UTEK

PROGRAMMING COMPETITION
COMPETITORS' PACKAGE

University of Toronto Engineering Kompetition
January 12-13, 2019

Director: ***Nirmit Zinzuwadia***

UTEK 2019 Sponsors:

Google

McKinsey &Company

UTEK Programming Package

Welcome to UTEK's 2019 programming competition.

Rules

- You may not collaborate with anyone other than members of your team.
- Cite all the libraries used.
- If you are not able to finish later parts of the competition, outline your solution for the presentation.

Deliverables

1. *Code*
   All code and documentation you have written for the competition, including a README that contains relevant citations and running instructions. The code is due at 5:00.

2. *Test case output*
   We will provide test inputs. Run your code on them and submit the output. The outputs are due at 5:00.

3. A *presentation* summarizing the key aspects of the problem, your approach and your solution. The slides are due at 7:00. Please note that due to space constraints, only the top six teams will be presenting. You will be notified by email on Saturday night.

	Time	**Additional Information**
Team Presentation	10 minutes	Reminder will be at 1-minute mark. There is a grace period of 30 seconds, after which the presenters **will** be cut off.
Judges Q&A	5 minutes	All members of the team must be ready to answer questions.

Submission Method

Submit your code and input/ output files zipped to programming@utek.skule.ca, with the subject line [UTEK 2019 Submission] Team N, where N is your team number. Please cc all of your teammates and include your names in the body of the email. Email your presentation to the same email address with the subject [UTEK 2019 Presentation] Team N, where N is your team number.

Problem Statement

The advent of e-commerce has ushered in the golden era for autonomous warehouse robots. As these autonomous robots become ubiquitous around the world, large investments are being made by major tech firms and venture capitalists to develop optimal path planning algorithms. Path planning algorithms enable robots to efficiently travel within warehouses while collecting items, avoiding static and dynamic obstacles, and communicating with other robots to distribute workload. You and your team have recently launched a start-up called eBotics™ and have secured funding from Sequoia Capital to develop a state-of-the-art path planning algorithm for an autonomous warehouse robot. The details of the algorithm you are required to create are presented in the following sections.

eBotics™ has several prototype autonomous warehouse robots they are using to develop and evaluate the efficacy of their path planning algorithm. Each robot is given a list of products as well as the locations of the individual products. All locations are provided in cartesian coordinates with the robot starting and ending its path at the origin or (0,0). Robots are free to traverse any path within the 100 by 100 square grid that is free of obstacles. The primary objective of eBotics™ is to develop an algorithm for any robot to collect all the required items in the shortest time possible. Solutions will be judged on both completeness and optimality.

Input/ Output Details

Part	Given Files	Submitted Files
1	1a.in, 1b.in, 1c.in	1a.out, 1b.out, 1c.out
2	2a.in, 2b.in, 2c.in	2a.out, 2b.out, 2c.out
3	3a.in, 3b.in, 3c.in	3a.out, 3b.out, 3c.out
4	4a.in, 4b.in, 4c.in	4a.out, 4b.out, 4c.out
5	5a.in, 5b.in, 5c.in	5a.out, 5b.out, 5c.out
6	See question	See question

Each question 1-5 has three test cases, which will be in files e.g. 1a.in, 1b.in, 1c.in. One test case (a) will be provided at the beginning of the competition, the rest will be provided one hour before the end of the competition.

The output should be in files with the same name, but extension .out, e.g. 1a.out, 1b.out, 1c.out. Make sure you submit all the files with correct naming and formatting as described below. Parts 2-5 will be auto-graded.

Input

The format will be as follows. Comments are provided for explanation, and will not be present in the actual input

```
2, 1, 1
# Three comma-separated integers for the
# number of robots, number of items and number of obstacles respectively
# There will be fewer than 100 robots
(1, 1, 3, 2.5)
# A tuple of four values for each item to be collected:
# (x, y, product number, weight)
# x, y, product number will be integers, weight will be an integer or float
(1, 2, 3, 5)
# A tuple of four integers for each obstacle: (x1, y1, x2, y2)
# Bottom-left corner is (x1, y1), the top-right corner is (x2, y2)
# i.e. x1 <= x2, y1 <= y2
```

Output

Each robot has four possible actions, and is controlled independently: move, pick, drop, rest. Each action takes one time unit.

```
move x y
# move to the given x,y coordinates. The robot can move to any of the eight
# adjacent squares (like a king in chess).
# Takes two integer arguments
pick product_number
# pick up one object with the given product_number at the current location.
# Takes one argument
drop product_number
# drop the object with the given product number at the current location.
# Takes one integer argument
rest # Do nothing. Takes no arguments
```

The output format for parts 2-5 will be as follows. Comments are provided for explanation, and must not be present in the actual output.

```
# commands for the i-th robot are in the i-th column, separated by semicolons
move 1 1; rest
# first robot moves diagonally to 1, 1; second robot does nothing
pick 3; rest
# first robot picks up product #3 at 1, 1
move 0 0; rest
drop 3; rest
# robot moves back to the origin and deposits the object
```

Challenge

Part 1

At the end of the day, robot is given a list of orders that consumers purchased. The items are not aggregated. As orders come into the system, the system simply stores the results into a file. Write a program to aggregate the location, product number and weight for each product.

