## Windwardopolis

Welcome to the booming city of Windwardopolis. The largest high-tech companies all have corporate headquarters here in Windwardopolis. You own a limousine service with one limo (yes it’s a small operation, but a proud one).

These CEOs need to travel to the other corporate headquarters. Your job is to provide them the transportation from one location to the next. And to do so with a smile – no one likes an unhappy driver.

## The Game

There are 12 CEOs, each at their own office. Each CEO wants to visit the other 11 offices, in a specific order (their route). Your job is to transport 8 of the CEOs on one leg of their trips.

In addition, each CEO has a list of 1 – 6 enemies. A CEO refuses to disembark the limo at a bus stop that one of their enemies is at. The enemies list of the CEOs presently at a bus stop do not stop a CEO from disembarking (the enemy check is one-way).

* Example: Larry Page has Tim Cook as an enemy. But Tim Cook does not have Larry Page as an enemy. So Larry Page will not exit a limo at an office where Tim Cook is waiting. But Tim Cook will exit a limo at an office where Larry Page is waiting.

The challenges you face are:

1. You need to arrive at a bus stop (each office has a bus stop) when the passenger is still there. Another car may get there first and then you don’t have anyone to transport.
2. You have a passenger you want to drop off and one of their enemies is at the drop off bus stop – which stops you from dropping them there. In this case you can abandon them at a different bus stop, but that does not count as a completed trip.
3. You want to find the shortest trips, near where you are. The points are identical for completing a trip that is 5 squares long and one that is 60 squares long.
4. You have to find the best route on the map. Drive times are heavily influenced by what the other cars are doing.

## Game Setup

1. The map is randomly selected at the start of each game.
2. The companies are randomly assigned to the bus-stop locations.
3. Each CEO starts at their company but the order of their destinations to the other 11 companies are randomly assigned each game.
4. The enemies of each CEO are randomly assigned each game.

## What Information you are provided

1. At game start you are given the map, list of players, list of companies, and list of passengers.
2. The game is “real-time” but in practice that means everything moves once per tick. You will get a status update every 16 ticks. This will provide the position and passenger of every car, the status of every passenger (where they are, their next destination), and your car’s path and list of passengers to pick up.
3. When anyone arrives at any bus stop all players get a status listing if the passenger was dropped off and if one was picked up.
4. When your car’s path is used up or bad you will receive a status message.

## Sending Orders to the Game

1. You can send orders at any time. These orders may contain:
   1. A new path
   2. A new list of passengers to pick up, in preferred order, at the next bus stop.

The path and pickup are the only orders you can send. When your car arrives at a bus stop, if the passenger in your car can disembark, they will. If your car is empty and a passenger in your pickup list can embark, they will. There is no option in your orders to override this.

If the orders you send has no path, it will not update the path (just the pickup list). If the pickup list is empty it will not update the pickup list (just the path).

* Note: The game is running when messages are sent to your A.I. and when you’re A.I. sends orders to the server. The messages are asynchronous and the server does not wait for a reply.

## Points

1. The game will be played in 10 rounds with the score from all 10 rounds added up for a final score.
2. There are 12 passengers who need to go from one company to another.
   1. Each passenger has a series of destinations so after arriving at one, they then want to go on to another.
   2. Each passenger starts at one office and has the other 11, in a random order, to visit.
   3. If a passenger has visited all 12 offices, they can no longer be transported.
3. A passenger is in a car or waiting at a bus stop (or if they’ve had 11 trips – done).
4. There are 9 other cars (you have the tenth) so there are always at least 3 passengers somewhere available for you to pick up.
5. The round ends when one car has transported 8 passengers to their destination.
   1. All players receive:
      1. 1 – 3 points for each complete delivery. (Each passenger has their point value as a property. Shirley Clawson as CEO of Windward has 3 points, the CEOs of all contest sponsors have 2 points, and the rest are 1 point).
      2. If a delivery is in process, the percentage complete of the trip measured by the Manhattan distance times 0.5 will be added to the score. This part will be 0 if the car is further away than the pick-up bus stop when the round ends.
   2. The player who delivered 8 passengers receives an additional 2 points.

