



Minesweeper

Spring 2021, CS171

Created By : Sakshi Agarwal (sakshia1@uci.edu), Mike Heddes (mheddes@uci.edu), Sadeem Al Sudais (salsudai@uci.edu)

This is a tutorial to get you started with your Minesweeper Project. This document contains the steps to set up the environment on your systems before implementing your algorithms and certain pointers on how to run your codes. This will help you get accustomed to the code, so you could get started with thinking about the different approaches to solve the problem. Another document regarding team formulations, grading etc will be disclosed soon.

Student Questions

Firstly, if you have any questions regarding setting-up the environment, this document should help you solve some problems. Try to use the outline and search(Ctrl+F) to find the answer.

If there is no answer in this document, you should go to the piazza and click [project](#). Before you post any questions, you should first use the search bar to find existing questions related to your question. If there are no answers, then you do a new post. If the question is good and not in this document and piazza before, your question will be marked as good and you will receive credit for it.

PART 1

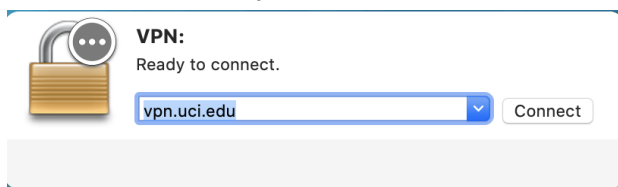
Step 1- Set Environment

VPN

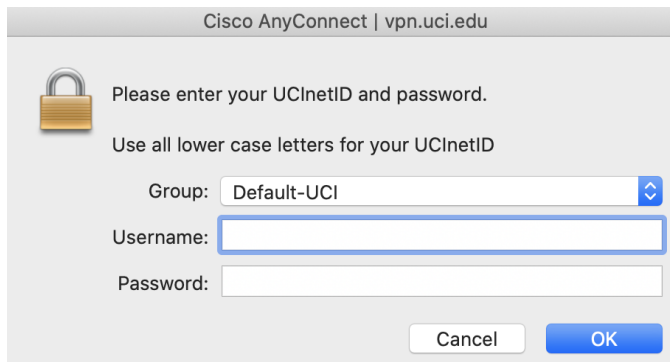
1. Go to <https://www.oit.uci.edu/help/vpn/>.

Download, install and configure the Software VPN Client

- [macOS](#) (**Recommended Version 10.13 High Sierra or newer** - see below for more details)
 - [Windows](#)
 - [Linux](#)
2. Click the version that matches your system.
 3. Download and install the vpn.
 4. Start the VPN and type vpn.uci.edu and click connect.



5. Enter you UCInetID and password



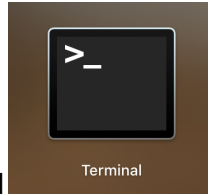
6. Click OK to connect

Openlab

Important : Before you use openlab, you need to have it active your ICS account
https://www.ics.uci.edu/~lab/students/acct_activate.php

