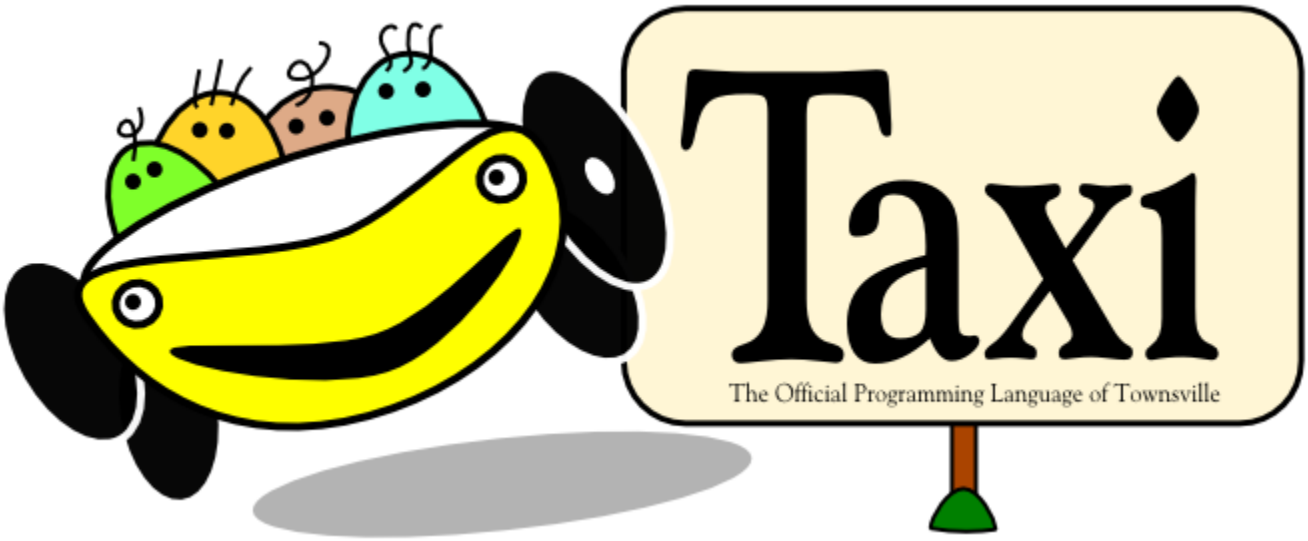


Important news!

[06/28/05] Version 1.2: Removed the first space in "Fuel 'er Up". The space+quote caused a problem with the parser.
[06/30/05] Version 1.3: Renamed "Fuel'er Up" to "Fueler Up", fixed some bugs, and raised the fare to 0.07 credits per mile.
[07/05/05] Version 1.4: Tiny change to allow compile on Windows. (I hope.)
[07/06/05] Version 1.5: Fixed a parser bug. (Thanks Paul!)
[12/21/05] Version 1.6: Added newline support for strings.
[12/21/05] Implemented **99 Bottles of Beer**.
[03/29/07] Nick Turner sent an **awesome 4 function RPN calculator** and a **patch** for the interpreter (included in version 1.7)!
[11/04/08] Jay Campbell sent me a **fun logo** for Taxi! (see above). Thanks Jay!

Check out the downloads.

Let's be honest here, what does a programmer fundamentally do? A programmer moves data from one memory location to another. Over and over. In this respect, a programmer is not unlike a taxi driver. A taxi driver moves people from place to place all day long. The people get dropped off, do their things, and new people get loaded back up to be taken someplace else - except unlike programmers, the things taxi drivers shuffle around all day pay them for the service! Well programmers no longer need to put up with that kind if ingratitude from their data: introducting the Taxi Programming Languge!

Put away your Dilbert comic clippings and hang up your cubicle pass - you're getting out of the office and about to hit the streets of Townsburg for the famous Townsburg Taxi Company, Inc. transportation service. Townsburg is filled to the brim with bits of data that need to move from here to there and back again. It will be your duty to assist these colorful folks with their tasks in the most efficient and timely manner possible.

