User Manual

Name: Lazy picker

Author: JYZ Team

- Jiati Zhao
- Zeqiang Zheng
- Yukkei He

Version: Alpha 1.0.0

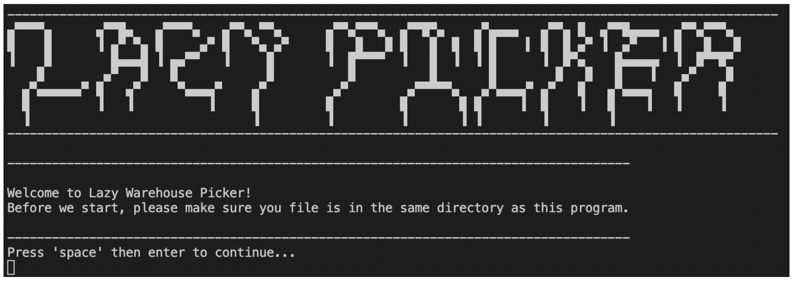
Contents

Glossary:

- UI: User Interface.
- BFS: the breadth-first search algorithm
- DFS: the depth-first search algorithm
- A Star: or A*, a graph traversal and path search algorithm
- Dijkstra: Dijkstra's algorithm, used for finding the shortest paths

Overview

The Lazy Picker is a standalone program designed to assist warehouse workers in efficiently picking products for orders. The program takes the product's unique ID, xLocation, and yLocation from the warehouse's existing database and provides the worker with the location of the product in the warehouse. This program does not require an internet connection and does not rely on a web server.



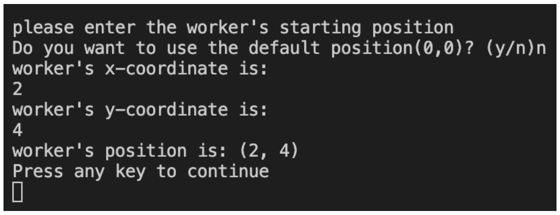
```
--------------------------------------------------------------------------------

Welcome to Lazy Warehouse Picker!
Before we start, please make sure you file is in the same directory as this program.

--------------------------------------------------------------------------------
Press 'space' then enter to continue...
```

Figure 1: Welcome Page

Data initialization & Settings

- Set up the worker's position.

  The starting point can be left by default or be defined by users.

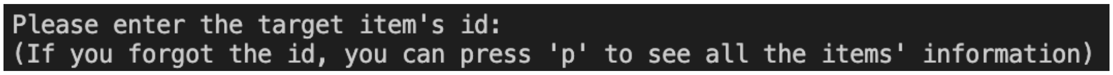


```
please enter the worker's starting position
Do you want to use the default position(0,0)? (y/n)n
worker's x-coordinate is:
2
worker's y-coordinate is:
4
worker's position is: (2, 4)
Press any key to continue
```

Figure 2: Setup worker position

- Input the ID of target items when asked.

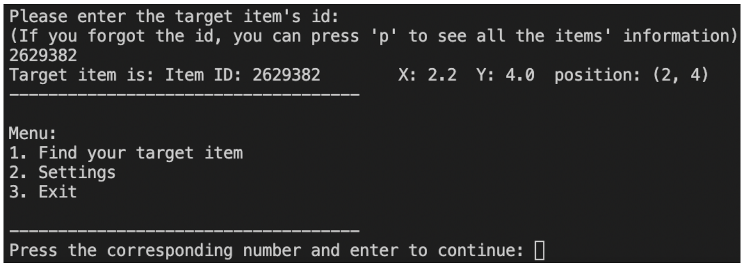
  The current lazy picker program can only support single ID input.



```
Please enter the target item's id:
(If you forgot the id, you can press 'p' to see all the items' information)
```

Figure 3: Product selection

- After the initializing the worker's starting point and the target items, a menu will display for the users to start path searching by pressing '1' or reset the input by pressing '2'.



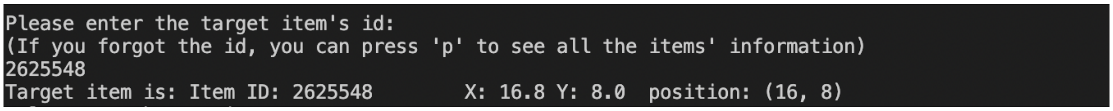
```
Please enter the target item's id:
(If you forgot the id, you can press 'p' to see all the items' information)
2629382
Target item is: Item ID: 2629382       X: 2.2  Y: 4.0  position: (2, 4)
------------------------------------

Menu:
1. Find your target item
2. Settings
3. Exit

------------------------------------
Press the corresponding number and enter to continue:
```

Figure 4: Menu

Features

Basic features

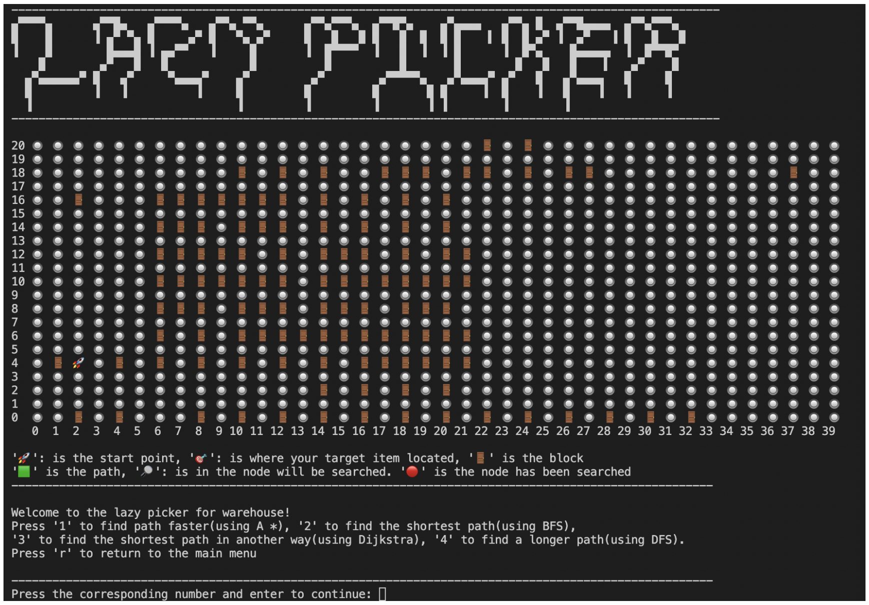
- Enter a Product ID to retrieve shelf number and the product's location in the warehouse.



```
Please enter the target item's id:
(If you forgot the id, you can press 'p' to see all the items' information)
2625548
Target item is: Item ID: 2625548        X: 16.8 Y: 8.0  position: (16, 8)
```

