.ToList();
    oTripPattern.cWaitTime = c6pWaitTime.ToList();
    oTripPattern.repeatMinutes = 60;
    oTripPattern.firstRepeatTime = 15 * 60;
    oTripPattern.lastRepeatTime = 19 * 60;
    oTripPattern.setcTrip();
    route6.cTripPattern.Add(oTripPattern);

    #endregion


    allRoutes.Add(route1);
    allRoutes.Add(route3);
    allRoutes.Add(route4);
    allRoutes.Add(route5);
    allRoutes.Add(route6);
    foreach (Route r in allRoutes)
    {
        foreach (TripPattern tp in r.cTripPattern)
        {
            cDest = cDest.Union(tp.cDest).ToList<string>();
        }
        r.setcTrip();
    }

    listBoxStart.Items.AddRange(cDest.ToArray<string>());
    listBoxEnd.Items.AddRange(cDest.ToArray<string>());

}



public string startDest, endDest;
public int hour, minute;

public string clockTime(int min)
{
    var hour = (min / 60).ToString();
    var minutes = (min % 60);
    string digits;
    if (minutes < 10)
```

```
    {
        digits = "0" + minutes.ToString();
    }
    else
    {
        digits = minutes.ToString();
    }
    var clockTime = hour + ":" + digits;
    return clockTime;
}

public string routeDestTime(Route r, string d, int min)
{
    var routeName = r.name;
    var destName = d;
    var time = this.clockTime(min);
    return time + " " + destName + " " + routeName;
}

private void button1_Click_1(object sender, EventArgs e)
{

    if (listBoxEnd.Text == listBoxStart.Text)
    {
        MessageBox.Show("Stay where you are.");
        return;
    }

    startDest = listBoxStart.Text;
    endDest = listBoxEnd.Text;
    hour = int.Parse(listBoxHour.Text);
    minute = int.Parse(listBoxMinute.Text);
    var endRoutes =
        from r in allRoutes
        where r.cDest().Contains(endDest)
        select r;
    var startRoutes =
        from r in allRoutes
        where r.cDest().Contains(startDest)
        select r;

    var goodRoutes = startRoutes.Intersect(endRoutes);
    var aArrivalTime = int.Parse(listBoxHour.Text) * 60 + int.Parse(listBoxMinute.Text);
    var goodTrips = new List<Trip>();
    foreach (Route r in goodRoutes)
    {
        var startandEndTime = r.startandEndTime(startDest, endDest, aArrivalTime);
        var startTime = startandEndTime[0];
        var endTime = startandEndTime[1];
        textBox1.AppendText("Start Time is " + this.routeDestTime(r, startDest, startTime));
        textBox1.AppendText("\r\nEnd Time is " + this.routeDestTime(r, endDest, endTime));
    }

    if (goodRoutes.Count() == 0)
    {
        foreach (Route rs in startRoutes)
```

```
            {
                foreach (Route re in endRoutes)
                {
                    var changeDest = re.joinStartandEnd(rs, startDest, endDest);
                    var startandEndTime = re.startandEndTime(changeDest, endDest, aArrivalTime);
                    var getOnTime = startandEndTime[0];
                    var arriveTime = startandEndTime[1];
                    startandEndTime = rs.startandEndTime(startDest, changeDest, getOnTime);
                    var startTime = startandEndTime[0];
                    var getOffTime = startandEndTime[1];
                    textBox1.AppendText("\r\nStart Time is " + this.routeDestTime(rs, startDest, startTime));
                    textBox1.AppendText("\r\nGet off Time " + this.routeDestTime(rs, changeDest, getOffTime));
                    textBox1.AppendText("\r\nSwitch Time " + this.routeDestTime(re, changeDest, getOnTime));
                    textBox1.AppendText("\r\nEnd Time is " + this.routeDestTime(re, endDest, arriveTime));
                    textBox1.AppendText("\r\n");
                }
            }

        }
    }

    private void button2_Click_1(object sender, EventArgs e)
    {
        listBoxEnd.Text = null;
        listBoxHour.Text = null;
        listBoxMinute.Text = null;
        listBoxStart.Text = null;
        textBox1.Text = null;
    }
  }
}
```

Route Class:

```
using System;

using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace test6
{
   public class Route
   {

       public string name;
       public List<TripPattern> cTripPattern;
       public List<Trip> cTrip;
       public List<Trip> setcTrip()
       {
           cTrip = new List<Trip>();
           foreach (TripPattern tp in cTripPattern)
           {
               cTrip.AddRange(tp.cTrip);
           }
           return cTrip;
```

```
    }
    public List<string> cDest()
    {
        return cTripPattern.First<TripPattern>().cDest;
    }

    public List<int> cDestIndex(string aDest)
    {
        var cDestIndex = new List<int>();
        var cDest = this.cDest();
        for (int i = 0; i < cDest.Count; i++)
        {
            if (aDest == cDest[i])
            {
                cDestIndex.Add(i);
            }
        }
        return cDestIndex;
    }

    public List<Trip> cTripNearestTime(string aDest, int time)
    {
        var cTripNearestTime = new List<Trip>();
        var cDestIndex = this.cDestIndex(aDest);
        for (int i = 0; i < cTrip.Count; i++)
        {
            var aTrip = cTrip[i];
            if (cDestIndex.Any(delegate (int index) { return aTrip.cArrivalTime[index] < time;}))
            {
                cTripNearestTime.Add(aTrip);
            }
        }
        int j = cDestIndex.First();
        cTripNearestTime.Sort(delegate(Trip t1, Trip t2) { return
t1.cArrivalTime[j].CompareTo(t2.cArrivalTime[j]); });
        return cTripNearestTime;
    }

