Assignment 3 - Fruit Bot

spec version: 1.1 | last updated: 2018-05-21 14:00:00 +1100

Introduction

Bots have been in the news for doing negative things such as spreading fake news.

Your task in this assignment is to write a bot that makes the world a better place.

Your bot will pilot a drone which ferries fruit to UNSW students so they stop eating junk food, have a healthy diet.

Your bot must successfully buy, transport and sell fruit in a simulated world.

You will write a C program to implement this bot.

Virtual World

The virtual world in which your bot will operate has locations spread around a ring.

It is possible to move in only two directions in the virtual world: **east** and **west**. You can not move **north** or **south**.

The word is circular so if you keep moving **east** or **west** you eventually return to where you started.

The details of the virtual world are randomly generated for each simulation. An example follows.

Locations are listed going east. As the world is circular if you go **east** from **Kora Lab** you go to **CSE**. Similarly if you go **west** from **CSE** you go to **Kora Lab**.

There is a diagram here that may help you to better understand how the world fits together. To obtain a copy you can edit on draw.io, copy the contents of this txt file into your browser's address bar.

Name	Type	Fruit	Quantity (kg or kJ)	Price
CSE	*Other*			

Bondi Banana Growers	*Seller*	Bananas	32	$48
Paddington Banana Plantation	*Seller*	Bananas	25	$61
Physics Theatre	*Buyer*	Bananas	8	$111
Clovelly Watermelons	*Seller*	Watermelons	85	$14
Wentworth Watermelons	*Seller*	Watermelons	37	$13
Quadrangle	*Other*			
Kingsford Oranges	*Seller*	Oranges	50	$23
Mathews B	*Buyer*	Watermelons	73	$22
Kensington Apple Farm	*Seller*	Apples	43	$18
J17 G03	*Buyer*	Oranges	30	$49
Smelly Fruits R Us	*Seller*	Durian	37	$28
Physics Lawn	*Other*			
Drone Servicing	*Seller*	Electricity	100	$4
CLB 7	*Buyer*	Apples	33	$42
K17 Basement	*Buyer*	Bananas	8	$100
Rosebery Orange Grove	*Seller*	Oranges	69	$19

Campus Compost Heap	*Buyer*	Anything	1000	$1
Viola Lab	*Buyer*	Durian	17	$40
Mathews A	*Buyer*	Apples	44	$33
Anzac Parade	*Other*			
Science Theatre	*Buyer*	Apples	58	$32
Clancy Auditorium	*Buyer*	Oranges	23	$24
Campus Citrus Centre	*Seller*	Oranges	56	$19
Randwick Apple Orchard	*Seller*	Apples	85	$22
Maroubra Melons	*Seller*	Watermelons	110	$12
Campus Computers	*Seller*	Apples	101	$21
Campus Charging	*Seller*	Electricity	100	$4
Sitar Lab	*Buyer*	Durian	27	$37
Kora Lab	*Buyer*	Durian	33	$47

Note most locations sell or buy a particular fruit - for example **Clovelly Watermelons** has 85 kg of watermelons for sale at $14/kg.

What A Fruit Bot Should Do

Your bot has a fixed number of turns in the *Fruit Bot* world. Its only goal is to have as much cash as possible after the last turn of the simulation.

Your program is run once per turn and given a description of the world on stdin and is asked to choose a single action for your bot by printing a single line of output.

There are three types of actions: **Move**, **Buy** and **Sell**.

A bot makes money by buying fruit from one location and selling to another location at a higher price.

For example, it might buy Bananas from **Bondi Banana Growers** at $32/kg each and sell them to Physics Theatre at $111/kg.

A bot must move to a location before it can buy or sell from that location.

A bot's only goal is to maximize the cash it has at the end of the simulation. A bot makes cash by buying fruit at one location, and then moving to a location where it can sell the fruit.

There is a maximum weight of fruit that a bot can carry. This limit does not change during the simulation.

Each location buys or sells at most one type of fruit (exception below). No location both buys and sells fruit. Some locations do not buy or sell any fruit.

Move

A bot making a **Move** action specifies the number of locations they wish to move. If the number is positive the bot moves East. If the number is negative the bot moves West. It is not possible to move North or South.

The Fruit Bot world is circular - you can move from the last location to the first location and vice-versa. So if a bot keeps moving East or West it will eventually return to where it started.

For example, in the world above a bot at **Physics Theatre** which makes a **Move 3** action would go to **Quadrangle**. If instead a bot at **Physics Theatre** made a **Move -5** action it would go to **Sitar Lab**.

