



یادگیری تقویتی در کنترل
تمرین اول: نصب **Gymnasium**

استاد: دکتر سعید شمسقدری

دانشجو: سیده ستاره خسروی

پائیر ۱۴۰۳

صورت تمرین

Gymnasium یک پکیج open-source در پایتون است که مجموعه‌ای از محیط‌های از پیش تعریف شده را برای توسعه و آزمایش الگوریتم‌های مختلف یادگیری تقویتی ارائه می‌دهد.

در این تمرین، هدف نصب Gymnasium در سه محیط

- Windows/MacOS
- Linux
- Google Colab

می‌باشد. (روی کامپیوترتان می‌توانید همزمان با ویندوز، سیستم عامل لینوکس (Ubuntu) را نصب کنید)

(۱) مراحل نصب در هر سه محیط را در سه فایل صوتی مجزا توضیح دهید. در گزارش نیز با مشخص نمودن فایل صوتی مربوط به هر بخش، عکسی از مرحله پایانی نصب را گزارش نمایید.

(۲) برای اطمینان از درستی نصب این پکیج، کدی را نوشته و در هر سه محیط اجرا نمایید. پس از گزارش کد و نتیجه حاصل در هر کدام از محیط‌ها، مراحل مختلف کد را توضیح دهید.

(۳) کدی در محیط Gymnasium بنویسید که با استفاده از آن نتیجه ۱۰۰۰ عمل تصادفی در محیط Mountain Car نشان داده شود. پاسخ حاصل را گزارش نمایید.

واژه‌های کلیدی: یادگیری تقویتی، OpenAI, Gymnasium

فهرست مطالب

صفحه	عنوان
ب.....	فهرست مطالب
ج.....	فهرست تصاویر و نمودارها
۱.....	فصل ۱: نصب بر روی ویندوز
۲.....	۱.۱ پیش‌نیاز نصب
۴.....	۱.۲ نصب Gymnasium
۱۰.....	۱.۳ اجرای کد نمونه در ویندوز
۱۴.....	فصل ۲: نصب در محیط colab
۱۵.....	۲.۱ نصب در محیط colab
۱۵.....	۲.۲ اجرای کد نمونه در colab
۱۸.....	فصل ۳: نصب در محیط لینوکس
۱۹.....	۳.۱ نصب لینوکس
۲۱.....	۳.۲ نصب anaconda در لینوکس
۲۷.....	۳.۳ اجرای کد نمونه در اینوکس
۲۹.....	فصل ۴: اجرای mountain car
۱۶.....	۴.۱ اجرای کد mountain car

فهرست تصاویر و نمودارها

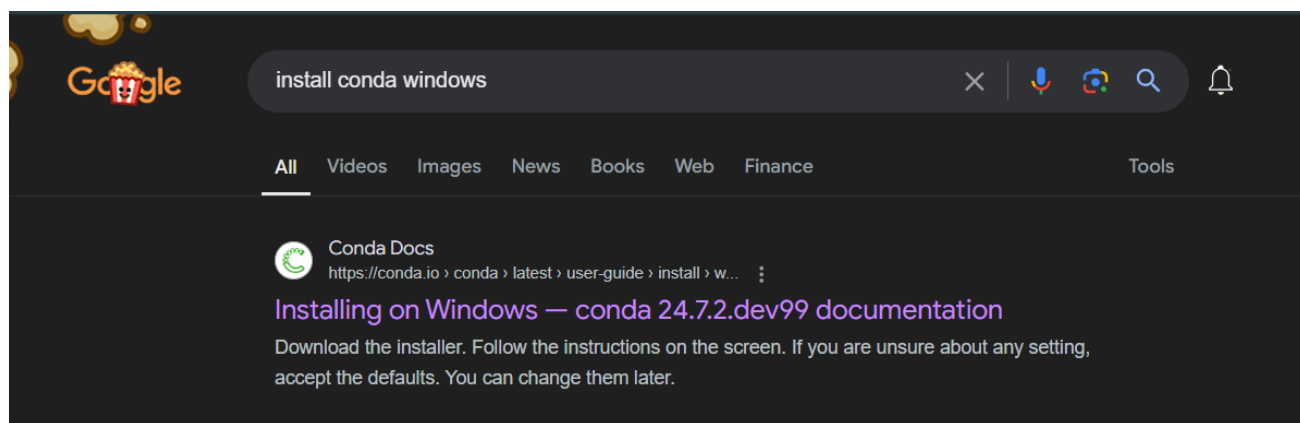
صفحه	عنوان
۲	شکل ۱: Conda Docs
۲	شکل ۲: صفحه راهنمای نصب anaconda بر ویندوز
۳	شکل ۳: صفحه دانلود miniconda
۳	شکل ۴: نصب آناکوندا
۴	شکل ۵: صفحه آخر نصب آناکوندا
۶	شکل ۶: راهنمای نصب Gymnasium
۶	شکل ۷: خطا در نصب Gymnasium
۷	شکل ۸: نصب swig
۷	شکل ۹: راهنمای حل مشکل نصب Box2d
۸	شکل ۱۰: خطای نصب box2d
۸	شکل ۱۱: دانلود و نصب ابزار مایکروسافت
۹	شکل ۱۲: نصب ابزار مایکروسافت (فرایند دانلود و نصب)
۹	شکل ۱۳: اتمام نصب gymnasium در ویندوز
۱۱	شکل ۱۴: اجرای محیط lunar lander
۱۲	شکل ۱۵: خروجی محیط mountain car در ویندوز
۱۲	شکل ۱۶: خروجی محیط cart pole در ویندوز
۱۳	شکل ۱۷: خروجی محیط taxi در ویندوز
۱۵	شکل ۱۸: نصب gymnasium بر روی colab
۱۶	شکل ۱۹: خطای ضبط خروجی در colab
۱۶	شکل ۲۰: کد مورد استفاده برای اجرا در colab
۱۷	شکل ۲۱: کد مورد استفاده برای چسباندن ویدیوها
۱۷	شکل ۲۲: خروجی محیط colab
۱۹	شکل ۲۳: آغاز نصب لینوکس
۲۰	شکل ۲۴: انتخاب زبان
۲۰	شکل ۲۵: انتخاب نوع نصب
۲۱	شکل ۲۶: انتخاب نحوه نصب