    public List<int> startandEndTime(string startDest, string endDest, int aArrivalTime)
    {
        var cTripNearestTime = this.cTripNearestTime(endDest, aArrivalTime);
        var bestTrip = cTripNearestTime.Last().Copy();
        var continueBestTrip = cTripNearestTime[cTripNearestTime.Count - 2].Copy();
        var secondBestTrip = cTripNearestTime[cTripNearestTime.Count - 3].Copy();
        if (bestTrip.overlaps(continueBestTrip))
        {
            secondBestTrip = continueBestTrip;
            continueBestTrip = cTripNearestTime[cTripNearestTime.Count - 3].Copy();
        }
        continueBestTrip.Append(bestTrip);
        var bestStartandEndTime = continueBestTrip.startandEndTime(startDest, endDest, aArrivalTime);
        var secondBestStartEndTime = secondBestTrip.startandEndTime(startDest, endDest, aArrivalTime);
        bestStartandEndTime.AddRange(secondBestStartEndTime);
        return bestStartandEndTime;
    }
```

```
        public string joinStartandEnd(Route startRoute, string startDest, string endDest)
        {
            var csDest = new List<string>();
            var ceDest = new List<string>();
            csDest.AddRange(startRoute.cDest());
            ceDest.AddRange(this.cDest());
            var commonDest = csDest.Intersect(ceDest);
            var endIndex = -1;
            var cEndDest = this.cDest();
            for (int i = cEndDest.Count - 1; endIndex < 0; i--)
            {
                if (cEndDest[i] == endDest)
                {
                    endIndex = i;
                }
            }
            var startIndex = -1;
            for (int i = endIndex - 1; (startIndex < 0) & (i >= 0); i--)
            {
                if (cEndDest[i] == endDest)
                {
                    endIndex = i;
                }
                if (commonDest.Contains(cEndDest[i]))
                {
                    startIndex = i;
                }
            }
            return cEndDest[startIndex];
        }
    }
}
```

Trip Class:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace test6
{
    public class Trip
    {
        public string name;
        public List<string> cDest;
        public List<int> cArrivalTime;
        public List<int> cWaitTime;
        public Boolean isAtDestNearTime(string aDest, int aTime, int aTolerance)
        {
            for (int i = 0; i < cDest.Count; i++)
            {
                if (cDest[i] == aDest)
```

```
        {
            var time = cArrivalTime[i];
            if ((time < aTime) & (time >= aTime - aTolerance))
            {
                return true;
            }
        }
    }
    return false;
}

public Boolean overlaps(Trip bTrip)
{
    var lastTime = cArrivalTime.Last();
    var firstTime = cArrivalTime.First();
    if ((lastTime > bTrip.cArrivalTime.First()) & (lastTime < bTrip.cArrivalTime.Last()))
    {
        return true;
    }
    if ((firstTime > bTrip.cArrivalTime.First()) & (firstTime < bTrip.cArrivalTime.Last()))
    {
        return true;
    }
    return false;
}

public List<int> cTime(string aDest)
{
    var cTimeDest = new List<int>();

    for (int i = 0; i < cDest.Count; i++)
    {
        if (cDest[i] == aDest)
        {
            cTimeDest.Add(cArrivalTime[i]);
        }
    }

        return cTimeDest;
}

public Trip Copy()
{
    var answer = new Trip();
    answer.cDest = new List<string>();
    answer.cDest.AddRange(cDest);
    answer.cArrivalTime = new List<int>();
    answer.cArrivalTime.AddRange(cArrivalTime);
    answer.cWaitTime = new List<int>();
    //answer.cWaitTime.AddRange(cWaitTime);
    return answer;
}

public Trip Append(Trip aTrip)
{
    cDest.AddRange(aTrip.cDest);
```

```
        cArrivalTime.AddRange(aTrip.cArrivalTime);
        //cWaitTime.AddRange(aTrip.cWaitTime);
        return this;
    }
    public List<int> startandEndTime(string startDest, string endDest, int aArrivalTime)
    {
        var endIndex = -1;
        for (int i = cArrivalTime.Count - 1; endIndex < 0; i--)
        {
            if ((cDest[i] == endDest) & (cArrivalTime[i] < aArrivalTime))
            {
                endIndex = i;
            }
        }
        var startIndex = -1;
        for (int i = endIndex - 1; (startIndex < 0) & (i >= 0); i--)
        {
            if (cDest[i] == endDest)
            {
                endIndex = i;
            }
            if ((cDest[i] == startDest) & (cArrivalTime[i] < cArrivalTime[endIndex]))
            {
                startIndex = i;
            }
        }
        var startTime = cArrivalTime[startIndex];
        var endTime = cArrivalTime[endIndex];
        var startandEndTime = new List<int>();
        startandEndTime.Add(startTime);
        startandEndTime.Add(endTime);
        return startandEndTime;
    }

}
```

TripPattern Class:

```
public class TripPattern : Trip
{
    public int repeatMinutes;
    public int firstRepeatTime;
    public int lastRepeatTime;
    public List<Trip> cTrip;
    public List<Trip> setcTrip()
    {
        cTrip = new List<Trip>();
        for (int time = firstRepeatTime; time <= lastRepeatTime; time=time+repeatMinutes)
        {
            var aTrip = new Trip();
            var cTime = new List<int>();
            foreach (int aArrivalTime in cArrivalTime) { cTime.Add(time + aArrivalTime); }
            aTrip.cArrivalTime = cTime;
            aTrip.cDest = cDest;
            aTrip.name = name;
```

```
            cTrip.Add(aTrip);
        }
        return cTrip;
    }

  }
}
```