There is a limit on the number of locations a bot can move in one turn. This limit does not change during the simulation.

Each location a bot moves will use up 1 kJ of electricity from its battery. A bot with battery level 0 can not move - it can still buy or sell at its current location, but unless it can buy Electricity at this location, it will not be able to move again during the simulation.

A good bot will likely need to charge its battery by buying Electricity

A bot attempting to make a move that would use up more than its available battery or exceed the move limit is not penalised. It is moved the maximum possible distance.

Buy

A bot making a **Buy** action must specify how many kg of fruit they wish to buy. A bot may receive less kg than it requests. This will occur if:

- the quantity exceeds the kg of fruit the seller has
- the bot has insufficient cash
- the kg would exceed the bot's weight limit
- the bot already has a different type of fruit on board - a bot can only carry one sort of fruit
- other bots simultaneously try to buy the fruit from the seller and the requests can't be fairly satisfied (see Multi-bot Worlds description)

A bot's cash is reduced by the total price of the fruit (cost/kg * kg) it receives in a **Buy** action.

A bot will usually then use one or more **Move** actions to go to a location which buys this fruit. A bot may however choose to buy more fruit of the same type.

Sell

A bot making a **Sell** indicates how many kg of fruit they wish to sell. A bot making a **Sell** action must be on a location which buys this type of fruit. The amount that is actually sold may be limited by the amount the buyer wishes to buy. This and the price the buyer will pay is indicated in the location's description.

A bot's cash will be increased by the total price of the fruit (cost/kg * kg) it sells in a **Sell** action.

Bots are not penalised for attempting to buy or sell more fruit than is possible. The transaction will be carried out as much as is possible. For example, if a bot attempts to buy 10000 kg of fruit from a location which only has 25 kg, they will be sold 25 kg (other restrictions permitting).

The initial state of the world is randomly determined at the beginning of each simulation. The details of the world including prices do not change during the simulation. The only exception is that the amount of fruit buyers and sellers are willing to trade reduces as bots buy and sell fruit.

You can not assume a location of a particular name will be in the world. For example, you can **not** assume a location named "Randwick Apple

Orchard" will be present in the world.

If a location of a particular name is present, you can **not** assume it it is a buyer or a seller, or that it buys or sells a particular fruit.

For example, if "Randwick Apple Orchard" is present you can **not** assume it is a seller and you can assume it sells or buys fruit.

You can not assume a location of a particular fruit will be in the world. For example, you can **not** assume "Apples" will be present in the world.

Fruit Bot worlds will always contain one or more locations which will buy any fruit. These locations indicate they buy "**Anything**".

Fruit Bot worlds will always contain one or more locations which sell "**Electricity**". Buying electricity charges a bot's battery. It does not affect a bot's weight limit. A bot can buy electricity when carrying fruit. Electricity cannot be sold by a bot. Bots always start with a full battery.

Multi-bot Worlds

Your bot will be tested in Fruit Bot worlds where it is the only bot present.

It will also be tested in Fruit Bot worlds where other bots are present.

All bots start at the same location, with the full batteries of the same size & cash and have the same number of turns. All bots have the same limits on how far they can move and the maximum fruit weight they can carry.

When multiple bots are present in a world each turn all bots are simultaneously asked for their action. The actions then occur simultaneously.

Multiple bots may be present on one location. If multiple bots on the one location make buy or sell requests which cannot all be satisfied the trade is divided fairly between the bots. Fruit will be left unsold if it is not possible to divide trade fairly.

If, for example, there is 7 kg of fruit for sale at a location, and 3 bots simultaneously request to buy, respectively 7kg, 18kg and 2kg, each bot would be sold 2 kg and 1 kg would be left unsold.

If instead the 3 bots simultaneously requested to buy, respectively 18kg, 7kg and 1kg, the bots would sold 3 kg, 3 kg and 1kg respectively.

Getting Started

The week 11 lab exercises will take you through getting started.

First download **fruit_bot.c** which contains the starting code for the assignment.

You will also need **fruit_bot.h** which **fruit_bot.c** includes. Read but do not change **fruit_bot.h**

The code to read the input describing the world for this assignment is difficult.

fruit_bot.c calls a function named **fruit_bot_input**. You will also need **fruit_bot_input.c** which defines **fruit_bot_input**. It is a requirement you use **fruit_bot_input** for input. You don't need to understand the code in **fruit_bot_input.c**. Your program must not use scanf, fgets or getchar, it must just call **fruit_bot_input**

fruit_bot_input returns a pointer to a struct. A full description of the Fruit Bot world can be obtained by following pointers starting with this struct.