Input: As described above, with no obstacles. the format is (x, y, product number, weight (in kg)>)

```
1, 7, 0 # number_robots, number_items, number_obstacles
(2, 1, 1, 5.0)
(2, 5, 3, 2.5)
(4, 3, 2, 2.0)
(2, 5, 3, 2.5)  # note that this is a duplicate from above, so the qty is 2
(5, 6, 8, 2.5)
(7, 1, 9, 0.5)
(6, 6, 4, 79.0)
```

Output: Aggregated output must be summarized in a file with the following format and increasing product number.

```
Product Number: 1; Weight: 5.0; Qty: 1; Location: (2, 1)
Product Number: 2; Weight: 2.0; Qty: 1; Location: (4, 3)
Product Number: 3; Weight: 2.5; Qty: 2; Location: (2, 5)
Product Number: 4; Weight: 79.0; Qty: 1; Location: (6, 6)
Product Number: 9; Weight: 0.5; Qty: 1; Location: (7, 1)
```

Part 2

The robot starts at the origin (0, 0). It will need to pick up all of the items, but has a maximum capacity of 100kg. The robot is free to roam anywhere in the 100x100 warehouse. Note in the diagram, only a 10x10 sample is shown.

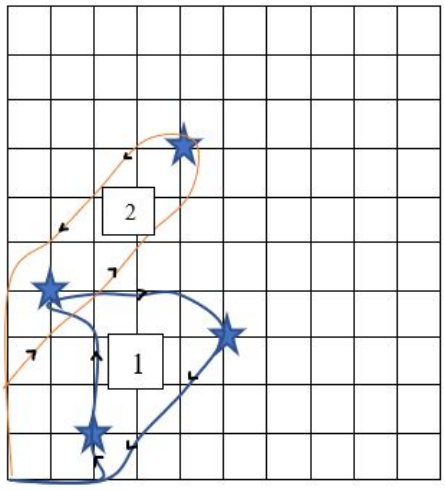

Figure 1: Example solution for part 2. The origin is in the bottom-left corner. The stars denote the position of items. Note that the robot returns to its origin to drop off items.

Input: Format as given above. There will only be one robot and no obstacles.
Output: Format as given above.

Part 3

As before, the robot will start at location (0,0) and must to the several locations to pick up all the items. Now, there are multiple obstacles present within a warehouse.

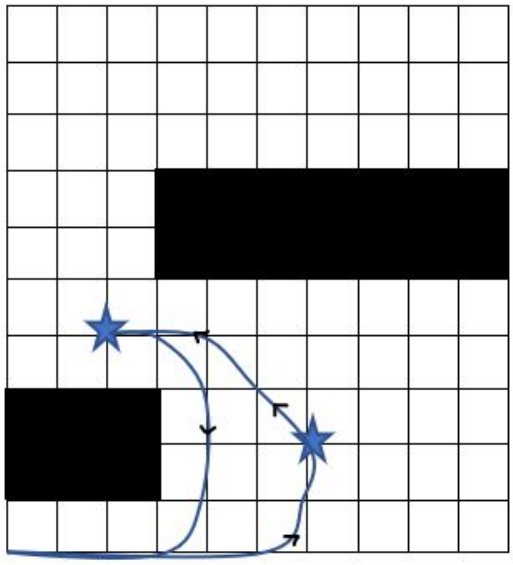

Figure 2: Example solution for part 3. The black boxes denote obstacles. Note that the robot cannot touch them.

Input: Format as given above. There will only be one robot with obstacles.
Output: Format as given above

Part 4

To expedite the orders, eBotics™ decides to buy more robots that will operate simultaneously. In this section, the robots should work together to pick up the items. Make sure that the robots do not collide. There are no obstacles in this section. The i'th robot will start at (i, 0). All items must be delivered to the origin (0, 0).

