

## EXAMPLE: How to generate and update dYdX Trading Rewards reports

Author: Lawrence Chiu ([lawrence@dydx.exchange](mailto:lawrence@dydx.exchange))

Twitter: <https://twitter.com/LawrenceChiu14>

Date: 4/16/2022 (updated 4/19/2022)

Tested on: Ubuntu Server 20.04.4 LTS, Windows 10 (See Appendix A), MacOS 12.3.1 (Monterey) (See Appendix B)

We will generate dYdX Trading Rewards reports with the `update_rewards.sh` shell script. In addition, you will need Merkle tree files from epoch 0 up to the last epoch. Links to these are provided in the dYdX Foundation reviews. For example, for epoch 7, the review is at

<https://dydx.foundation/blog/en/epoch-7> and the link to the Merkle tree file is at

<https://gateway.ipfs.io/ipfs/bafybeiauyvb7mxv3xjuu2wlxojzkyr27nsekkpr52un5hywi26s3ohfs2m>

Each Merkle tree file must be named `epoch<#>.json` where `<#>` is the epoch number.

Epoch 7 Review

Claim your X DYDX rewards

The Merkle root was [proposed](#) on-chain on March 15 at 19:51 UTC and the 7-day waiting period has begun. Epoch 7 rewards will be claimable [here](#) on March 22, at ~19:51 UTC (7 days after the end of the epoch plus a ~5 hour delay). Once tokens have been claimed, they can be transferred, staked to the Safety module, or delegated to dYdX governance.

The Merkle tree data, which is a list of (address, reward) pairs, is available [here](#).

Under the hood, the [Merkle Distributor smart contract](#) will distribute X DYDX token rewards according to a Merkle tree of balances. The tree will be updated at the end of each epoch with each user's cumulative reward balance. An update is performed by setting the proposed Merkle root to the

The program and epoch files (up to epoch 8) are available at

<https://github.com/chiwalfrm/dydxexamples>

## PART 1: The First Time



1. If this is the first time running it, your directory should have just the script and the epoch files.

```
vmware@ubuntu20041a: ~/update_rewards
vmware@ubuntu20041a:~/update_rewards$ ls -al
total 23036
drwxrwxr-x  2 vmware vmware  4096 Apr 16 20:08 .
drwx----- 27 vmware vmware  4096 Apr 16 20:08 ..
-rw-rw-r--  1 vmware vmware 2323419 Dec 31 1969 epoch0.json
-rw-rw-r--  1 vmware vmware 2560115 Dec 31 1969 epoch1.json
-rw-rw-r--  1 vmware vmware 2908914 Dec 31 1969 epoch2.json
-rw-rw-r--  1 vmware vmware 3005265 Dec 31 1969 epoch3.json
-rw-rw-r--  1 vmware vmware 3087872 Dec 31 1969 epoch4.json
-rw-rw-r--  1 vmware vmware 3137100 Dec 31 1969 epoch5.json
-rw-rw-r--  1 vmware vmware 3215228 Dec 31 1969 epoch6.json
-rw-rw-r--  1 vmware vmware 3320103 Dec 31 1969 epoch7.json
-rwxr-xr-x  1 vmware vmware  5534 Apr 16 19:48 update_rewards.sh
vmware@ubuntu20041a:~/update_rewards$
```

2. Note: On Linux, you will be prompted for your password in order to execute a few sudo commands to set up a ramdisk. If you are not comfortable providing your password, remove the sudo commands and run them yourself (before running the script). The commands are:

```
echo "STEP 1: Setting up ramdisk..."
if [ "`uname`" = "Darwin" ]
then
    diskutil erasevolume HFS+ "epochdisk" `hdiutil attach -nomount ram://2097152`
    TMPFOLDER=/Volumes/epochdisk
else
    sudo mkdir -p /mnt/epochdisk
    sudo mount -t tmpfs -o rw,size=1G tmpfs /mnt/epochdisk
    sudo chmod 777 /mnt/epochdisk
    TMPFOLDER=/mnt/epochdisk
fi
mkdir -p $TMPFOLDER/update_rewards$/output
echo "STEP 2: Determining starting and last epoch..."
```

comment these lines out if you want to run them yourself

```
cleanup ()
{
    if [ "`uname`" = "Darwin" ]
    then
        hdiutil detach $TMPFOLDER
    else
        sudo umount $TMPFOLDER
    fi
}

pid=$$
trap 'echo; echo "Ctrl-C detected. Cleaning up..."; kill -TERM $ps h --pid $pid -o pid; cleanup; exit' INT HUP
prepare_epoch_file ()
```