A significant component of this assignment is understanding the representation of the Fruit Bot world. Start by reading **fruit_bot.h**

The command `1511 fruit_bot_referee fruit_bot.c` will automatically test your program on a random world.

For example:

```
$ cp -n /web/cs1511/18s1/activities/fruit_bot/fruit_b
$ cp /web/cs1511/18s1/activities/fruit_bot/fruit_bot
$ cp /web/cs1511/18s1/activities/fruit_bot/fruit_bot
$ dcc fruit_bot.c fruit_bot_input.c -o fruit_bot
$ 1511 fruit_bot_referee fruit_bot.c|more
seeding with 14555711
dcc -o fruit_bot fruit_bot.c -I/home/c
dcc --valgrind -o fruit_bot-valgrind f

*** Fruit Bot Parameters ***
battery_capacity=105
maximum_fruit_kg=32
maximum_move=6

*** Turn 1 of 33 *** ***

Campus Charging: will sell 100 kJ of E
Science Theatre: will buy 2 kg of Appl
CLB 7: will buy 2 kg of Apples for $10
```

```
Mathews A: will buy 1 kg of Apples for
Quadrangle: other

"Botty McBotbot" is at "Campus Chargin
```

An interactive player is available allowing you to control a fruit bot directly:

```
$ dcc fruit_bot.c fruit_bot_input.c -o fruit_bot
$ 1511 fruit_bot_referee --interactive_player
...
```

The default world used by the fruit bot referee has only a few locations and is designed for debugging.
A larger world is available via the **-w** option.

```
$ 1511 fruit_bot_referee -w medium fruit_bot.c|mor
Version: 0.2
seeding with 8208139
dcc -o fruit_bot fruit_bot.c -I/home/c
fruit_bot/fruit_bot_input.c
dcc --valgrind -o fruit_bot-valgrind f
/private/activities/fruit_bot/fruit_bo

*** Fruit Bot Parameters ***
battery_capacity=39
maximum_fruit_kg=21
maximum_move=3

*** Turn 1 of 128 *** ***

John Lions Garden: other
Bondi Banana Growers: will sell 25 kg
Kensington Kiwifruit: will sell 23 kg
Campus Citrus Centre: will sell 15 kg
Mathews C: will buy 22 kg of Kiwifruit
Rosebery Orange Grove: will sell 18 kg
Viola Lab: will buy 34 kg of Durian fo
Wentworth Watermelons: will sell 13 kg
....
```

Other worlds will be added to the referee and you can create your own world in a file (see lab exercises for examples):

```
$ 1511 fruit_bot_referee -f my_world.txt fruit_bot.c
```

These tutor-made solution videos for Traversing the Fruit Bot World and Go West Young Bot from week 11's lab might be helpful for understanding this assignment.

Fruit Bot Tournaments

Tournaments will be run on all bots submitted with *give*, starting Sun May 20 with the results displayed on the class web page.

This will allow you to see how your bot performs in a world with many other bots present.

Submitted bots will be first tested in a single-bot-world. They will only be be added to the tournament if they reach a qualifying-level of performance in a single-bot world.

Assumptions/Restrictions/Clarifications

You should follow discussion about the assignment in the class forums. Questions about the assignment should be posted there so all students can see the answer.

You must use **fruit_bot.h** and **fruit_bot_input.c**.

You can not change **fruit_bot.h** or **fruit_bot_input.c**.

You can not submit **fruit_bot.h** or **fruit_bot_input.c**.

You may submit only one file. It must be named fruit_bot.c

You submitted code must be C only. You may not submit code in other languages. You may not use *system* or other C functions to run external programs.

You may call functions from the standard C libraries (e.g. the functions from `stdio.h`, `stdlib.h`, `string.h`) and the maths library (`math.h`).

You may not use functions from other C libraries (libraries that require the -l flag for dcc). In other words as long as dcc compiles your program without the -l flag you are fine.

Your program must take at most 10 seconds to print a move on a CSE computer when compiled with dcc --valgrind.

There is no way for your bot to pass information from one turn to the next. Your bot is not permitted to create files.

Sellers do not grow more fruit during the simulation. The kg of fruit they have for sale is reduced when bots buy fruit from them. The kg of fruit they have for sale does not otherwise change.