- شکل ۲۷: راهنمای نصب آناکوندا، نصب پیش‌نیازها..... ۲۲
- شکل ۲۸: دانلود آناکوندا..... ۲۲
- شکل ۲۹: نصب آناکوندا..... ۲۳
- شکل ۳۰: اتمام نصب..... ۲۳
- شکل ۳۱: اضافه کردن خطوط مرتبط با آناکوندا به bashrc..... ۲۴
- شکل ۳۲: چک نصب gymnasium..... ۲۴
- شکل ۳۳: چک نصب gymnasium در محیط بینایی..... ۲۵
- شکل ۳۴: خطا در نصب gymnasium در محیط لینوکس..... ۲۶
- شکل ۳۵: نصب موفقیت آمیز gymnasium در لینوکس..... ۲۶
- شکل ۳۶: چک نصب gymnasium در محیط لینوکس..... ۲۶
- شکل ۳۷: ایجاد کد..... ۲۷
- شکل ۳۸: خروجی gymnasium و محیط cart pole در لینوکس..... ۲۷
- شکل ۳۹: حذف gymnasium..... ۲۸
- شکل ۴۰: بازگشت به base و ساخت کد پایتون..... ۱۶
- شکل ۴۱: کد اجرای mountain car..... ۱۶
- شکل ۴۲: خروجی ۱۰۰۰۰ عمل تصادفی در محیط لینوکس..... ۱۷
- شکل ۴۳: اتمام اجرای کد..... ۱۷
- شکل ۴۴: اجرای مجدد mountain car حالت continous..... ۱۸
- شکل ۴۵: خروجی mountain car در هنگام اجرا..... ۱۸

فصل ۱: نصب بروی ویندوز

۱.۱ پیش‌نیاز نصب

در این بخش ابتدا لازم است برای نصب Gymnasium بر روی سیستم عامل ویندوز، Anaconda/Miniconda را بر روی سیستم نصب کنیم. برای نصب Miniconda را انتخاب می‌کنیم. برای نصب به وبسایت conda مراجعه می‌کنیم.



شکل ۱: Conda Docs

سپس در صفحه‌ای که باز می‌شود به عنوان مثال Miniconda را انتخاب می‌کنیم.