- Mac

On Mac you will use terminal to connect to openlab



1. Start Terminal

2. After \$ type 'ssh yourUCnetID@openlab.ics.uci.edu'

```
$ ssh yourUCnetID@openlab.ics.uci.edu
```

3. Type your password, it is invisible, so just type and hit 'return'

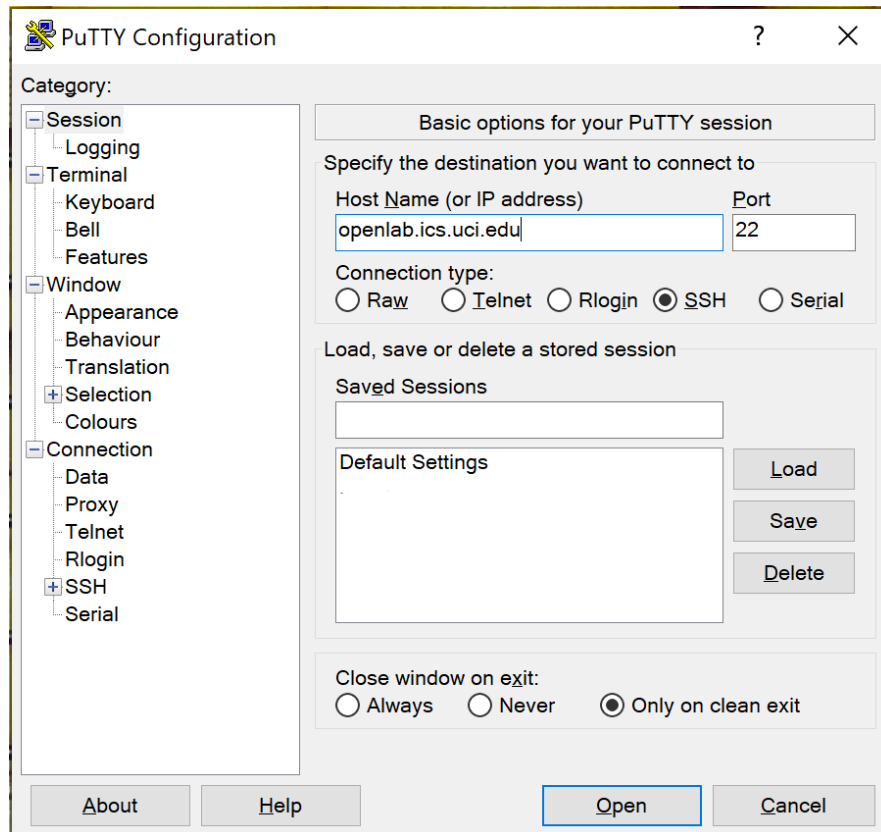
```
password: 
```

4. After you see something like this, you succeed connect to the openlab

```
##### MOTD #####
#
# IMPORTANT NOTES to users of the openlab cluster.
#
# * Please follow important announcements here:
# *
# *   https://swiki.ics.uci.edu/doku.php/announce:spring-2019
#
Any additional queries should be directed to <helpdesk@ics.uci.edu>.
##### MOTD #####
```

- Windows

1. Start your PuTTY software
2. Under hostname enter **openlab.ics.uci.edu**
Under port enter **22**



3. Press open to login
4. The following screen will show up
 - a. Follow the prompt
 - i. First enter your UCINetId
 - ii. Then enter your ICS account password (NOT THE ONES YOU USE TO LOGIN TO VPN unless you set them to be same password)
 - b. The following screen indicates they you have logged in successfully

```
xinyah@circinus-44:~
login as: net1d
net1d@openlab.ics.uci.edu's password:
Last login: Thu Feb 27 02:41:35 2020 from vcv065170.vpn.uci.edu
##### MOTD #####

You have logged into circinus-44.ics.uci.edu
  Operating System: CentOS Linux release 7.7.1908 (Core)
  Architecture:    x86_64
  RAM:             90.80 GiB/94.24 GiB
  Uptime:          9 days

- Please check for ICS System regularly for system announcements

  http://support.ics.uci.edu/ics.system

- Recommended rc files are located in /opt/local/etc/skel

- Use the `module` command to configure your environment

Additional ICS and campus computer resources can be found at the following
websites:

  https://swiki.ics.uci.edu
  https://support.ics.uci.edu
  https://www.oit.uci.edu

##### MOTD #####
#
# IMPORTANT NOTES to users of the openlab cluster.
#
# * Please follow important announcements here:
# *
# *   https://swiki.ics.uci.edu/doku.php/announce:spring-2019
#
Any additional queries should be directed to <helpdesk@ics.uci.edu>.
##### MOTD #####

Visit the following URL for information about the ICS Vagrant
Compute Cluster:

  https://swiki.ics.uci.edu/doku.php/virtual_environments:inst

net1d@circinus-44 22:48:19 ~
$
```