## Passengers

1. You can only successfully transport a specific passenger once (if your trip is incomplete, you can pick them up again later).
2. You can only have 1 (or 0) passenger in your car.
3. Every passenger refuses to go to a bus stop that has any of their enemies.
   1. Each Passenger will have 2 – 5 enemies (randomly assigned at the start of the game).
   2. If their enemy is at their destination, you either have to wait for the enemy to be picked up or drop them at another company (bus stop). If you abandon them elsewhere, that trip does not count.
   3. You may drop off a passenger if someone at that bus stop has them listed as an enemy (enemy lists are not reciprocal). It’s just your passenger’s setting that limits this.
4. You may drop a passenger off at the wrong location.
   1. Anyone else (or you) can then pick that passenger up, complete their trip, and get credit for that trip.
   2. You still can carry that passenger later for a successful trip.
5. When your car goes to a bus stop, it will drop off the passenger you have, unless either of their enemies are there.
6. When your car goes to a bus stop, if it has room (no passenger or passenger could be dropped), it will pick up 1 passenger from your list.
   1. You cannot pick up the passenger you dropped off there. You can go around the block and load (or more likely, transport someone else and then return).

## Vehicle paths

1. The path is defined by tile units, not by map units. The various units are described in the map section below but for the A.I., everything is in tile units.
2. The engine will handle direction changes required by a new path. If your car is Northbound and the path says go southbound, it will make a U-turn as soon as possible:
   1. Turns can only occur upon entrance to a straight map tile. So the car may have to travel in its present direction for several tiles before a U-turn can be made.
   2. Same issue for a change requiring a right/left turn in a tile when the car has already entered the tile. It has to continue out of that tile and then U-turn, and then right/left turn.
   3. If you provide a new path where the turn will occur after the map tile the car is presently in, then it will happen on that first tile (unless it is a curve).
3. If you provide no or a bad path, your car will move randomly.
   1. If you hit a bus stop it will drop off the passenger (if allowed) and will pick one up if one is there in your pickup list.
4. When provided a new path, because the system is real-time and your vehicle may be beyond where it was when you calculated the path, it will:
   1. Walk the new path looking for a map tile matching the tile the car is presently on. If it finds that, it will append the new path at that point.
   2. If no match, it will walk the old path, looking for a map tile where it can join to the new path, looking the length of the new path too. If it finds that, it will append the new path at that point.
   3. If no match, the engine will post a message to your A.I. saying it could not use your new path. It will then continue on the old path. If the old path runs out, it will then move randomly.
5. If your path is not continuous or has illegal steps (park or office squares), the remaining path will be discarded and the engine will post a message to your A.I. saying it ran out of path.

## Maps

1. We created 8 maps and rated them on a scale of 1 (easiest) to 8. Maps 2, 4, & 6 are not in the sample. Map 4 will be added for the school finals and maps 2 & 6 will be added for the international final. This is to stop you from hard-coding solutions for a known set of maps.
2. For the school (and international) finals, the 5 sample maps will be slightly different from the ones used for the school semi-final. These differences are minor but enough to eliminate the possibility of pre-calculating paths. For example every bus stop will be moved a square or two, roads will be slightly shifted and/or have different curves.
3. There are tile units and map units. The game moves cars using map units and there are 24 map units per tile. **The API for the A.I. is based solely on tile units** and the only reason you need to be aware of map units is that once a car is moving through a tile, that is locked until it exits the tile.
   1. Collision avoidance is implemented by dividing each tile into 4 sub-tiles and tracking car movement in those sub-tiles. You probably do not need to know this.

## Movement

1. Drivers in Windwardopolis are very safe. They leave 1 car length between cars. They always come to a full stop at stop signs and yellow & red lights.
2. Cars decelerate & stop instantly (great brakes!)
3. Cars require time to speed up. Each turn they will increase their speed by 0.1 up to the maximum.
   1. The maximum is 6 map units/turn on straight-aways and 3 map units/turn in curves.
4. When a car reaches a yellow or red light (front bumper of car will enter the intersection on this tick):
   1. If there is no car in the intersection, the light will turn green and they will continue at full speed.
   2. If another car is in the intersection, your car stops instantly and the other direction turns yellow. All cars in the intersection will continue but no cars will enter the intersection.
   3. Once there is no car in the intersection, it turns green and your car accelerates.
5. When a car reaches a stop sign.
   1. It will stop even if everything is clear.
   2. It will then go only if no car is within 42 map units of the intersection (from the no stop directions). This is adequate distance for crossing straight, more than needed for a right turn, and less than needed for a left turn (which can then slow the car it moves in front of).
   3. It will wait for a right of way car to clear the intersection even for a right turn when the other car is coming from the right.
6. Cars wait on others even when moving is safe. For example cars in all 4 directions can make a right turn at the same time. But in Windwardopolis they will not. A car in the intersection is handled the same regardless of curve.
7. A car wishing to make a left turn always yields to oncoming cars. It will not enter the intersection until it is clear from all other directions.
   1. If opposing cars both wish to make a left turn, the one to the intersection first will turn and the other will wait.
8. If two cars block each other (they can jammed up if several are trying to get in 1 bus stop at the same time), after 20 ticks the cars will be moved even if they end up on top if each other.