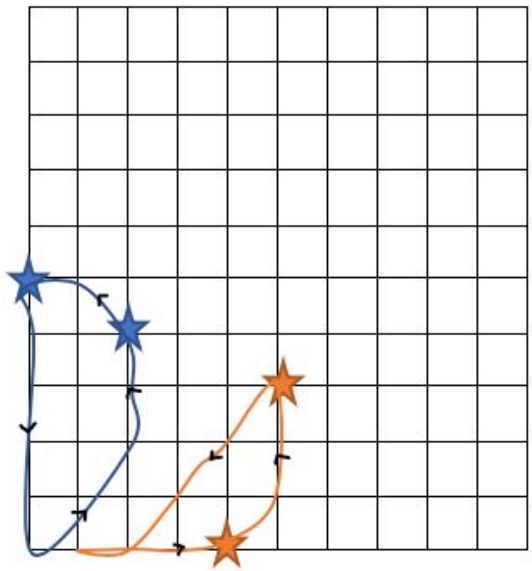

Figure 3: Possible solution for part 4, with the two robots working as a team. Note that the 2nd robot starts at (i, 0). The last couple steps, where the orange robot moves to the origin are omitted for clarity.

Input: Format as given above. There will be multiple robots, no obstacles.
Output: Format as given above

Part 5

Unfortunately, eBotics™ continues to have obstacles in their warehouse. Ensure that the robots do not collide with each other or with obstacles.

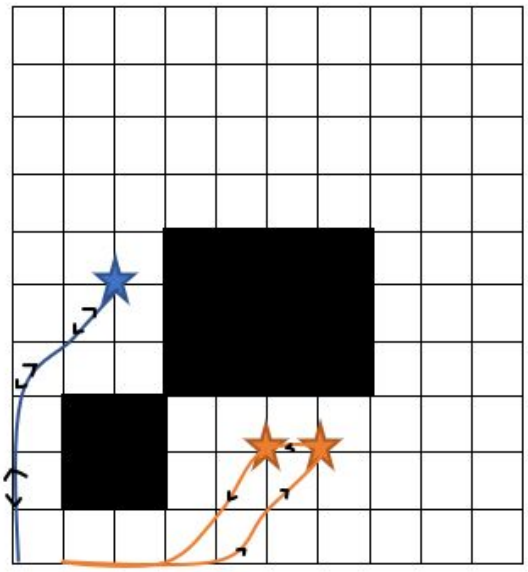

Figure 4: Possible solution for part 5. Again, the orange robot is not shown moving to the origin for clarity.

Input: Format as given above. There will be multiple robots with obstacles.
Output: Format as given above

Part 6

eBotics™ would like to showcase their product to investors, and needs a dashboard and fancy animations to show their robots using their innovative algorithms. Your demonstration video/ screenshots should showcase something interesting about your algorithms.

(Note: you should show this during your presentation!).

Input: Use any input(s) from a previous question or make your own
Output: A video (youtube/ Drive etc. link is sufficient), gif or screenshots

Scoring

Parts 1 and 6 will be graded manually. Parts 2-5 will be auto-graded using the provided script (parser.py). Please make sure your output passes the script or you will receive no points for correctness or speed.

The parser verifies that:

- the max weight (100 kg) is never exceeded
- the robot only takes one step at a time
- all items are collected
- the robot never leaves the field (0,0) to (100, 100) inclusive
- the robot never occupies the same space as another robot. (Note that swapping spaces with another robot is allowed)
- the robot never touches an obstacle. x1 <= x <= x2, y1 <= y <= y2 is never true

Points for correctness will be awarded if the robot(s) collect all of the items and never crash. Full points for speed will be awarded to the team with the fastest solution (by number of timesteps). Other teams will be awarded points based on the following formula:

```
15 * min_timesteps / your_timesteps
```

Note that the fastest team receives a bonus 5 points per test case. In the case of a tie, both/ all teams that are tied will receive the bonus

Part	**Maximum points for correctness**	**Maximum points for speed**
1	30 (across all 3 test cases)	0
2	30	60
3	30	60
4	30	60
5	30	60
Total	*150*	*240*

Part	**Criterion**	**Maximum points**
6	Clarity (It should be visually obvious what is being presented)	25
6	Interactivity (You should be able to vary parameters e.g. number of robots)	25

6	Demonstration choice (The visualization should show something interesting about your algorithm)	50
Presentation	Well-justified (Explanations should be detailed, all with solid reasoning)	50
Presentation	Presentation Quality (Presentation should flow well and be easily followed)	30
Presentation	Choice of Visuals (Visuals contribute to the presentation)	20
Total		*200*

Advice

- Part 1 is useful to ensure you're parsing the input correctly
- Parts 3 and 4 can be worked on independently, but your solution should allow for them to be combined in part 5
- You can submit suboptimal solutions if you don't finish parts
- Part 6 can be worked on independently, and may be useful for debugging other parts

Good luck!