Step 2-Get files

Put file on openlab

In your openlab shell

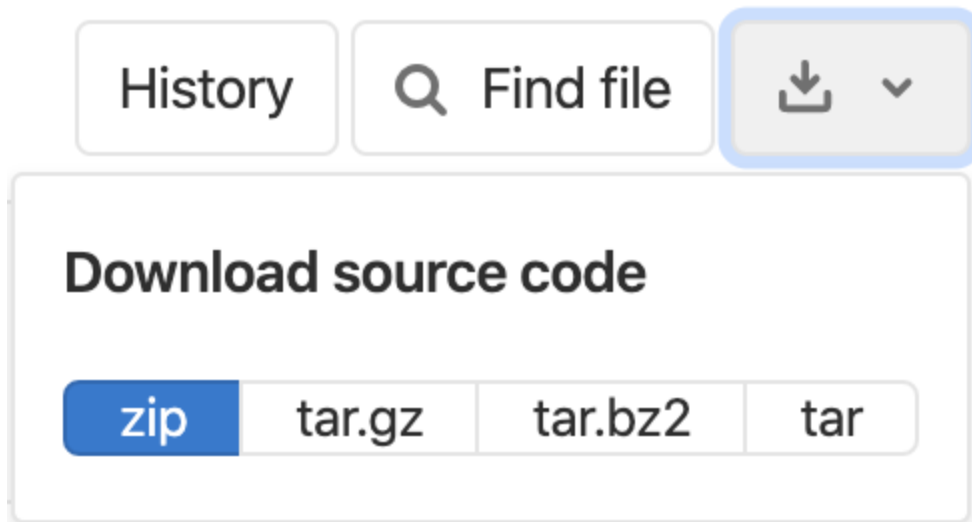
1. Type 'git clone https://gitlab.ics.uci.edu/ai-projects/Minesweeper_Student.git'
2. Type 'cd Minesweeper_Student/' to get into the folder

Get file local

You may want to work on local

1. Go to https://gitlab.ics.uci.edu/ai-projects/Minesweeper_Student

2. Find the download button and click the zip



3. Unzip the zip file

//download directly from gitlab

Transfer file to openlab

If you want to work local, you need to use special application to transfer your files to openlab and download from openlab

Remember, before transfer, you need to connect to uci VPN

We recommend you use Cyberduck, you can use what you prefer

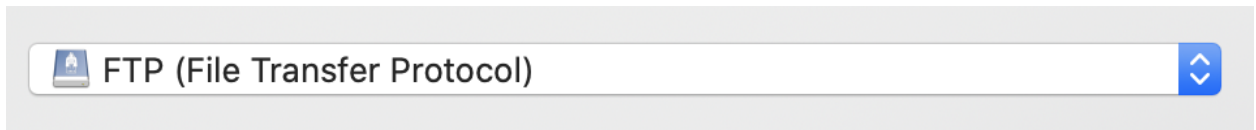


Cyberduck

1. Go to <https://cyberduck.io/>
2. Download Cyberduck
3. Open Cyberduck
4. Find Open Connection



5. Click the dropdown box

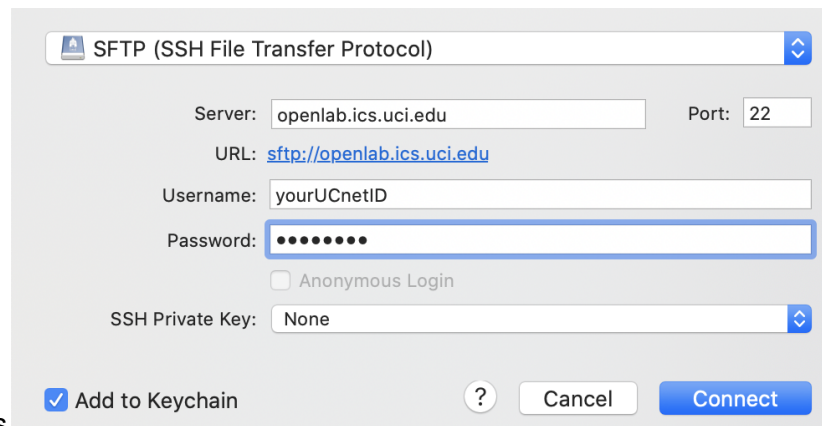


A screenshot of a file transfer protocol selection interface. It features a dropdown menu with a small icon of a folder and a document on the left, the text "FTP (File Transfer Protocol)" in the center, and a blue arrow icon on the right. The entire interface is set against a light gray background.