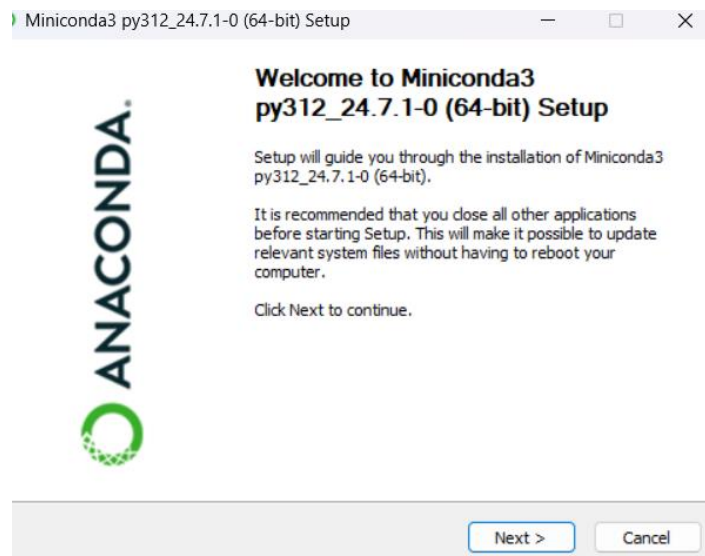
شکل ۲: صفحه راهنمای نصب anaconda بر ویندوز

با انتخاب گزینه‌ی miniconda به صفحه‌ی دیگری هدایت می‌شوید که لازم است نسخه مورد نیاز (نسخه ۶۴ بیتی) را انتخاب کنید که پس از آن فایل exe. دانلود می‌شود. (شکل ۳)

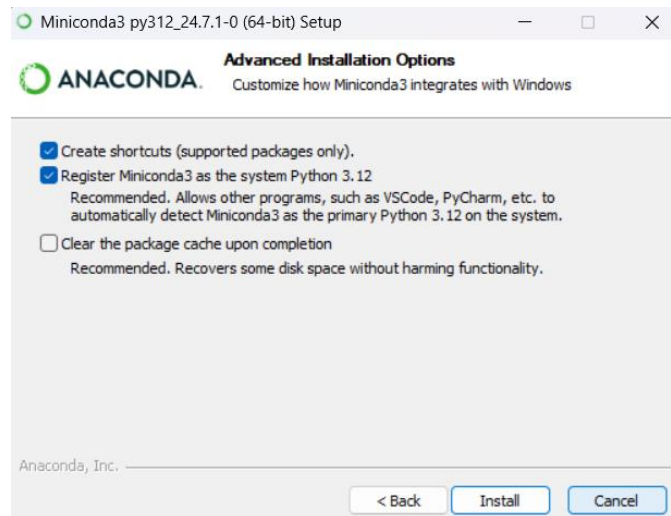
Platform	Name	SHA256 hash
Windows	Miniconda3 Windows 64-bit	ff8ab50f0303c7b9097387967ac2a721016d020069187eff4e172fc14930ebb7
macOS	Miniconda3 macOS Intel x86 64-bit bash	5cfb85d81d94dfe3ef3265f2247aef32a35aeb450ea71c3a204ced384fb87d
	Miniconda3 macOS Intel x86 64-bit pkg	e31844adec03a69e274538a796c3b4183cd2cbc7b90fd8ea98b591a6313a330b
	Miniconda3 macOS Apple M1	e7ef5a899f9383d14d5b15aef61d54a8cd9bf3c4de18a372af0455d8f5f78cd2

شکل ۳: صفحه دانلود miniconda

پس از اتمام دانلود فایل لازم است فایل نصبی اجرا شود. با مشاهده شکل ۴ در تمامی مراحل next را انتخاب کرده تا آناکوندا نصب شود، با توجه به اینکه از قبل بر روی سیستم نصب بود، تا شکل ۵ پیش رفته و سپس فرایند نصب را قطع کردیم.



شکل ۴: نصب آناکوندا



شکل ۵: صفحه آخر نصب آناکوندا

با توجه به اینکه از قبل آناکوندا نصب بوده است می‌توانیم برای ادامه کار از محیط Anaconda Prompt استفاده کنیم.

۱.۲ نصب Gymnasium

حال محیط Anaconda Prompt را باز می‌کنیم و با استفاده از دستور زیر یک محیط می‌سازیم.

```
(base) C:\Users\asus>conda create -n RLClass
```

پس از ایجاد محیط خروجی زیر قابل مشاهده است.

```
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate RLClass
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

با استفاده دستور زیر محیطی که ساختیم را فعال می‌کنیم.

```
(base) C:\Users\asus>conda activate RLClass
(RLClass) C:\Users\asus>
```

لازم است در محیطی که ساختیم پایتون را نصب کنیم. بدین منظور لازم است نسخه‌ای سازگار با Gymnasium نصب شود. با استفاده از دستور زیر پایتون نسخه ۳.۱۰ را نصب می‌کنیم.