# Programming Environment

The provided code has the server (C# application running on Windows) and 4 sample clients (C#, C++, Java, & Python). You will use the server and one client.

You can run the server and client both on a single computer or the server on one and the client on a second.

* Disable the firewall on your computers. If running both on a single computer, you need to disable when running for the school semi-final.
* If running on 2 computers, the clients will connect to the provided IP address. So you can run “PlayerCSharpAI 123.456.789.000” and it will connect to the server at that IP address. If you do not pass an IP address it uses 127.0.0.1 (the local machine).

Start with the sample AI program and revise that for your AI. You generally will only need to edit the SimpleAStar & MyPlayerBrain classes, plus any additional classes you create. The other classes in the sample you should be able to use as is.

The sample AI and A\* code is bad. Really, really bad. (It was difficult to write such a poor A\*). Do **not** depend on any of that code to be effective.

## Programming Environment

* C# & C++ - Microsoft Visual Studio 2010 or 2012
* Java – JDK 1.6 or 1.7
* Python – 2.7.3
* Server component – Windows 7 or 8. If running under Visual Studio – 2010 or 2012

# Our Hints

1. [Turn off every distraction](http://blogs.windward.net/davidt/2012/04/22/want-to-be-a-better-developer-cheat/).
2. Random numbers are your friend. If your calculation scores moving passenger A as a 95 (scale 1 – 100) and passenger B an 89, don’t always take passenger A. Instead get a random number between 0 and 184 (89 + 95) and select based on the returned value.
3. Set the server logging level to INFO and your client to DEBUG. A lot of the time you will need to review the logs to find out what went wrong. By the time you see the problem, the game has moved well beyond that point and so the debugger is useless.

## Using the web

You can look on the web for information. And anything you can find you can use. So if you find a better A\* on the web, all coded up, you can use that. What you may not do is ask anyone for help. So no posting questions, emailing friends, etc.

# The Server Program

The maps used are set in windwardopolis.exe.config. You can change this to be a specific map for your testing.

## Running the server

Run windwardopolis.exe.

* Make sure the files windwardopolis.exe.config and log4net.dll are in the same directory.

There are two command line modes for the server. Normally you will run it as a Forms app and use the menu. However you may find these modes useful.

windwardopolis.exe /t “Anders Hejlsberg”

The /t stands for test  
The server will run, wait for your client to start, send the setup command to your client, wait for the ready reply, then send an exit to your client and exit with a code of 0.   
This is used by our game server to test clients when they are uploaded for the final.  
Replace “Anders Hejlsberg” with your team name.

windwardopolis.exe /a 3 c:\temp\game.xml "Anders Hejlsberg"

The /a means automatic.   
3 means run 3 games.   
File is where the scores will be written.  
Then list 1 - 10 names of the players you want run.  
When the client AIs listed have all conneected to the server, it will then run at full speed, for the specified number of games, write the scores to the specified XML file, tell the client(s) to exit, and then exit with a return code of 0.   
This is used to run all games in the world-wide final.

## Running under the debugger

If you wish to run the server component under the debugger (to break and step through the engine), load this program in Visual Studio by going to File, Open, Project/Solution… and selecting the file Windwardopolis.sln.

* Everything interesting occurs within Windwardopolis.game\_engine.Engine.Tick().
* On each tick the vehicles are moved 0 or 1 map unit (unless un-sticking vehicles where it can be several ticks). The number of ticks to move across a tile unit depends on the movement (straight, inner turn, outer turn, or U-turn).
* If you change a limo location, it must be on a valid map unit following a TileMovement through a tile. A limo starts moving through a tile with one of X/Y set to 0 and the other set to 6 or 18.

## Creating maps

If you wish to create your own map, change the set as startup project to MapBuilder. Make sure you have 12+ bus stops and 10+ start positions in the map you create.