Figure 5: Get item info

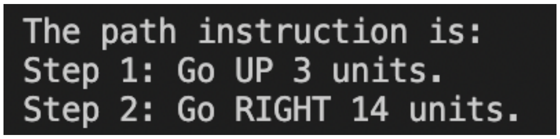
- Command line user interface



```
'🚀': is the start point, '🎯': is where your target item located, '▨' is the block
'🟩' is the path, '🔎': is in the node will be searched. '🔴' is the node has been searched
------------------------------------------------------------------------------------------

Welcome to the lazy picker for warehouse!
Press '1' to find path faster(using A *), '2' to find the shortest path(using BFS),
'3' to find the shortest path in another way(using Dijkstra), '4' to find a longer path(using DFS).
Press 'r' to return to the main menu

------------------------------------------------------------------------------------------
Press the corresponding number and enter to continue: ▯
```

Figure 6: Command line interface

- Descriptive directions:

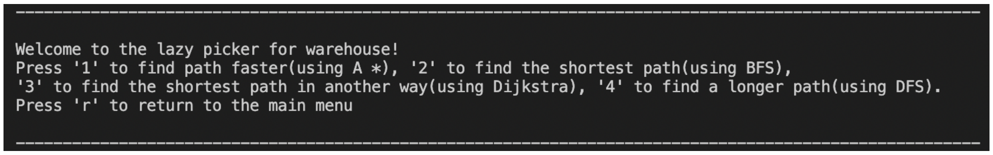


```
The path instruction is:
Step 1: Go UP 3 units.
Step 2: Go RIGHT 14 units.
```

Figure 7: Descriptive directions

- Algorithm selection:

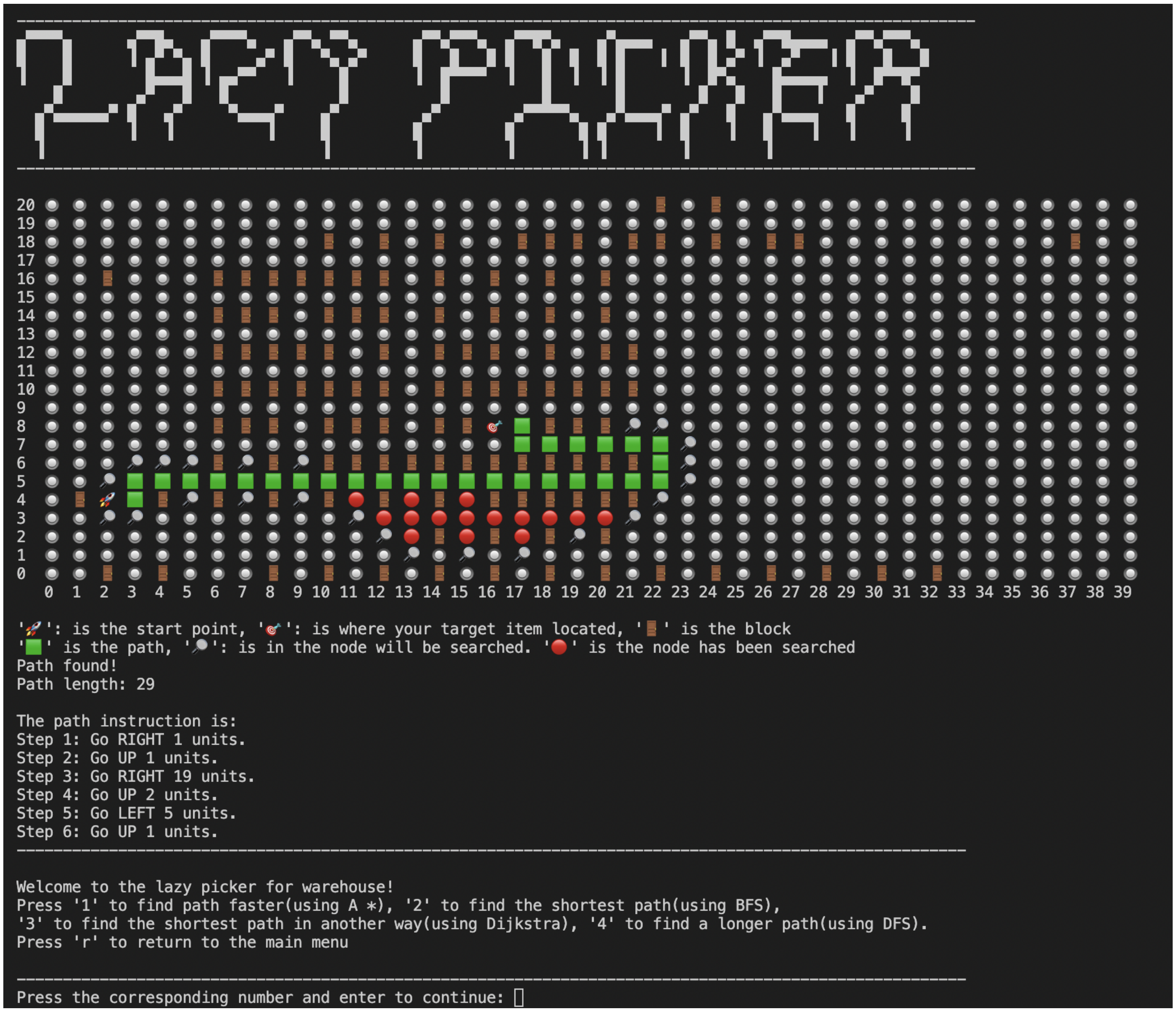
The user can select between different algorithms to find the path.



```
----------------------------------------------------------------------------------------------------

Welcome to the lazy picker for warehouse!
Press '1' to find path faster(using A *), '2' to find the shortest path(using BFS),
'3' to find the shortest path in another way(using Dijkstra), '4' to find a longer path(using DFS).
Press 'r' to return to the main menu

----------------------------------------------------------------------------------------------------
```