```
(RLClass) C:\Users\asus>conda install python=3.10
```

پس از وارد کردن دستور نصب پایتون، تعدادی پکیج نصب می‌شود و خروجی زیر قابل مشاهده است:

```
Anaconda Prompt - conda cr  X + v

setuptools      pkgs/main/win-64::setuptools-75.1.0-py310haa95532_0
sqlite          pkgs/main/win-64::sqlite-3.45.3-h2bbff1b_0
tk              pkgs/main/win-64::tk-8.6.14-h0416ee5_0
tzdata          pkgs/main/noarch::tzdata-2024a-h04d1e81_0
vc              pkgs/main/win-64::vc-14.40-h2eaa2aa_1
vs2015_runtime  pkgs/main/win-64::vs2015_runtime-14.40.33807-h98bb1dd_1
wheel           pkgs/main/win-64::wheel-0.44.0-py310haa95532_0
xz              pkgs/main/win-64::xz-5.4.6-h8cc25b3_1
zlib            pkgs/main/win-64::zlib-1.2.13-h8cc25b3_1

Proceed ([y]/n)? y

Downloading and Extracting Packages

Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(RLClass) C:\Users\asus>
```

برای نصب Gymnasium مطابق راهنمای نصب آن (شکل ۶) پیش می‌رویم. برای نصب Gymnasium و تمامی پکیج‌های آن دستور زیر اجرا می‌گردد.

```
(RLClass) C:\Users\asus>pip install gymnasium[all]
```

Installation

To install the base Gymnasium library, use `pip install gymnasium`

This does not include dependencies for all families of environments (there's a massive number, and some can be problematic to install on certain systems). You can install these dependencies for one family like `pip install "gymnasium[atari]"` or use `pip install "gymnasium[all]"` to install all dependencies.

We support and test for Python 3.8, 3.9, 3.10, 3.11 on Linux and macOS. We will accept PRs related to Windows, but do not officially support it.

شکل ۶: راهنمای نصب Gymnasium

پس از اجرای دستور نصب، نصب کامل انجام نمی‌شود و در خروجی خطا مشاهده می‌شود.

```

Anaconda Prompt - conda cr
Download etils-1.9.4-py3-none-any.whl (164 kB)
Download importlib_resources-6.4.5-py3-none-any.whl (36 kB)
Download pycparser-2.22-py3-none-any.whl (117 kB)
Download zipp-3.20.2-py3-none-any.whl (9.2 kB)
Building wheels for collected packages: box2d-py, moviepy
Building wheel for box2d-py (setup.py) ... error
error: subprocess-exited-with-error

x python setup.py bdist_wheel did not run successfully.
exit code: 1
[16 lines of output]
Using setuptools (version 75.1.0).
C:\Users\asus\conda\envs\RLClass\lib\site-packages\setuptools\_distutils\dist.py:261: UserWarning: Unknown distribution option: 'test_suite'
  warnings.warn(msg)
running bdist_wheel
running build
running build_py
creating build\lib.win-amd64-cpython-310\Box2D
copying library\Box2D\Box2D.py -> build\lib.win-amd64-cpython-310\Box2D
copying library\Box2D\__init__.py -> build\lib.win-amd64-cpython-310\Box2D
creating build\lib.win-amd64-cpython-310\Box2D\b2
copying library\Box2D\b2\__init__.py -> build\lib.win-amd64-cpython-310\Box2D\b2
running build_ext
building 'Box2D.Box2D' extension
swigging Box2D\Box2D.i to Box2D\Box2D_wrap.cpp
swig.exe -python -c++ -IBox2D -small -O -includeall -ignoremissing -w201 -globals b2Globals -outdir library\Box2D -keyword -w511 -D_SWIG_KWARGS -o Box
2D\Box2D_wrap.cpp Box2D\Box2D.i
error: command 'swig.exe' failed: None
[end of output]

note: This error originates from a subprocess, and is likely not a problem with pip.
ERROR: Failed building wheel for box2d-py
Running setup.py clean for box2d-py
Building wheel for moviepy (setup.py) ... done
Created wheel for moviepy: filename=moviepy-1.0.3-py3-none-any.whl size=110755 sha256=943736a3c552e10c25664704906ad161aac88d5a52b861cc1867cf9908366e55
Stored in directory: c:\users\asus\appdata\local\pip\cache\wheels\96\32\2d\e10123bd88fbfc02fed53cc18c80a171d3c87479ed845fa7c1
Successfully built moviepy
Failed to build box2d-py
ERROR: ERROR: Failed to build installable wheels for some pyproject.toml based projects (box2d-py)