# The User Interface

## Main Window



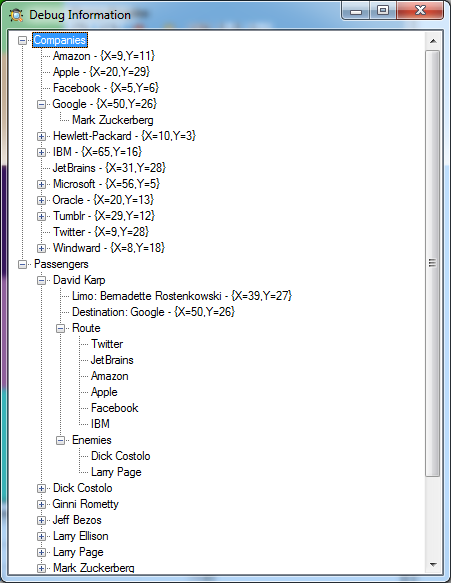
* Join – Drop all A.I. players and wait for new players to join.
* Lock – Set game ready to start. No new AI players can join and internal AI players will be added as needed to raise the total to 10.
* Play – Starts or continues the game.
* Step – Executes 1 tick of the game.
* Stop – the game is ended. Additional games may be run for the players joined.
* Speed – set how many ticks/second to run the game. Values are 1 (slowest) to 1,000. However faster speeds are also limited by the speed of the computer.
* View – set:
  + The map zoom level
  + Coordinate numbering on the top and left
  + Limo shadow – will shadow only remote A.I.s.
  + Limo shadow all – will shadow all limos.
* Turn sound effects on/off.
* Player stats window (see below)
* System messages window.
* Debug Window (see below).
* Debug Start – if checked (will have the green triangle) then as soon as a remote A.I. joins, the game will start.
* Debug Reset – click to end the game, drop all players, and open for joining.
* Full speed – will run at full speed and not update the map. It will update the player status window once per second.

## Player Stats



* Avatar – if one is not set you will get a care bear.
* Vehicle – your car.
* Score – your score for this game. Scores across all games are listed below the status bar part.
* Passenger – if you are carrying a passenger, the passenger you are carrying. No picture and the text {none} if you do not have a passenger.
* Destination – the requested destination of your passenger. If you do not have a passenger, the bus stop your path is going to.
* Team name
* The passengers you have successfully transported.

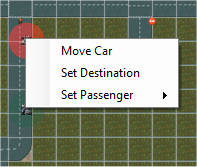
## Debug Window



This is a live list of the Companies and passengers.

* Company – lists the map location of the company.
  + All passengers waiting at that company.
* Passengers – lists by name
  + The Limo or Lobby they are in.
  + The destination they next wish to go to.
  + Their route
    - The companies, in order, they next want to visit. Does not include their present destination.
  + Their enemies
    - The other passengers they will not exit a limo if those enemies are at the bus stop.

## Right Mouse Button



When the game is ready to start or paused, you can click the right mouse button on a car and get this menu.

* Move Car – select this and then click on any straight road or a 4-way intersection. It will place the car there. For the intersection, click on the quarter tile where it enters the intersection. This will also re-calculate the path from the new location to the destination the car is presently headed to.
* Set Destination – select this and then click on the bus stop you want the car to next go to. This will also re-calculate the path to the new destination.
* Set Passenger – select the passenger you want in your car (already delivered passengers are greyed out). You can select passengers in lobbies or in other cars. When you do this the destination is **not** changed.

# Uploading to the Windward server

If you are one of the finalists at your school, you then need to upload your code to Windward.

* You will be provided a username and password for <http://labs.windward.net:81>
* Root of the ZIP file must contain your compiled executable as well as other files needed to run it directly.
  + When you upload your final code to us, upload your one client. Do not upload the server. Do not upload the other clients. (Yes people did both last year – and the multiple clients made it hard to figure out which was theirs.)
* For the provided C# client, you will need "PlayerCSharpAI.exe", **"log4net.dll", and any third-party libraries used**.
* Include source in the ZIP files as well.
* You can submit as many times as you want up to the deadline.
* Only the latest uploaded version is saved.

# Key Points

These are the issues commonly tripped over when we did play testing at Windward.

1. You have 2 programs – the server and the one client you will use. You need to run both programs.
2. You start with the sample client code and revise it. The A.I. and A\* sample code sucks.
3. You can send orders to the server at any time. You are not restricted to responding only when you get a message.
4. All units passed to/from the A.I. are in tile units. Map units are used solely within the server.
5. When cars hit a red/yellow signal, stop sign, or wish to turn left they operate the same as in real life if:
   * Cars had no turn signal until they were entering an intersection.
   * Cars would not make a right turn if going straight was not allowed.