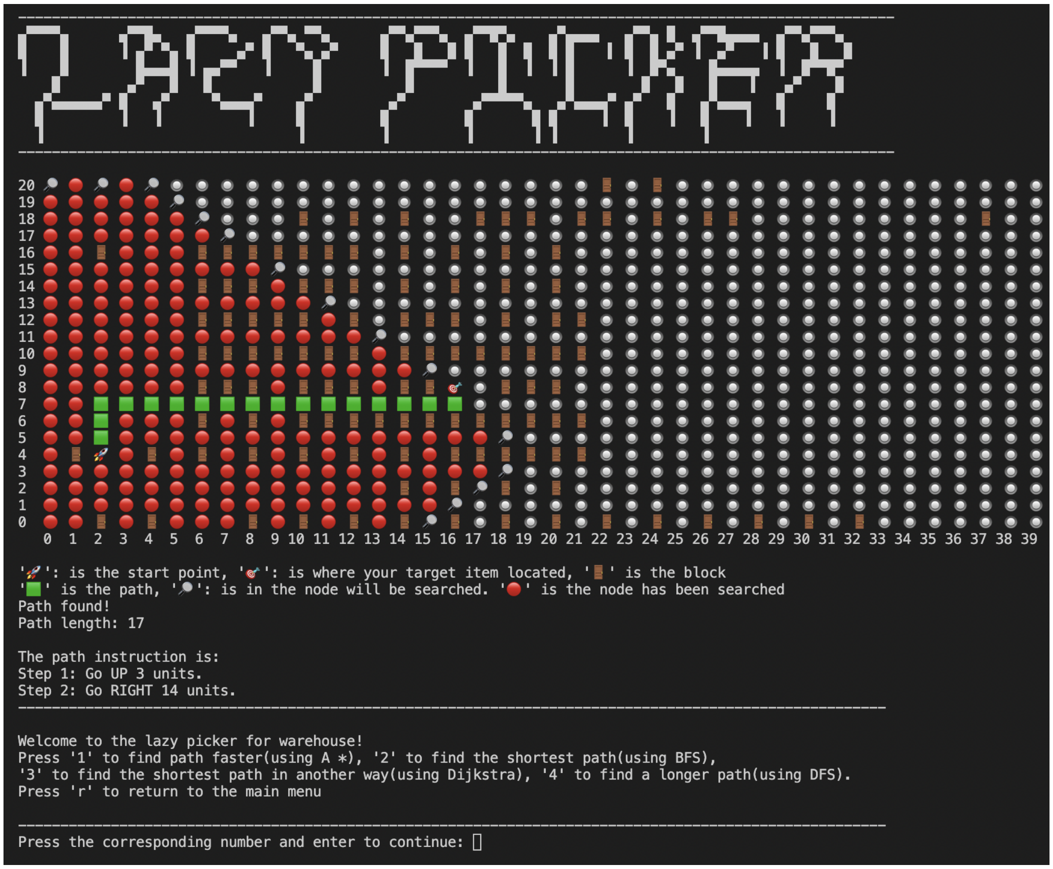
Figure 8: Algorithm selection

For current version, four different algorithms are implemented for the users: A star, BFS, DFS and Dijkstra. Users can choose desired path searching algorithm by pressing corresponding number. The results of different searching algorithms will be displayed as following:



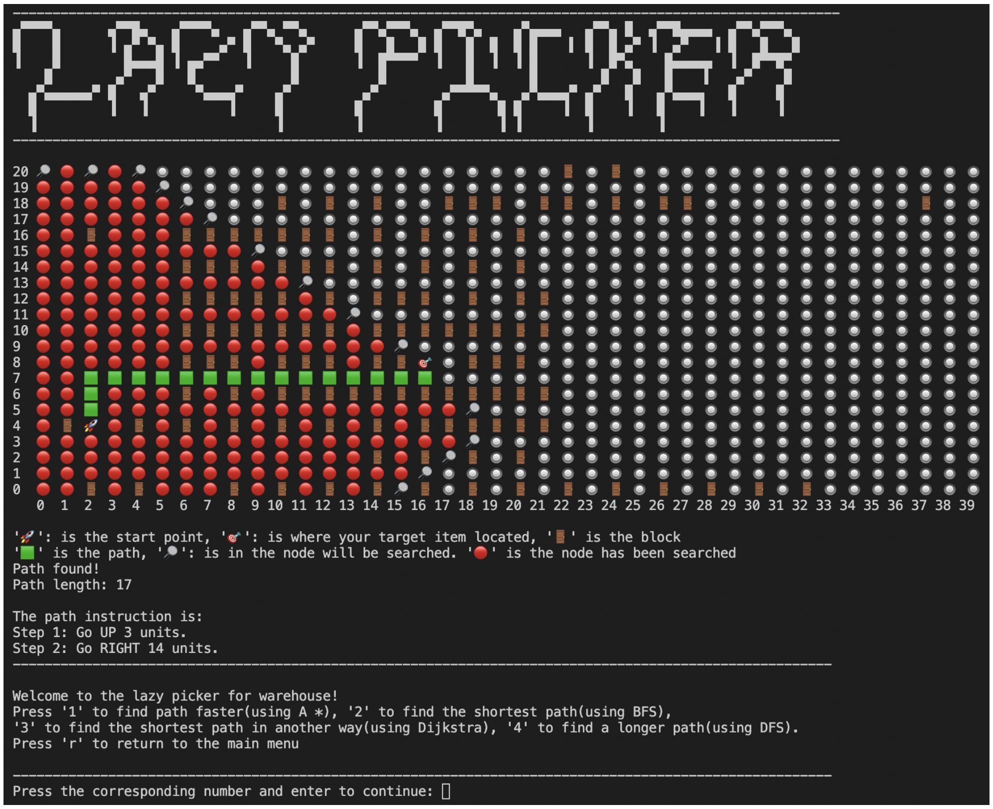
```
'🚀': is the start point, '🎯': is where your target item located, '🟫' is the block
'🟩' is the path, '🔎': is in the node will be searched. '🔴' is the node has been searched
Path found!
Path length: 29

The path instruction is:
Step 1: Go RIGHT 1 units.
Step 2: Go UP 1 units.
Step 3: Go RIGHT 19 units.
Step 4: Go UP 2 units.
Step 5: Go LEFT 5 units.
Step 6: Go UP 1 units.
----------------------------------------------------------------------------------------------------

Welcome to the lazy picker for warehouse!
Press '1' to find path faster(using A *), '2' to find the shortest path(using BFS),
'3' to find the shortest path in another way(using Dijkstra), '4' to find a longer path(using DFS).
Press 'r' to return to the main menu

----------------------------------------------------------------------------------------------------
Press the corresponding number and enter to continue: 
```

Figure 9: result of A star Algorithm



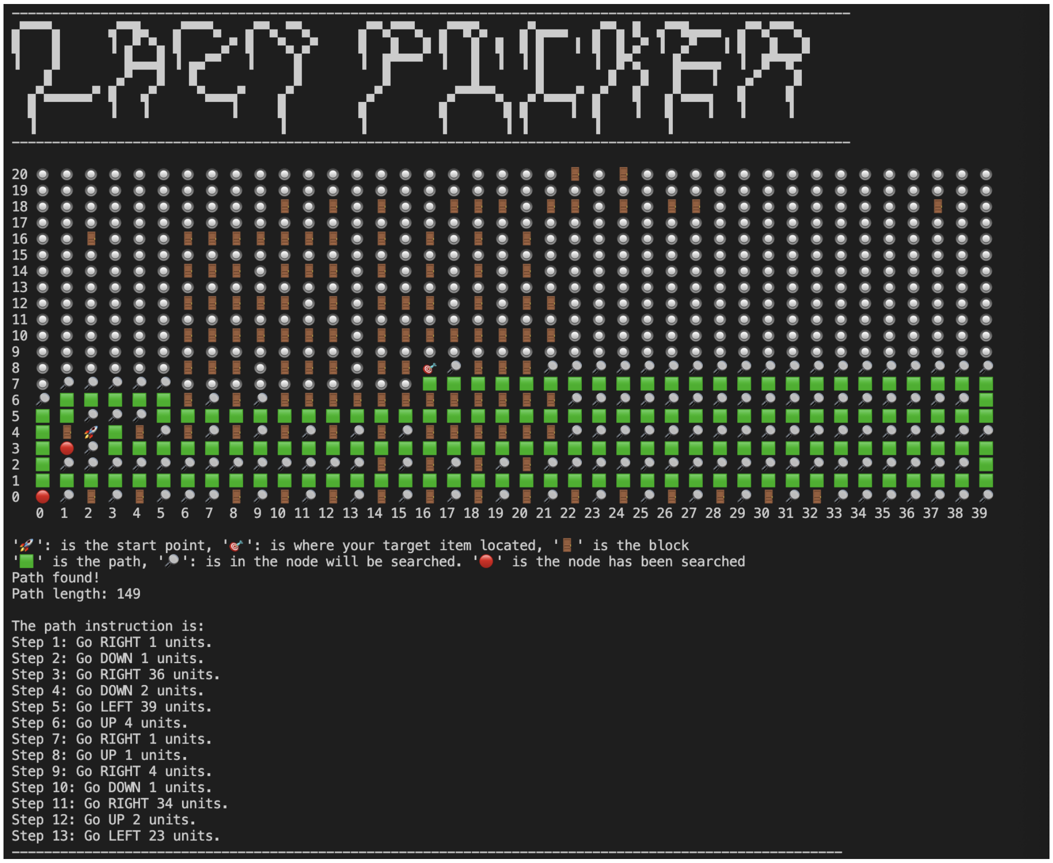
```
'🚀': is the start point, '🎯': is where your target item located, '🟫' is the block
'🟩' is the path, '🔍': is in the node will be searched. '🔴' is the node has been searched
Path found!
Path length: 17

The path instruction is:
Step 1: Go UP 3 units.
Step 2: Go RIGHT 14 units.
--------------------------------------------------------------------------------------------------------

Welcome to the lazy picker for warehouse!
Press '1' to find path faster(using A *), '2' to find the shortest path(using BFS),
'3' to find the shortest path in another way(using Dijkstra), '4' to find a longer path(using DFS).
Press 'r' to return to the main menu

--------------------------------------------------------------------------------------------------------
Press the corresponding number and enter to continue: 
```