6. Find and click SFTP(SSH)

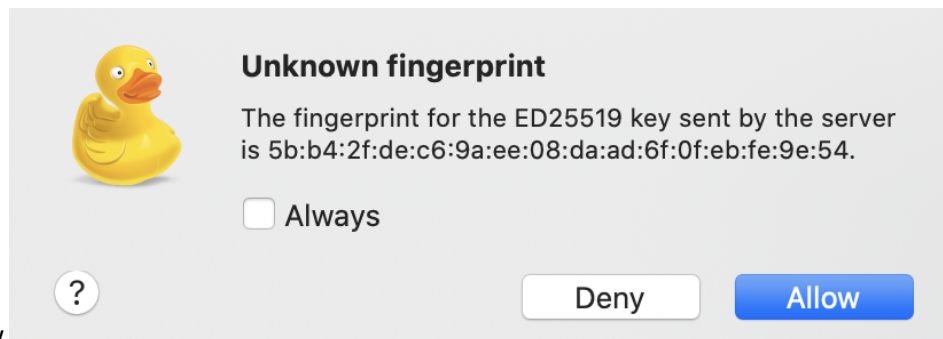


A screenshot of a file transfer protocol selection interface. It features a dropdown menu with a small icon of a folder and a document on the left, the text "SFTP (SSH File Transfer Protocol)" in the center, and a blue arrow icon on the right. The entire interface is set against a light gray background.



A screenshot of the SFTP connection configuration dialog box. It has a title bar that says "SFTP (SSH File Transfer Protocol)". Inside, there are fields for "Server:" (openlab.ics.uci.edu), "Port:" (22), "URL:" (sftp://openlab.ics.uci.edu), "Username:" (yourUCnetID), and "Password:" (masked with dots). There is an "Anonymous Login" checkbox and an "SSH Private Key:" dropdown (set to None). At the bottom, there is a checked "Add to Keychain" checkbox, a help icon, and "Cancel" and "Connect" buttons.

7. Type like this
Click Connect



A screenshot of the "Unknown fingerprint" warning dialog box. It features a yellow duck icon on the left. The text reads: "Unknown fingerprint", "The fingerprint for the ED25519 key sent by the server is 5b:b4:2f:de:c6:9a:ee:08:da:ad:6f:0f:eb:fe:9e:54.", and an "Always" checkbox. At the bottom, there is a help icon, "Deny" and "Allow" buttons.

8. Click Allow
9. Find the folder, you just need to click and drag the item to transfer your files

Filename	Size	Modified
▶ Minesweeper_Student		-- Today, 6:49 PM

Step 3-Program

You should modify the code inside the MyAI file. You should not change the MyAI class interface (the member functions that are used by the World class to interact with your AI). Other than that you are free to modify the code as you wish - e.g. add member variables, add other functions, etc. You can include standard header files whatever you need.

To get some idea of how to write the code, you can view the code inside RandomAI. You also should read and explore the code written in other files like World.

Step 4-Compile and run

You will use Makefile to compile the file.

Python

In Minesweeper_Student-master/Minesweeper_Python

1. Type 'cd /src'
2. Type 'python3 Main.py' to test your AI can run

Java

In Minesweeper_Student-master/Minesweeper_Java

1. Type 'make'
2. Type 'cd bin/'
3. Type 'java -jar mine.jar' to test your AI can run

C++

In Minesweeper_Student-master/Minesweeper_Cpp

1. Type 'make'
2. Type 'cd bin/'
3. Type './Minesweeper' to test your AI can run

Step 5-Test

You need to test your code before submitting.