You will be driving a Townsburg Taxi Company, Inc. state of the art taxi cab 9000R. The 9000R gets a respectible 18 miles per gallon and can hold up to 20 gallons of fuel in a full tank. Unlike the previous model 8002GT car, the 9000R seats up to 3 passengers at one time which maximizes the profit per mile ratio meaning you get more bang for your buck than ever before.

Each day your car will be waiting for you at the Taxi Garage with a full tank of gas and an empty cash box. Each passenger pays a standard fare of 0.07 credits per mile for the distance they have been riding in the cab. Passengers are not expected to pay until they are dropped off, so be sure to collect their fare at that time. Keeping the cab full of fuel throughout the day is your responsibility, so you better be sure to make enough money that you can stop at a gas station now and then to fill up the tank. If you run out of gas there is no one to come to your rescue and you might as well forget about coming to work the next day. In any case, if you keep it moving, make some money, and bring the cab back to the garage at the end of the day, you'll do fine as an employee of Townsburg Taxi Company, Inc.

Mechanics

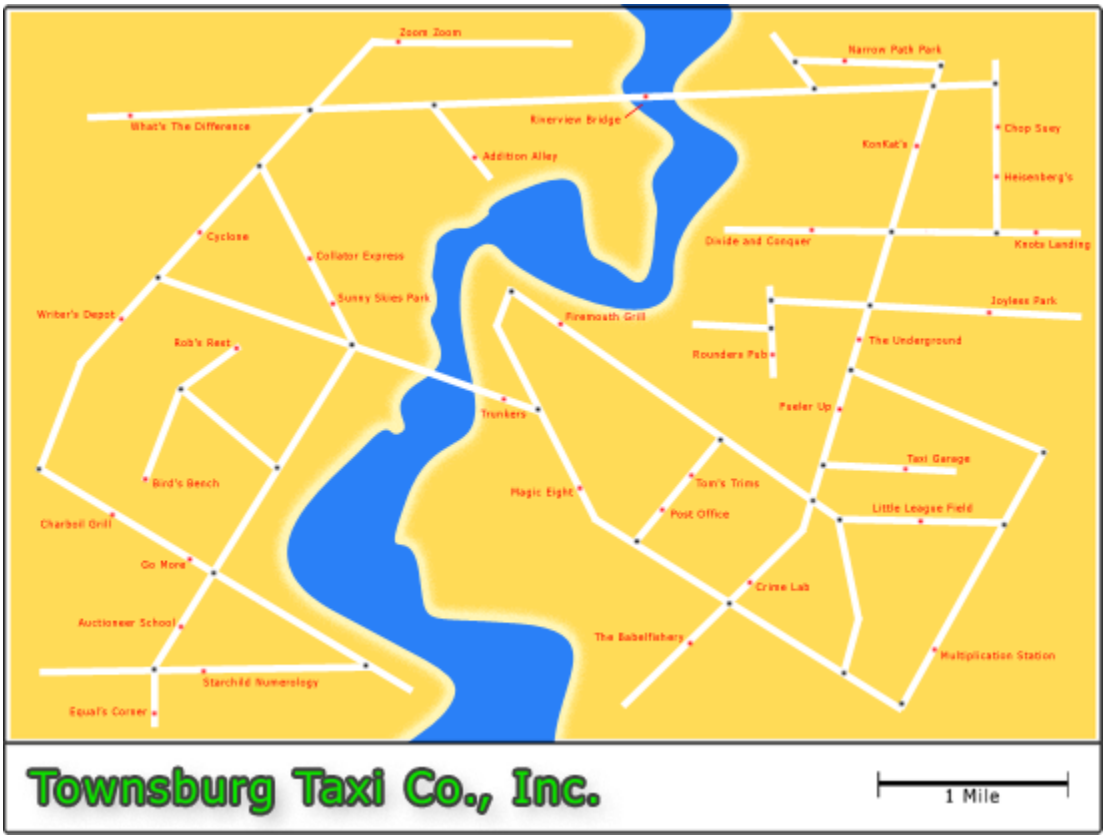
The Taxi programming language consists of instructions to you, the taxi driver. These instructions consist of directions to places to pickup or drop off passengers, where passengers are waiting to be picked up, and what to do if no one comes to the curb. Your car starts out with a full 20 gallon tank of gas and an empty cash box. The car gets 18 miles per gallon which means you cannot just drive around aimlessly all day or you'll eventually run out of gas and with an empty cash box, you won't be able to afford any more fuel. The solution to this is to pickup and drop off passengers as you move around town. They will pay you for your miles once they get where they were going and then you can pay for gas from a gas station to keep the process rolling. The fare paid by the passengers is a flat 0.07 credits per mile. Since Townsburg employs a state of the art electronic money system and your cab comes loaded with a super-accurate digital odometer, fractional credits are common and the passenger pays for exactly the distance they traveled.