If commented out, run this command at the end when finished (\$TMPFOLDER = /mnt/epochdisk)

1. Start the program. Invoke the program with the <INTENSITY> parameter which has 3 choices: 'low', 'medium', or 'high' (if omitted, it defaults to 'medium'). This parameter determines how many CPU cores to use, as shown below along with a sample of estimated runtimes. Our own testing indicates there isn't much value to using 'high' (200% CPU cores) as opposed to 'medium' intensity.

	A	B	C	D
	CPU	Intensity	CPU usage	Runtime (8 epochs)
	12th Gen Intel(R) Core(TM) i9-12900K (4 cores)	low	50% cores	0:24:09
	12th Gen Intel(R) Core(TM) i9-12900K (4 cores)	medium	100% cores	0:12:45
	12th Gen Intel(R) Core(TM) i9-12900K (4 cores)	high	200% cores	0:11:51
	Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz	low	50% cores	1:57:21
	Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz	medium	100% cores	1:17:57
	Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz	high	200% cores	1:10:47
	Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz	low	50% cores	0:55:02
	Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz	medium	100% cores	0:30:53
	Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz	high	200% cores	0:30:19

```
$ ./update_rewards.sh medium
```

3. This will take some time. You can monitor the progress in another terminal with the command:

```
$ while true; do clear; tail /mnt/epochdisk/update_rewards*/xaa.stdout; sleep 1; done
```

```
vmware@ubuntu20041a: ~/update_rewards
FILEUPDATE 0x9ca7ff02ac1ed76bee086ba168170db21bb9ea5c 2803 / 5843
FILEUPDATE 0x755a8c444a35f87c8217c3fb18392ff38974fde7 2804 / 5843
FILEUPDATE 0x4e4f5c34a12f01e9bd8dbad4756bbf9972d913e5 2805 / 5843
FILEUPDATE 0x7e5dc59111f7e68241927f37c6b93cf50e938158 2806 / 5843
FILEUPDATE 0xbfa78ef9d97d9e06d56f66867f6a0a20a0600020 2807 / 5843
FILEUPDATE 0x606ac4dae71e42bff40a0a09bdc32baad8cb579a 2808 / 5843
FILEUPDATE 0x6e6208b37f371f137e3775a8093fde1311e254c5 2809 / 5843
FILEUPDATE 0xcde99233085bbfddcefcffalba7cb99294e383d4 2810 / 5843
FILEUPDATE 0xdc5c0e196a98e0514a0f6673c528898d05d9257e 2811 / 5843
FILEUPDATE 0x0815c1e34a819f48d480a173db83b58c076d7299 2812 / 5843
```

*This screenshot shows progress at 48%)*

4. The reports are stored in output/ directory. In the same directory you will also find fulllist.html and whalelist.html, with the former containing links to addresses in alphanumeric order, and the latter sorted by total rewards in descending order.



```

vmware@ubuntu20041a: ~/update_rewards
vmware@ubuntu20041a:~/update_rewards$ ./update_rewards.sh
STAGE 1 Setting up ramdisk...
[sudo] password for vmware:
STAGE 2 Determining starting and last epoch...
STAGE 3 Copying existing output/ directory data (if any)...
STAGE 4 Preparing epoch files...
STAGE 5 Generating address lists...
STAGE 6 Generating/updating trading reports...
STAGE 7 Generating parallel workloads...
STAGE 8 Executing parallel jobs...
STAGE 9 Waiting for completion of jobs...
STAGE 10 Finalizing...
[sudo] password for vmware:
vmware@ubuntu20041a:~/update_rewards$ █