This is also true for buyers. The kg of fruit a buyer is willing to buy reduces when bots sell fruit to them. The kg of fruit they are willing to buy does not otherwise change during the simulation.

fruit_bot.h will not be changed when your bot is compiled for marking.

Attribution of Work

This is an individual assignment. The work you submit must be your own work and only your work apart from exceptions below. Joint work is not permitted.
You may use small amounts (< 10 lines) of general purpose code (not specific to the assignment) obtained from site such as Stack Overflow or other publically available resources. You should attribute clearly the source of this code in a comment with it.

You can not use other people's code to generate code you use or data you use. For example, you can not use other people's machine learning software to generate neural network weights or a decision tree.

You are not permitted to request help with the assignment apart from in the course forum, help sessions or from course lecturers or tutors.

Do not provide or show your assignment work to any other person (including by posting it on the forum) apart from the teaching staff of COMP1511. Do not post your code in a publically available - e.g. a public github repo.

The work you submit must otherwise be entirely your own work. Submission of work partially or completely derived from any other person or jointly written with any other person is not permitted. The penalties for such an offence may include negative marks, automatic failure of the course and possibly other academic discipline. Assignment submissions will be examined both automatically and manually for such submissions.

Relevant scholarship authorities will be informed if students holding scholarships are involved in an incident of plagiarism or other misconduct. If you knowingly provide or show your assignment work to another person for any reason, and work derived from it is submitted you may be penalized, even if the work was submitted without your knowledge or consent. This may apply even if your work is submitted by a third party unknown to you.

Note, you will not be penalized if your work is taken without your consent or knowledge.

Submission of Work

You are required to submit intermediate versions of your assignment. Every time you work on the assignment and make some progress you should copy your work to your CSE account and submit it using the give command below.

It is fine if intermediate versions do not compile or otherwise fail submission tests.

Only the final submitted version of your assignment will be marked.

All these intermediate versions of your work will be placed in a git repo and made available to you via a web interface at this URL, replace *z5555555* with your zpass.
`https://gitlab.cse.unsw.edu.au/`*`z5555555`*`/18s1-comp1511-ass3/commits/master`

This will allow you to retrieve earlier versions of your code if needed.

You submit your work like this:

```
$ give cs1511 ass3 fruit_bot.c
```

Assessment

This assignment will contribute 12% to your final mark.

80% of the marks for this assignment will be based on the performance of your *Fruit Bot* program, i.e. how good it is at buying and selling fruit to make a profit.

20% of the marks for assignment 3 will come from hand marking of the readability of the C you have written. These marks will be awarded on the basis of clarity, commenting, elegance and style. In other words, your tutor will assess how easy it is for a human to read and understand your program.

Tutors will also examine your testing strategy and unit tests. We understand that for some approaches to this assignment unit tests will be hard to write, include those you can and make sureyou describe your testing strategy above the **run_unit_tests** function. Here is an indicative marking scheme.

HD (85-100%)	bot performs well in multi-bot simulations, well-explained beautiful code.
DN (75-85%)	bot makes good profits in single bot simulations, often makes profit in multi-bot simulations, very

	readable code.
CR (65-75%)	reliably makes a non-trivial profit in single bot simulations, readable code.
PS (50-65%)	often makes a non-zero profit in single player situations, code is mostly readable
40-50%	serious attempt on assignment
-70%	Knowingly providing your work to anyone and it is subsequently submitted (by anyone).
-70%	Submitting any other person's work. This includes joint work.
0 FL for COMP1511	Paying another person to complete work. Submitting another person's work without their consent.

The lecturer may vary the assessment scheme after inspecting the assignment submissions but it will remain broadly similar to the description above.

Due Date

This assignment is due Sunday 03 June 23:59:59
If your assignment is submitted after this date, each hour it is late reduces the maximum mark it can achieve by 5%. For example if an assignment worth 74% was submitted 5 hours late, the late submission would have no effect. If the same assignment was submitted 10 hours late it would be awarded 50%, the maximum mark it can achieve at that time.

Changelog

v1.1 (2018-05-21 14:00:00 +1100)	interactive player added to fruit_bot_referee fruit_bot_referee creates input file allowing you to reproduce an error example using interactive player added to spec minor bug in fruit_bot_input.c corrected

COMP1511 18s1: Programming Fundamentals is brought to you by
the School of Computer Science and Engineering at the University of New South Wales, Sydney.
For all enquiries, please email the class account at

cs1511@cse.unsw.edu.au

CRICOS Provider 00098G