The first thing you need to learn is '**Shell Manual**' (Part 2 below).

The second thing you need to learn is how to use script to create worlds. You need to check '**World Generator Manual**' (Part 2 below). We recommend you use bash script to create the world in an easy way.

Please use provided options to do the test.

There are several steps you need to consider.

1. First test with default
2. Use script to create worlds
3. Test worlds with -f

Step 6-Submit

You will use Makefile to zip your code to submit

This step is similar to '**Step 4-compile**'

In directory

1. Type 'make submission'
2. Follow the step to fill the questions
3. Transfer your zip file from openlab to local
4. Submit through Canvas

Shell Manual

This next section of the tutorial will teach you how to get started with running your codes using different options - like manual mode and then teach you how to work with the WorldGenerator folder.

Note : For python, use python3 Main.py instead of python3 Main.pyc in the following.

Synopsis

[Name] [Options] [InputFile] [OutputFile]

Name

The command line name used to invoke this program will change depending on the shells:

python3 Main.pyc if using python shell

java -jar mine.jar if using java Shell

./Minesweeper if using cpp shell

Options

-m Use the ManualAI instead of MyAI. If both -m and -r specified, ManualAI will be turned off.

-r Use the RandomAI instead of MyAI.

-d Debug mode, which displays the game board after every move.

-v Verbose mode, which displays name of world files as they are loaded.

-f Depending on the InputFile format supplied, this operand will trigger program **1)** Treats the InputFile as a folder containing many worlds. The program will then construct a world for every valid world file found. The program to display total score instead of a single score. The InputFile operand must be specified with this option **2)** Treats the inputFile as a file. The program will then construct a world for a single valid world file found. The program to display a single score.

Operands

[InputFile]: A path to a valid Minesweeper World file, or folder with -f. This operand is optional unless used with -f or OutputFile.

[OutputFile]: A path to a file where the results will be written. This is optional.

Examples

You can change "python 3 Main.pyc" to other [name] depending on the language you use

`python3 Main.pyc` Constructs a random 8x8 with 10 mines world, runs the MyAI agent on the world, and prints output to console.

`python3 Main.pyc -m` Constructs a random 8x8 with 10 mines world, runs the ManualAI agent on the world, and prints output to console.

`python3 Main.pyc -d` Constructs a random 8x8 with 10 mines world, runs the MyAI agent on the world, and prints output to console. After every turn, the game pauses and prints the current game state to the console.

`python3 Main.pyc -r` Constructs a random 8x8 with 10 mines world, runs the RandomAI on the world, and prints output to console.

`python3 Main.pyc -rd` Constructs a random 8x8 with 10 mines world, runs the RandomAI agent on the world using debug mode, and prints output to console. After every turn, the game pauses and prints the current game state to the console.

`python3 Main.pyc -f /path/to/world/file.txt` Constructs the world specified in the file, runs the MyAI agent on the world, and prints output to console.

`python3 Main.pyc -f /path/to/world/files/` Constructs all the worlds specified in the folder, runs the MyAI agent on all the worlds, and prints output to console.

`python3 Main.pyc -f /path/to/world/files/
/path/to/outputfile/yourscores.txt` Constructs all the worlds specified in the folder, runs the MyAI agent on all the worlds, and write output to txt file.

`python3 Main.pyc -fv /path/to/world/files/`

Constructs all worlds specified in the folder, runs the MyAI agent on all the worlds, and prints output to console. Before running every world, print a message indicating which specific world is running.

Notes

The Python shell uses Python version 3.5.2. When using debug mode or ManualAI, the board will be printed to the console. Each tile is represented as a full stop potentially followed by a series of characters. Every board that gets displayed starts with 1-indexing and bottom left.