```

## PART 2: Subsequent Updates

2. After the first time, updates for later epochs will run faster. Add the new epoch file(s) and run the program again.

```

vmware@ubuntu20041a: ~/update_rewards
vmware@ubuntu20041a:~/update_rewards$ ls -al
total 54940
drwxrwxr-x 3 vmware vmware 4096 Apr 16 21:17 .
drwx----- 27 vmware vmware 4096 Apr 16 20:52 ..
-rw-rw-r-- 1 vmware vmware 2323419 Dec 31 1969 epoch0.json
-rw-rw-r-- 1 vmware vmware 2128676 Apr 16 20:51 epoch0.txt
-rw-rw-r-- 1 vmware vmware 2560115 Dec 31 1969 epoch1.json
-rw-rw-r-- 1 vmware vmware 2344936 Apr 16 20:51 epoch1.txt
-rw-rw-r-- 1 vmware vmware 2908914 Dec 31 1969 epoch2.json
-rw-rw-r-- 1 vmware vmware 2663711 Apr 16 20:51 epoch2.txt
-rw-rw-r-- 1 vmware vmware 3005265 Dec 31 1969 epoch3.json
-rw-rw-r-- 1 vmware vmware 2751860 Apr 16 20:51 epoch3.txt
-rw-rw-r-- 1 vmware vmware 3087872 Dec 31 1969 epoch4.json
-rw-rw-r-- 1 vmware vmware 2827387 Apr 16 20:51 epoch4.txt
-rw-rw-r-- 1 vmware vmware 3137100 Dec 31 1969 epoch5.json
-rw-rw-r-- 1 vmware vmware 2872427 Apr 16 20:51 epoch5.txt
-rw-rw-r-- 1 vmware vmware 3215228 Dec 31 1969 epoch6.json
-rw-rw-r-- 1 vmware vmware 2943853 Apr 16 20:51 epoch6.txt
-rw-rw-r-- 1 vmware vmware 3320103 Dec 31 1969 epoch7.json
-rw-rw-r-- 1 vmware vmware 3039680 Apr 16 20:51 epoch7.txt
-rw-rw-r-- 1 vmware vmware 3483266 Dec 31 1969 epoch8.json
-rw-rw-r-- 1 vmware vmware 2009691 Apr 16 20:51 epochaccounts_sorted_alpha.txt
-rw-rw-r-- 1 vmware vmware 2009691 Apr 16 20:51 epochaccounts_sorted_rewards.txt
drwxrwxr-x 2 vmware vmware 3567616 Apr 16 21:14 output
-rwxr-xr-x 1 vmware vmware 5567 Apr 16 20:42 update_rewards.sh
vmware@ubuntu20041a:~/update_rewards$ █