Operations take place at destinations throughout Townsburg. Drop two or three passengers off at Addition Alley, and their numerical values are added together with the resulting value becoming a passenger waiting at the curb for pickup. Passenger drop off and payment happens automatically. When you are carrying one or more passengers who are going to the place you just stopped at, they will get out in the order they got into the taxi. In other words, the passenger who has been in your car and destined for that location the longest will get out first, followed by the next longest, and so on. Most destinations will result in new passengers to pick up, however they will not automatically board the taxi. You must explicitly pick them up which also allows for the passenger to declare their destination. Most places will allow any number of passengers to queue up on the curb which means you do not necessarily have to pickup the results right away - you could go and do something else and then come back for them later if needed.

When stopping at a gas station, the maximum amount of gas will automatically be bought and paid for from your store of credits in the cash box. If you have enough cash, the entire tank will be filled up. You are only charged for the exact amount of

fuel you purchase. Passengers cannot be dropped off at gas stations nor can any be picked up from them.

Road Map



(click map for larger image)

Destinations

Addition Alley	adds numerical passengers together, anything non-numeric is an error
Auctioneer School	converts string passengers to uppercase, non-string is an error
Bird's Bench	one passenger can wait here until later, but only 1
Charboil Grill	convert a numerical passenger to a single ASCII character (string) or vice-versa, strings longer than 1 are an error
Chop Suey	takes a string passenger and breaks it up into individual string passengers that hold one character each, so "Hi" results in 2 passengers: "H" and "i", non-string is an error
Collator Express	tests if the first string passenger is less than the second and returns the first if true or no one if not true, non-string is an error
Crime Lab	tests if all dropped off string passengers are equal to each other, if so returns 1 passenger with the value, otherwise no passenger is returned, non-string is an error

Cyclone	makes clones of passengers, drop 1 off, get original plus 1 copy back, drop 3 off, get original 3, plus 3 copies back, etc.
Divide and Conquer	divides numerical passengers, anything non-numeric is an error
Equal's Corner	tests if all dropped off numerical passengers are equal to each other, if so returns 1 passenger with the value, otherwise no passenger is returned, non-numeric is an error
Firemouth Grill	any number of passengers can be dropped off here, but picked up in random and unknown order
Fueler Up	gas station: 1.92/gallon
Go More	gas station: 1.75/gallon
Heisenberg's	pickup an unspecified random integer
Joyless Park	passengers dropped off here form a FIFO queue so they can be picked up again later
Knots Landing	inverts boolean logic via numerical passengers: non-zero becomes 0, 0 becomes 1, non-numerical is an error
KonKat's	concatenates string passengers, anything non-string is an error
Little League Field	converts string passengers to lowercase, non-string is an error
Magic Eight	tests if the first passenger is less than the second and returns the first if true or no one if not true, non-numerical is an error
Multiplication Station	multiplies numerical passengers, anything non-numeric is an error
Narrow Path Park	passengers dropped off here form a stack (LIFO) so they can be picked up again later
Post Office	drop off string passengers to print to stdout, pickup a passenger to read a string line from stdin
Riverview Bridge	passengers dropped off at Riverview Bridge seem to always fall over the side and into the river thus the driver collects no pay, but at least the pesky passenger is gone
Rob's Rest	one passenger can wait here until later, but only 1
Rounders Pub	rounds numerical passengers, non-numerical is an error
Starchild Numerology	pickup a specified numerical value
Sunny Skies Park	passengers dropped off here form a FIFO queue so they can be picked up again later
Taxi Garage	starting and termination point