World Generator Manual

***NOTE:** Before you go ahead creating your own custom boards and world files, create a folder called “Problems” inside the “WorldGenerator” folder. Any worlds you create using either the given Python script or bash scripts will automatically put the generated worlds inside this folder and refresh its contents with the newly created files.

The Minesweeper World File

If you’d like to create your own custom Minesweeper World, you can manually create a text file in the following format:

[rowDimension][space][colDimension]

[startingRow][space][startingColumn]

[2D grid of board]

- “StartingRow” and “startingColumn” represent the coordinates of the first tile that the world uncovers for you. This feature is designed to guarantee your agent will be safe on its’ starting tile.

- The 2D grid should be a sequence of 0’s and 1’s. Columns are separated by a single space while rows are separated by a new line. 0’s represent safe tiles while 1’s represent tiles with mines. The bottom left-most tile should be interpreted as (1, 1). When selecting starting tile, the tiles around it will always be safe, namely 0.

- If your 2D grid does not contain at least a 3x3 square of 0's (because the starting tile must be a safe tile), the world is considered invalid.

Using the Python script

- There is a Python script "WorldGenerator.py" that you can use to easily generate a set of worlds in the form of txt files.
- If you see "Error opening file" in your console when running the script, that means you haven't create a folder called "Problems" in the directory you are running this script, hence the script fails to locate.
- To run the script, issue the command:

```
python3 WorldGenerator.py [numFiles] [filename] [rowDimension] [colDimension]  
[numMines]
```

The arguments in square brackets are in that order and represent the following:

numFiles	- The number of files to generate
filename	- The base name of the file
rowDimension	- The number of rows
colDimension	- The number of columns
numMines	- The number of mines

Note that all arguments are required and have certain restrictions, which are listed below:

- (1) The minimum number of rows is 4.
- (2) The minimum number of columns is 4.
- (3) The minimum number of mines is 1.
- (4) The number of mines must also be less than or equal to

$$(\text{rowDimension}) * (\text{colDimension}) - 9$$

If any of these conditions are not met, the script will not generate any worlds. Another thing to note is that you can only generate a set of worlds of the same dimensions. In order to generate worlds of different sizes, you need to rerun the script with different command-line arguments

Using the bash script

- Another way to generate worlds is to use the bash script. There are two bash scripts provided. You can easily change the script to generate a different number of worlds or worlds of different dimensions. You may also write your own bash script.

- To run the scripts, issue the command:

```
./generateTournament
```

This script will be used to generate the tournament set of worlds and is a good simulation for your agent's actual performance in the tournament. This script creates a "Problems" folder and generate a random set of 1000 Beginner, 1000 Intermediate, and 1000 Expert worlds. Note that this is not the actual number of worlds used for grading.

```
./generateSuperEasy
```

This script will be used to generate the set of worlds for the Minimal submission. It generates 1000 random Easy worlds.

- if you run into a "permission denied" message, run `chmod +x generateTournament.sh` or `chmod +x generateSuperEasy.sh`

- If you get a "bin/bash^M: bad interpreter" error, use vi or vim to edit the script and type

and press ENTER before saving/exiting the file `:set ff=unix`

Example World

A txt file like this:

88

13

00111010

00000000

00000000

00000000

00100000

00000001

00001010

00100010

will generate (in console):

```
8 | 2  2  0  1  1  1  *  *  *  2  *  1
7 |  0  0  0  0  1  2  3  2  2  1  1
6 |  0  0  0  0  0  0  0  0  0  0  0
5 | 0  0  0  1  0  1  1  0  0  0  0
4 |  0  0  1  1  *  1  0  0  1  1
3 |  0  1  1  1  2  1  2  2  2  *
2 | 1  2  2  1  *  1  2  *  3  *  3
1 | *  3  *  1  3  *  2  1  3  *  2
0 | -  -  -  -  -  -  -  -  -  -
  | 1  3  *  2  2  3  4  5  6  7  8
```

Remember that arrays start with the index 0 while the minesweeper's dimension starts with the index of 1.