```

← added

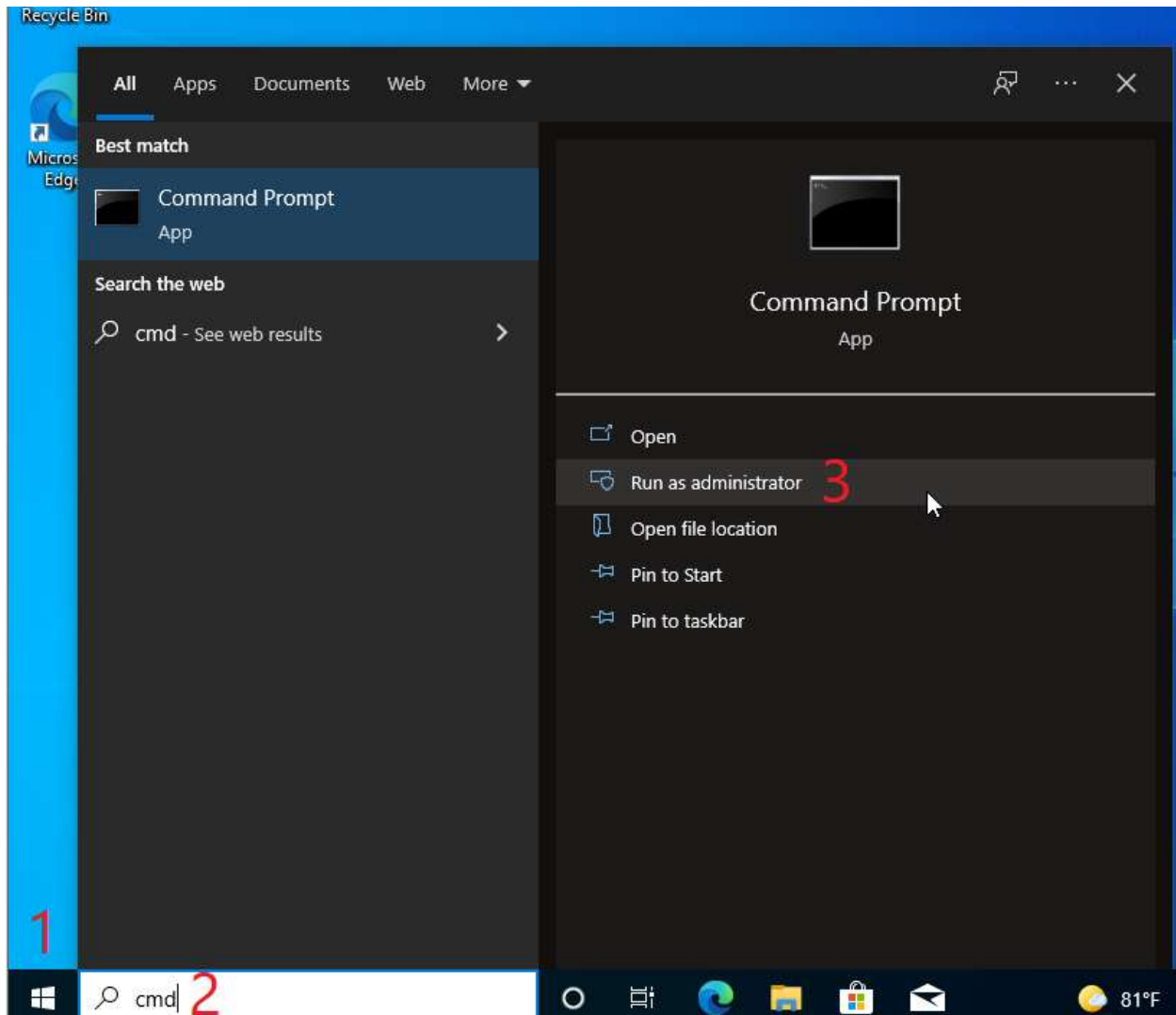
```
$ ./update_rewards.sh medium
```

## APPENDIX A: Windows 10

### (Win10) PART 1: Install Windows Subsystem for Linux (WSL)

#### 1) Start Command Prompt as Administrator:

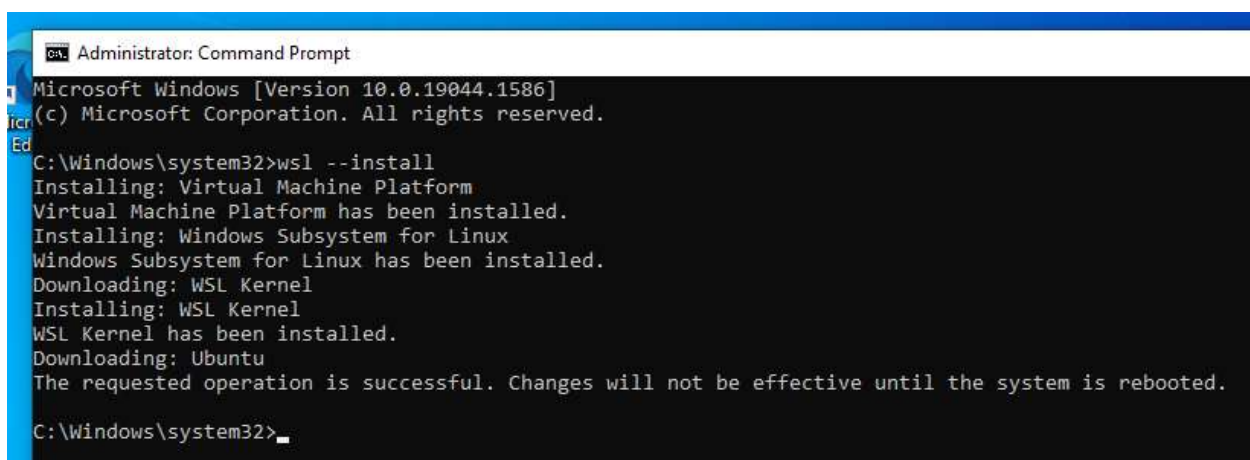
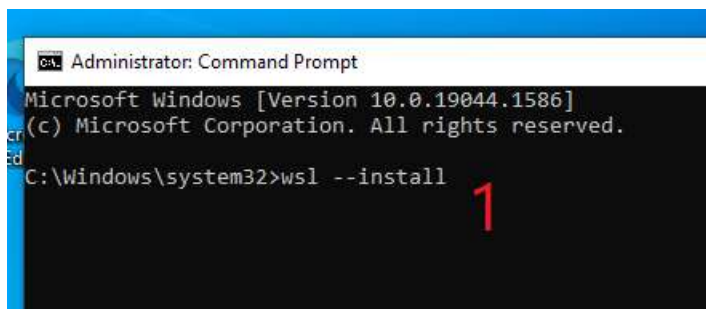
- Click the Windows button.
- Type 'cmd'.
- Click on 'Run as administrator'.



d. Click Yes.