Figure 10: result of BFS Algorithm



```
'🚀': is the start point, '🎯': is where your target item located, '🟫' is the block
'🟩' is the path, '🔍': is in the node will be searched. '🔴' is the node has been searched
Path found!
Path length: 17

The path instruction is:
Step 1: Go UP 3 units.
Step 2: Go RIGHT 14 units.
--------------------------------------------------------------------------------------------------------

Welcome to the lazy picker for warehouse!
Press '1' to find path faster(using A *), '2' to find the shortest path(using BFS),
'3' to find the shortest path in another way(using Dijkstra), '4' to find a longer path(using DFS).
Press 'r' to return to the main menu

--------------------------------------------------------------------------------------------------------
Press the corresponding number and enter to continue: 
```

Figure 11: result of Dijkstra Algorithm



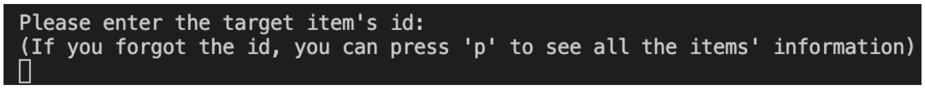
```
'🚀': is the start point, '🎯': is where your target item located, '🚪' is the block
'🟩' is the path, '🔍': is in the node will be searched. '🔴' is the node has been searched
Path found!
Path length: 149

The path instruction is:
Step 1: Go RIGHT 1 units.
Step 2: Go DOWN 1 units.
Step 3: Go RIGHT 36 units.
Step 4: Go DOWN 2 units.
Step 5: Go LEFT 39 units.
Step 6: Go UP 4 units.
Step 7: Go RIGHT 1 units.
Step 8: Go UP 1 units.
Step 9: Go RIGHT 4 units.
Step 10: Go DOWN 1 units.
Step 11: Go RIGHT 34 units.
Step 12: Go UP 2 units.
Step 13: Go LEFT 23 units.
```

Figure 12: result of DFS Algorithm

Advanced Features:

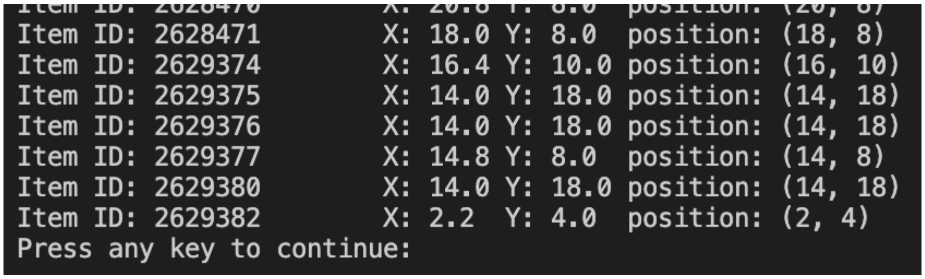
- Product Infomation Display

Users may forget the ID of the desired items. In this case, press 'p' and the product infomation will be shown as the following.



```
Please enter the target item's id:
(If you forgot the id, you can press 'p' to see all the items' information)
```

Figure 13: Ask for Product ID

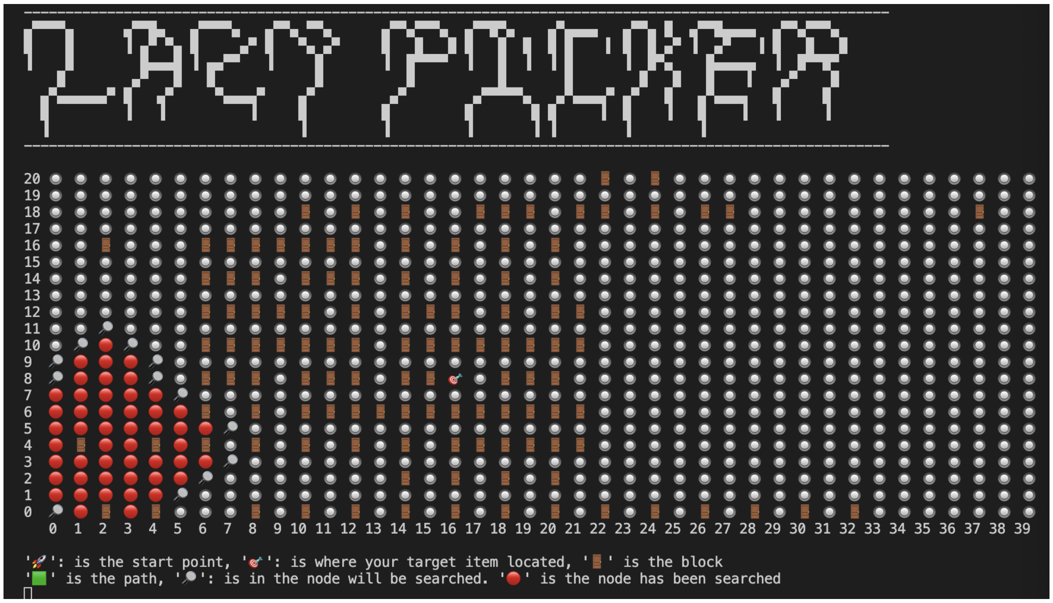


```
Item ID: 2628470        X: 20.8 Y: 8.0  position: (20, 8)
Item ID: 2628471        X: 18.0 Y: 8.0  position: (18, 8)
Item ID: 2629374        X: 16.4 Y: 10.0 position: (16, 10)
Item ID: 2629375        X: 14.0 Y: 18.0 position: (14, 18)
Item ID: 2629376        X: 14.0 Y: 18.0 position: (14, 18)
Item ID: 2629377        X: 14.8 Y: 8.0  position: (14, 8)
Item ID: 2629380        X: 14.0 Y: 18.0 position: (14, 18)
Item ID: 2629382        X: 2.2  Y: 4.0  position: (2, 4)
Press any key to continue:
```

Figure 14: Display Product Infomation

- Path Searching Visualization:

  The process of the path searching is visualized and can be seen by the user. Users can also get to know the path searching progress during their waiting.