The Babelfishery	translates a numerical passenger to a string passenger or vice-versa
The Underground	takes 1 numerical passenger and subtracts 1, if the result is 0 or less than 0, no passenger is returned otherwise the result is returned, non-numerical is an error
Tom's Trims	removes whitespace from beginning and ending of string passengers, non-string is an error
Trunkers	truncates numerical passengers to an integer, non-numerical is an error
What's The Difference	subtracts numerical passengers, anything non-numeric is an error
Writer's Depot	pickup a specified string
Zoom Zoom	gas station: 1.45/gallon

Syntax

The best way to learn Taxi is by example, however you should be aware of a couple issues with the syntax. Below are a few sample lines:

```
42 is waiting at Starchild Numerology.
43 is waiting at Starchild Numerology.
"Hello" is waiting at the Writer's Depot.
Go to Starchild Numerology: west 1st left, 2nd right, 1st left, 1st left, 2nd left.
Pickup a passenger going to The Underground.
Pickup another passenger going to The Underground.
Go to Writer's Depot: west 1st right, 1st left, 1st right.
Pickup a passenger going to the Post Office.
```

The text highlighted in red are data values and can be any value you wish to have waiting at the specified location. Note, however, that the only two locations where you are allowed to declare what passengers are waiting are Starchild Numerology and Writer's Depot. Also note that those locations are data-type-sensitive. If you try to declare passenger "Hi" is waiting at Starchild Numerology, you will most likely end up with a 0 waiting for you there instead of the string you might have thought would be there.

Text marked with orange is optional. In certain positions, the word "the" is allowed because it may make for a more naturally readable sentence. Also "another" is allowed when picking up passengers in place of the "a" for similar reasons.

All names marked with green are destinations on the Townsburg map. As with all text in Taxi, they are case sensitive and must match **exactly** with the names on the map and destination list. Note that some names include the word "The" in them (such as The Babelfishery) and so they must always be specified with "The" capitalized.

Traveling Townsburg is how you get your computing done. This is accomplished by given directions to your next destination that are relative to your current location. First you designate the initial starting direction of travel on the street you are currently on. This is done using normal cardinal directions (north/south/east/west).

What follows is the list of turns that must be made at each intersection encountered on the way to the destination. Each intersection on the map is marked with a black dot (the red dots are destination locations). Once you are on the street which has the destination, and you are traveling in the correct direction, the car should arrive where it is going including traveling straight through any intersections that might be between your present position and the final stop. Hopefully this becomes more clear as you trace through some of the below examples.

Branching is done using labels and another command format as seen in the sample lines below:

```
[loop]
.. many code lines ..
Switch to plan "end_loop" if no one is waiting.
Pickup a passenger going to The Underground.
Switch to plan "loop".
[end_loop]
.. continue on with more code ..
```

The red text is any label that you choose. Labels can be anywhere within the program script and have almost any name. Note that **Switch to plan "end_loop" if no one is waiting.** is a conditional jump and the only way to do such things within Taxi. Leaving off the "if no one is waiting" part makes it an unconditional jump.

Nearly all commands in Taxi are case sensitive, so best just model your code after the code samples seen here to be safe. All sentences must end in a period. Comments are easily supported by defining labels with the comments in them as labels are technically not code and so will not interfere with the rest of the script. Note however that you cannot insert comments or labels in the middle of a command sentence.