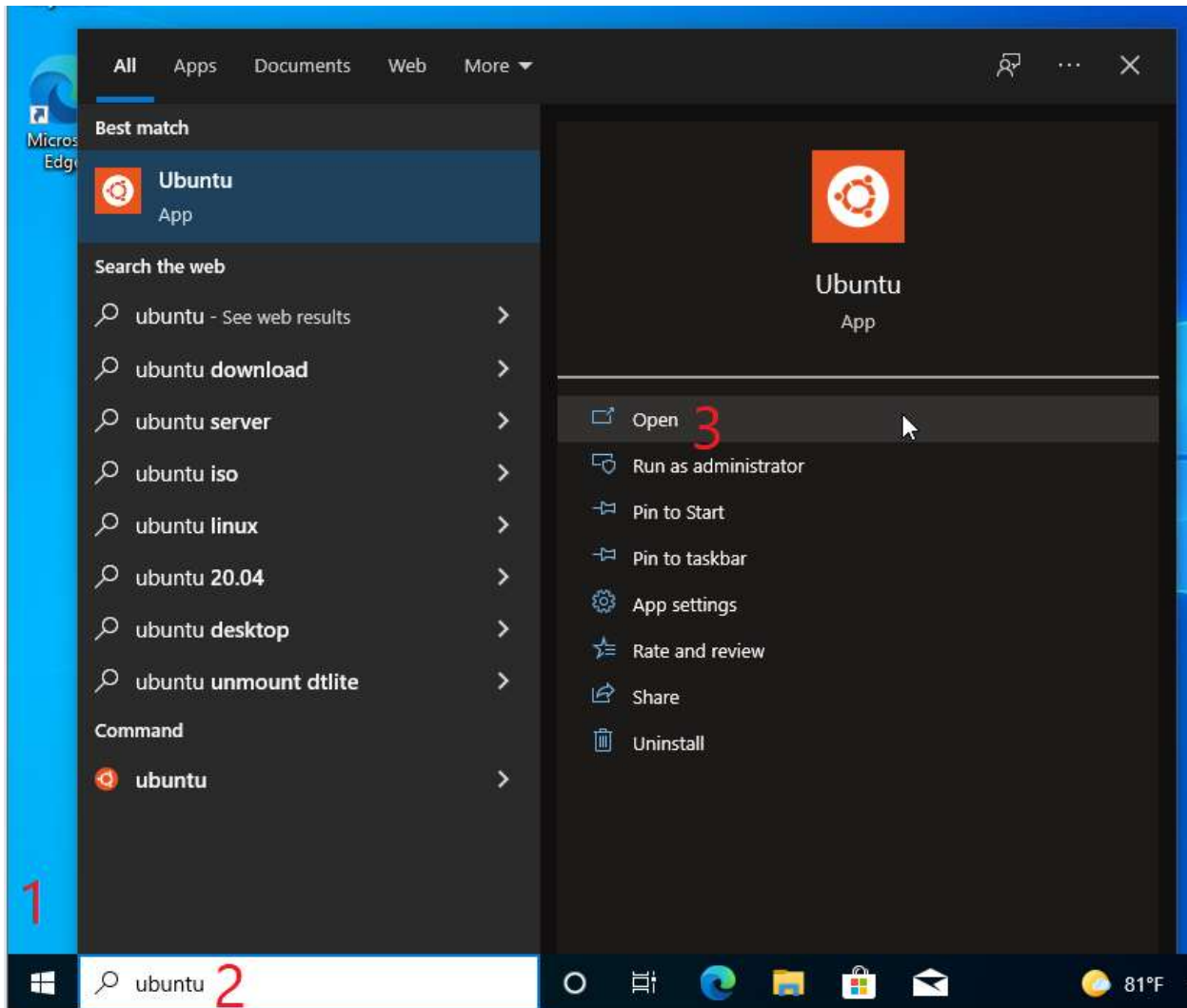
2) Type `wsl --install` and wait.