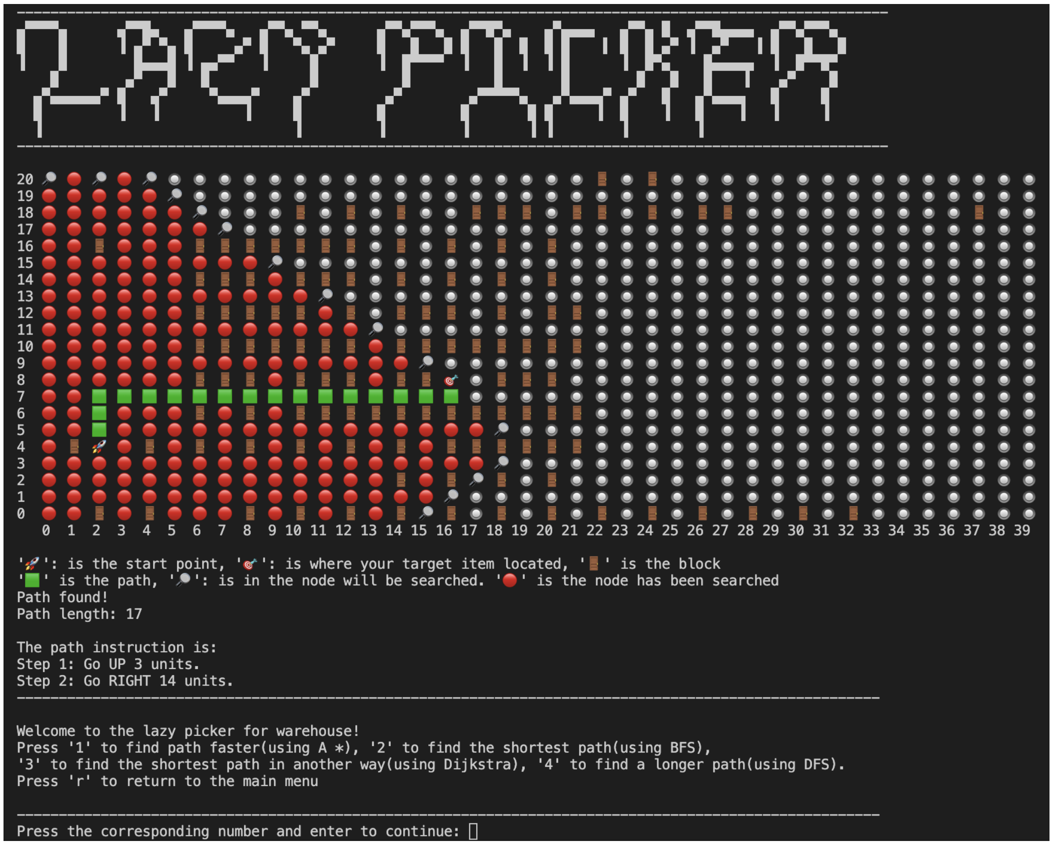


```
'🚀': is the start point, '🎯': is where your target item located, '🟫' is the block
'🟩' is the path, '🔍': is in the node will be searched. '🔴' is the node has been searched
```

Figure 15: Live map display

- Live Map Display & Path Guidance:

  The application displays a live grid that shows the current positions of workers and path taken. Both text and map guidance are displayed for the users to get the selected items.



```
'🚀': is the start point, '🎯': is where your target item located, '🟫' is the block
'🟩' is the path, '🔍': is in the node will be searched. '🔴' is the node has been searched
Path found!
Path length: 17

The path instruction is:
Step 1: Go UP 3 units.
Step 2: Go RIGHT 14 units.
------------------------------------------------------------------------------------------

Welcome to the lazy picker for warehouse!
Press '1' to find path faster(using A *), '2' to find the shortest path(using BFS),
'3' to find the shortest path in another way(using Dijkstra), '4' to find a longer path(using DFS).
Press 'r' to return to the main menu

------------------------------------------------------------------------------------------
Press the corresponding number and enter to continue: 
```

Figure 16: Map Visualization

Potential Future Features (Depends on the future survey results):

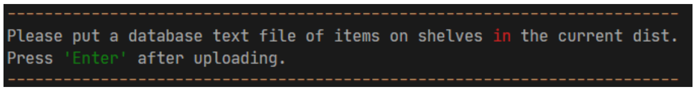
- Debug mode:

  The application provides an optional "debug" mode that displays useful information during execution, including:

  - Current status
  - Start point of the worker
  - Current positions of work and wanted items:

- Increase methods to modify the map setting:

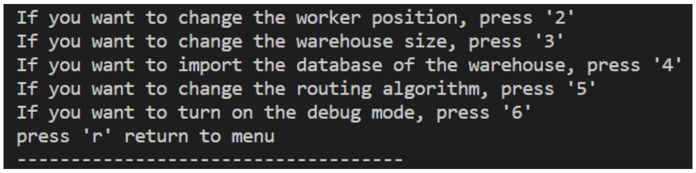
  - Import new warehouse database file:



```
------------------------------------------------------------------------
Please put a database text file of items on shelves in the current dist.
Press 'Enter' after uploading.
------------------------------------------------------------------------
```

Figure 17: Import database file (Sample)

  - Manually enter all data of warehouse:

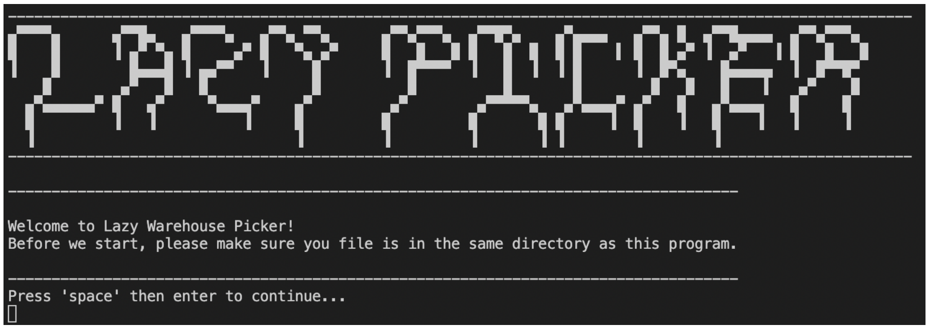


```
If you want to change the worker position, press '2'
If you want to change the warehouse size, press '3'
If you want to import the database of the warehouse, press '4'
If you want to change the routing algorithm, press '5'
If you want to turn on the debug mode, press '6'
press 'r' return to menu
------------------------------------
```

Figure 18: Manul data input (Sample)

Instructions

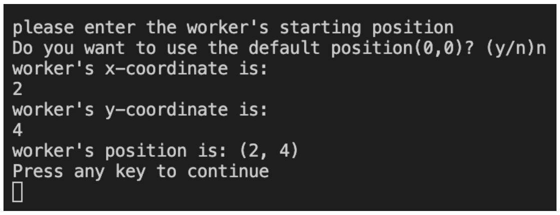
1. Make sure that the warehouse infomation file is under the same folder with the Warehouse Product Picker program, then launch the program.



```
--------------------------------------------------------------------------------

Welcome to Lazy Warehouse Picker!
Before we start, please make sure you file is in the same directory as this program.

--------------------------------------------------------------------------------
Press 'space' then enter to continue...
```

Figure 19: Welcome Page

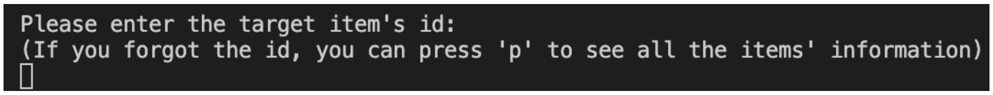
2 Set up the worker's starting position. it is allowed to leave it to the default.



```
please enter the worker's starting position
Do you want to use the default position(0,0)? (y/n)n
worker's x-coordinate is:
2
worker's y-coordinate is:
4
worker's position is: (2, 4)
Press any key to continue
```

Figure 20: Setup worker position

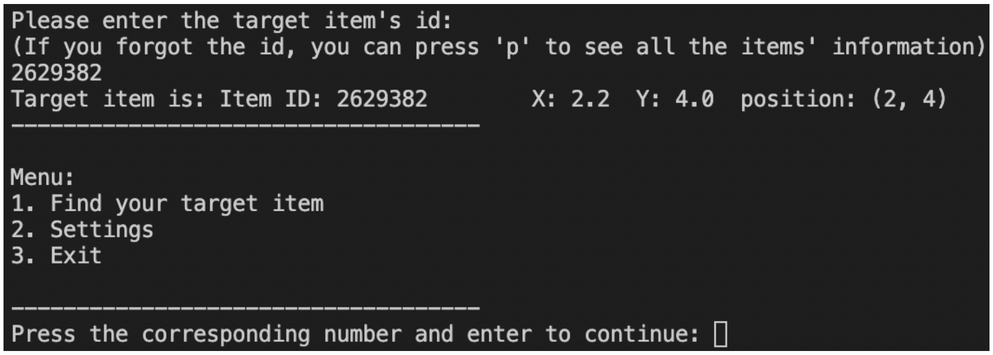
3 Enter the Product ID of the product you are looking for. The product infomation list can be displayed by pressing 'p' if users forget the id.



```
Please enter the target item's id:
(If you forgot the id, you can press 'p' to see all the items' information)
```

Figure 21: Ask for Product ID

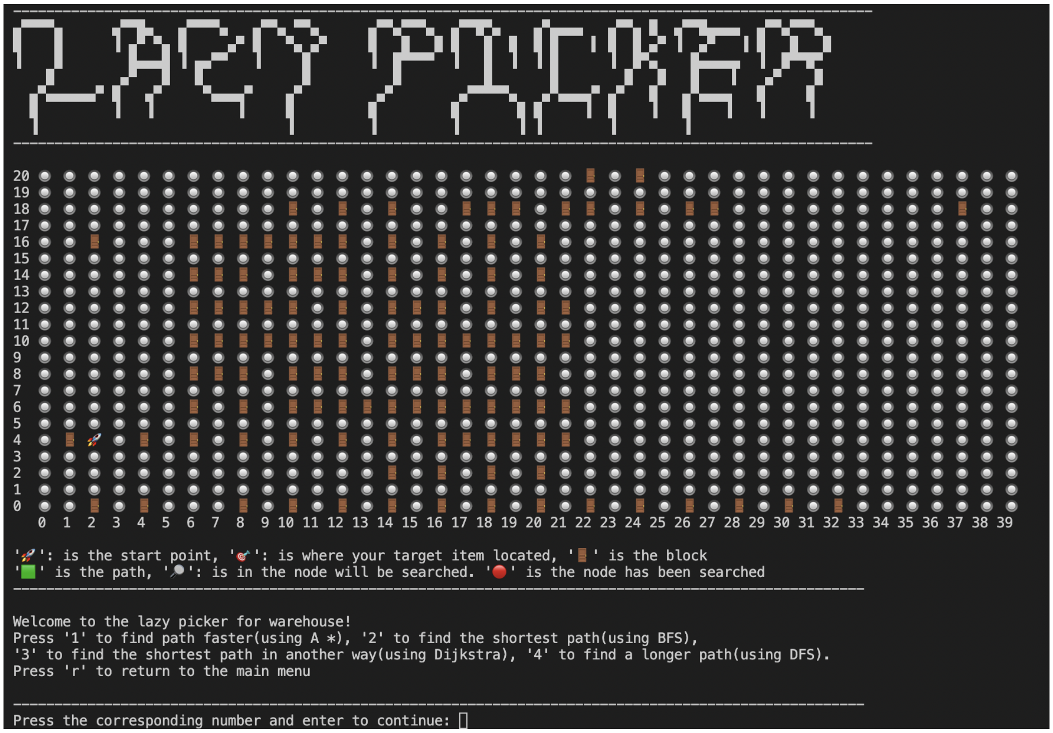
4 The program will display the location of the product on the screen, including the shelf number and the x and y coordinates of the shelf. A menu will be displayed for the user to go to the settings page or to the path searching.



```
Please enter the target item's id:
(If you forgot the id, you can press 'p' to see all the items' information)
2629382
Target item is: Item ID: 2629382       X: 2.2  Y: 4.0  position: (2, 4)
------------------------------------

Menu:
1. Find your target item
2. Settings
3. Exit

------------------------------------
Press the corresponding number and enter to continue:
```