Examples

Here is the classic "Hello, World!" program. For a program to be successful, you must end the run with your cab at the Taxi Garage. This program is short enough that we need not concern ourselves with visiting a gas station, but in longer programs it becomes critical to track fuel usage so that your car doesn't run out of gas.

```
"Hello, World!" is waiting at the Writer's Depot.
Go to Writer's Depot: west 1st left, 2nd right, 1st left, 2nd left.
Pickup a passenger going to the Post Office.
Go to the Post Office: north 1st right, 2nd right, 1st left.
Go to the Taxi Garage: north 1st right, 1st left, 1st right.
```

The following will calculate the first 30 numbers of the Fibonacci sequence and represents a more complex Taxi program. This program forces us to visit a gas station during the loop because otherwise we'd never have enough fuel to print all 30 numbers. It also uses both a conditional and unconditional jump as well as comments.

```
[ This is the first 30 Fibonacci numbers implemented in the Taxi programming language
]
[ Author: BigZaphod (sean -at- fifthace.com) ]
[ License: Public Domain ]
[ http://www.bigzaphod.org/taxi/ ]
```

```
1 is waiting at Starchild Numerology.
1 is waiting at Starchild Numerology.
30 is waiting at Starchild Numerology.
Go to Starchild Numerology: west 1st left, 2nd right, 1st left, 1st left, 2nd left.
Pickup a passenger going to Cyclone.
Pickup a passenger going to Rob's Rest.
Pickup a passenger going to Sunny Skies Park.
Go to Sunny Skies Park: west 1st right.
Go to Rob's Rest: south 2nd right, 1st right.
[B]
Go to Cyclone: south 1st left, 1st left, 1st left, 1st right.
Pickup a passenger going to The Babelfishery.
Pickup a passenger going to Addition Alley.
Go to The Babelfishery: south 1st left, 2nd right, 1st right.
Pickup a passenger going to the Post Office.
" " is waiting at Writer's Depot.
Go to Writer's Depot: north 1st left, 1st left, 2nd left.
Pickup a passenger going to the Post Office.
Go to Post Office: north, 1st right, 2nd right, 1st left.
Go to Rob's Rest: south 1st right, 1st left, 1st left, 1st right, 1st right.
Pickup a passenger going to Cyclone.
Go to Cyclone: south 1st left, 1st left, 1st left, 1st right.
Pickup a passenger going to Addition Alley.
Pickup a passenger going to Cyclone.
Go to Zoom Zoom: north.
Go to Addition Alley: west 1st left, 1st right.
Pickup a passenger going to Rob's Rest.
Go to Sunny Skies Park: north 1st left, 1st left, 1st left.
Pickup a passenger going to The Underground.
Go to The Underground: south 1st left, 1st right, 2nd left.
Switch to plan "C" if no one is waiting.
Pickup a passenger going to Sunny Skies Park.
Go to Sunny Skies Park: south 2nd right, 1st left, 1st right.
Go to Rob's Rest: south 2nd right, 1st right.
Switch to plan "B".
[C]
Go to the Taxi Garage: south 2nd left.
```

Downloads

The version 1.7 C++ source for the interpreter is available. (it is very scary)

Hello, World and Fibonacci sample code.

Echo will read a string from stdin, and then echo it back to stdout.

I implemented the classic 99 Bottles of Beer program. It doesn't handle the plural/singular "bottles" or the end case very well, but it works! Improvements are welcome.

Nick Turner's 4 function RPN calculator is an impressive piece of work. Here's what he has to say about it: *It should be pretty self explanatory, and it provides help and error messages as it goes along. Entering numbers pushes them to the stack; +, -, *, and / all pop two values and push their result; "top" shows the top value (without destroying it) and "empty" clears the stack; "bye" will show the top value if it isn't empty, then quits. Everything except "bye" cleans up the town after itself (no passengers left waiting anywhere except the stack) and the program manages to make a substantial profit.* Thanks Nick!

History

Conceived on June 21, 2005 and first released on June 27th, 2005.

check out my other stuff