3) Reboot.

## (Win10) PART 2: Launch WSL

- 1) Start Ubuntu:
  - a. Click the Windows button.
  - b. Type 'ubuntu'.
  - c. Click on 'Open'.



2) (First time) Enter username and password (twice).

```
lawrencedydx@DESKTOP-E67DGDB: ~  
Installing, this may take a few minutes...  
Please create a default UNIX user account. The username does not need to match your Windows username.  
For more information visit: https://aka.ms/wslusers  
Enter new UNIX username: lawrencedydx  
New password:  
Retype new password:  
passwd: password updated successfully  
Installation successful!  
  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.10.16.3-microsoft-standard-WSL2 x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
System information as of Mon Apr 11 18:57:55 CDT 2022  
  
System load:  0.31           Processes:            8  
Usage of /:   0.4% of 250.98GB Users logged in:       0  
Memory usage: 3%           IPv4 address for eth0: 172.30.252.228  
Swap usage:   0%  
  
0 updates can be installed immediately.  
0 of these updates are security updates.  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
This message is shown once once a day. To disable it please create the  
/home/lawrencedydx/.hushlogin file.  
lawrencedydx@DESKTOP-E67DGDB:~$
```

## (Win10) PART 3: Update WSL

1) Update WSL with the following commands:

```
$ sudo apt-get update  
$ sudo apt-get upgrade
```



## (Win10) PART 4: Download dYdX trading rewards files

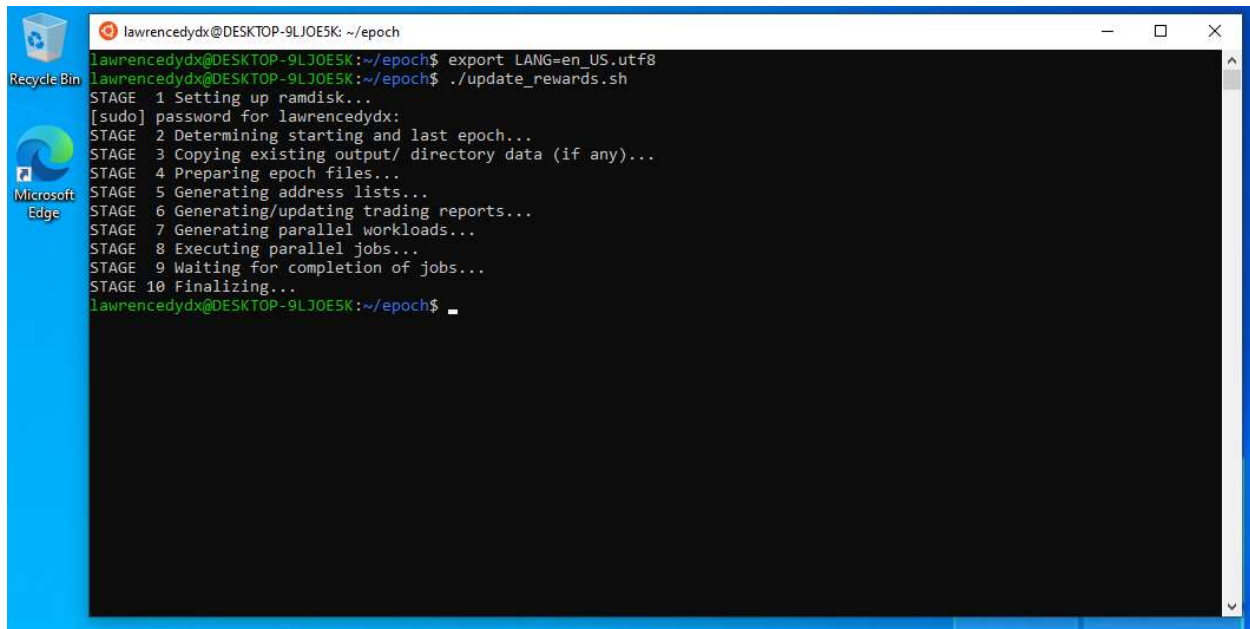
1) You can download the shell script and all epoch files with the following commands:

```
$ wget https://github.com/chiwalfrm/dydxexamples/raw/main/tradingrewards/epoch0.json
$ wget https://github.com/chiwalfrm/dydxexamples/raw/main/tradingrewards/epoch1.json
$ wget https://github.com/chiwalfrm/dydxexamples/raw/main/tradingrewards/epoch2.json
$ wget https://github.com/chiwalfrm/dydxexamples/raw/main/tradingrewards/epoch3.json
$ wget https://github.com/chiwalfrm/dydxexamples/raw/main/tradingrewards/epoch4.json
$ wget https://github.com/chiwalfrm/dydxexamples/raw/main/tradingrewards/epoch5.json
$ wget https://github.com/chiwalfrm/dydxexamples/raw/main/tradingrewards/epoch6.json
$ wget https://github.com/chiwalfrm/dydxexamples/raw/main/tradingrewards/epoch7.json
$ wget https://github.com/chiwalfrm/dydxexamples/raw/main/tradingrewards/epoch8.json
$ wget https://github.com/chiwalfrm/dydxexamples/raw/main/tradingrewards/update_rewards.sh
```