Figure 22: Menu

5 Choose desired algorithm to start the path searching. Please be patient and wait for some time to get the path guidance. Then go and pick the product!



```
'🚀': is the start point, '🎯': is where your target item located, '▨' is the block
'🟩' is the path, '🔎': is in the node will be searched. '🔴' is the node has been searched
------------------------------------------------------------------------------------------

Welcome to the lazy picker for warehouse!
Press '1' to find path faster(using A ∗), '2' to find the shortest path(using BFS),
'3' to find the shortest path in another way(using Dijkstra), '4' to find a longer path(using DFS).
Press 'r' to return to the main menu

------------------------------------------------------------------------------------------
Press the corresponding number and enter to continue: 
```

Figure 23: Choose Desired Algorithm

Installation

System Requirements

To use the program, you need to have

- A working Python 3 environment installed on your computer. This can be downloaded and installed from the official Python website (https://www.python.org/downloads/).
- 2 GB of RAM
- 50 MB of free disk space

Setup & config

- Install form the source code
  1. Download the program files from the source.
  2. Extract the files to a directory of your choice.
  3. Open a terminal or command prompt and navigate to the directory where the program files are located.
  4. Make sure that the warehouse product information file is under the same folder as the lazy_picker program, than run the program by typing the following command:

```shell
python lazy_picker.py
```

Uninstalling

- If your computer has a graphical user interface, simply click the `delete` button as you would when deleting a normal folder.
- If you insist to use the command line, navigate to the directory where the `lazy_picker` folder is located and execute the following command:
  - For Linux user

    ```shell
    rm lazy_picker
    ```

  - For Windows server

    ```shell
    del lazy_picker
    ```

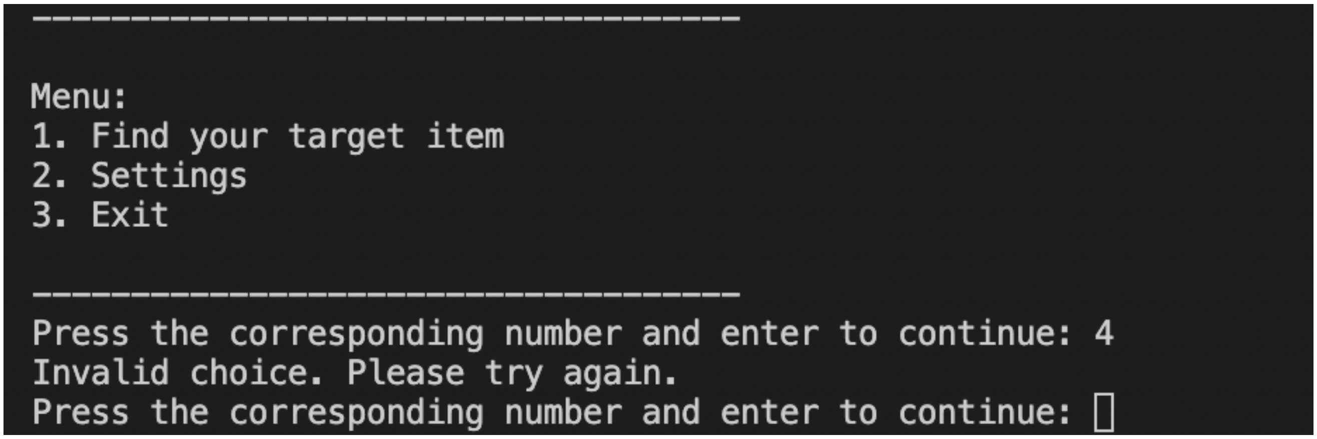
Copyright

Troubleshooting

This section provides solutions for common issues that you may encounter while using our program.

Error handling 1: Input error handling

The application has built-in error handling to prevent the user from entering invalid input. If invalid input is entered, the application will provide an error message indicating the issue.



```
-----------------------------------

Menu:
1. Find your target item
2. Settings
3. Exit

-----------------------------------
Press the corresponding number and enter to continue: 4
Invalid choice. Please try again.
Press the corresponding number and enter to continue: 
```

Figure 24: Invalid Input

Issue 1: Program is not responding after starting path searching

This may be due to the efficiency of the algorithm for finding paths. To prevent this, you could:

1. Avoid choosing the brute force algorithm such BFS and DFS when the warehouse size is large.
2. Grab a coffee and wait patiently.
3. Restart the program.

Issue 2: Program is freezing

If the program is freezing, please try the following:

1. Check that there are not too many other applications running in the background.
2. Restart the program.
3. Try updating the program to the latest version.

Further questions

If you had further questions or constructive suggestions or found bugs regarding our program, feel free to contact us or simply simply create an issue : in our official github repo: https://github.com/yuk-kei/EECS216/tree/main/lazy_picker