2) If you want commas in numbers for readability (e.g. display one million as 1,000,000), set the LANG variable:

```
$ export LANG=en_US.utf8
```

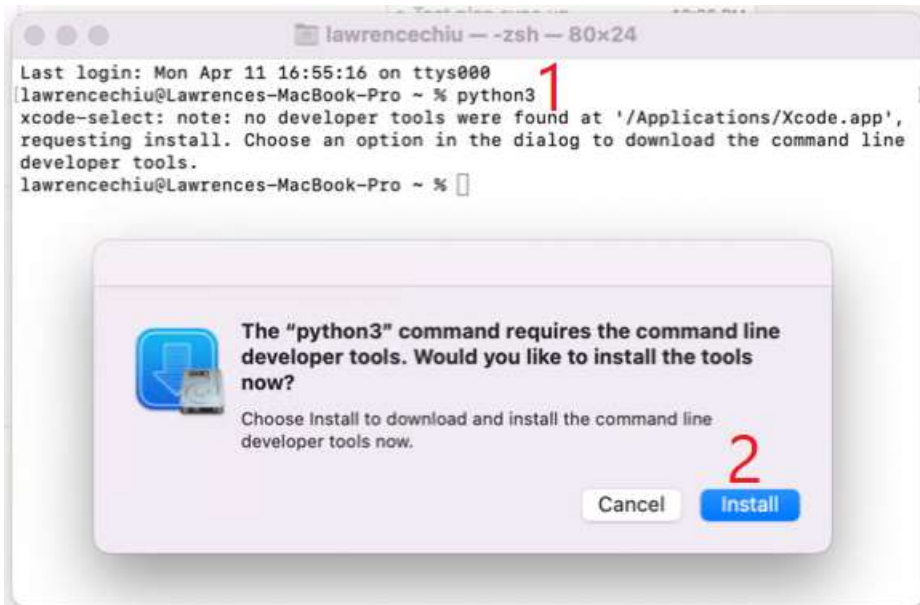
3) The rest of the guide follows the Linux version. Here is a picture of it running on Windows.



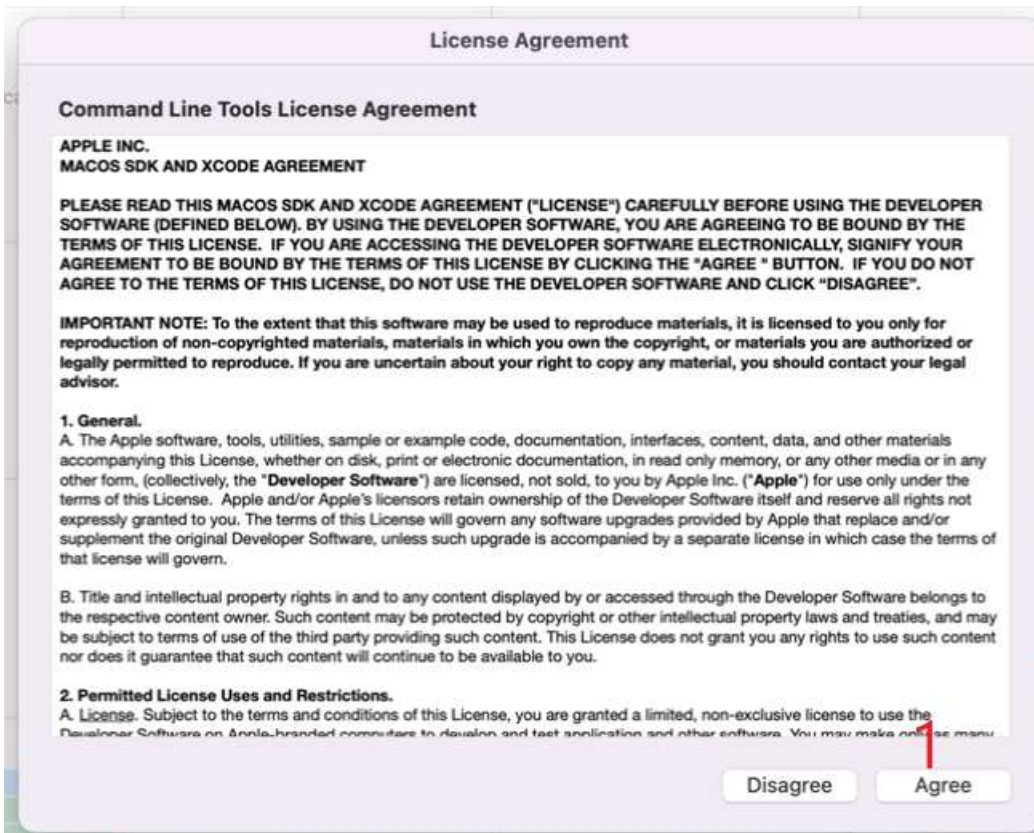
## APPENDIX B: MacOS

### (MacOS) PART 1: Install Python3

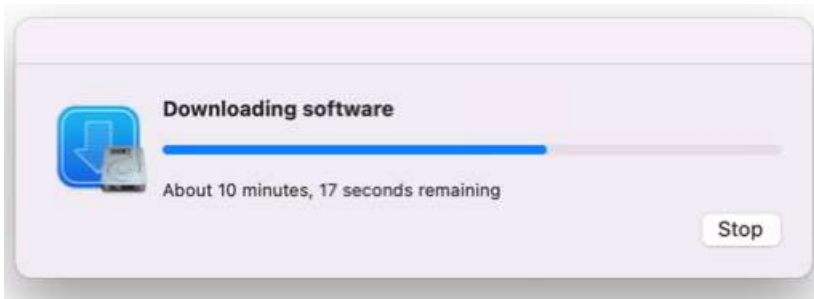
- 1) Launch Terminal and type 'python3'. Then click Install.



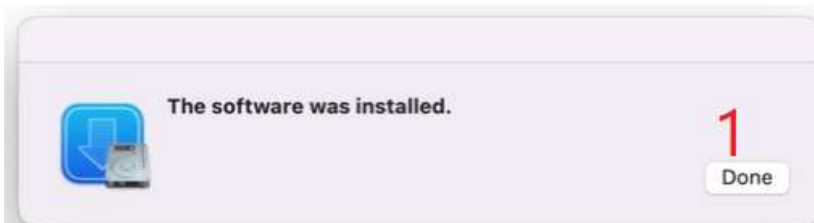
- 2) Click Agree.



3) Wait.



4) Click Done.



5) The rest of the guide follows the Linux version.

6) Here is a picture of it running on MacOS.

```
tradingrewards --zsh -- 80x24
lawrencechiu@Lawrences-MacBook-Pro tradingrewards % ./update_rewards.sh
STAGE 1 Setting up ramdisk...
Started erase on disk4
Unmounting disk
Erasing
Initialized /dev/rdisk4 as a 512 MB case-insensitive HFS Plus volume
Mounting disk
Finished erase on disk4 (epochdisk)
STAGE 2 Determining starting and last epoch...
STAGE 3 Copying existing output/ directory data (if any)...
STAGE 4 Preparing epoch files...
STAGE 5 Generating address lists...
STAGE 6 Generating/updating trading reports...
STAGE 7 Generating parallel workloads...
STAGE 8 Executing parallel jobs...
STAGE 9 Waiting for completion of jobs...
STAGE 10 Finalizing...
"disk4" ejected.
lawrencechiu@Lawrences-MacBook-Pro tradingrewards %
```

<END>