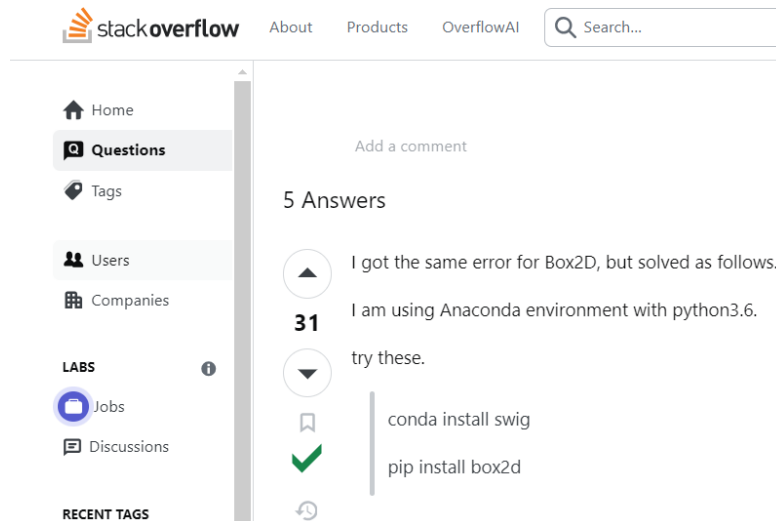
(RLClass) C:\Users\asus>conda install swig

```

شکل ۷: خطا در نصب Gymnasium

با جست و جوی خطا در اینترنت، راه حل آن را در stackoverflow یافتیم. برای حل آن گفته شده است که ابتدا باید پکیج swig را نصب کنیم (شکل ۸)، که این کار با دستور زیر انجام می‌شود.

```
(RLClass) C:\Users\asus>conda install swig
```



شکل ۹: راهنمای حل مشکل نصب Box2d

خروجی نصب swig مطابق شکل زیر است:

```
The following packages will be downloaded:

package | build
-----|-----
swig-4.0.2 | hd77b12b_4 1.5 MB
-----|-----
Total: 1.5 MB

The following NEW packages will be INSTALLED:

swig          pkgs/main/win-64::swig-4.0.2-hd77b12b_4

Proceed ([y]/n)? y

Downloading and Extracting Packages
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

شکل ۸: نصب swig

مجدد دستور نصب Gymnasium (با تمام محیط‌های آن) را اجرا می‌کنیم، همچنان نصب آن بخاطر خطا در نصب box2d با مشکل مواجه می‌شود.

```

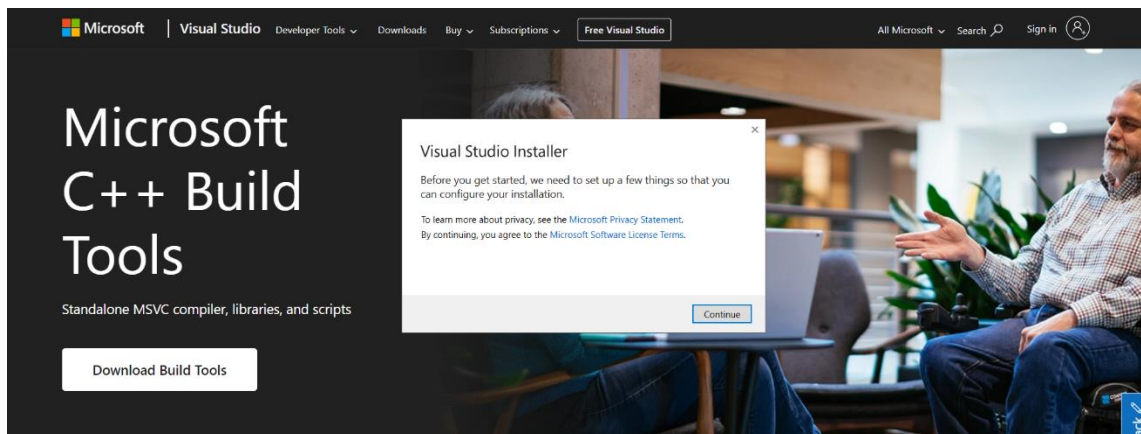
exit code: 1
[28 lines of output]
Using setuptools (version 75.1.0).
C:\Users\asus\conda\envs\RLClass\lib\site-packages\setuptools\_distutils\dist.py:261: UserWarning: Unknown distribution option: 'test_suite'
  warnings.warn(msg)
running bdist_wheel
running build
running build_py
creating build\lib.win-amd64-cpython-310\Box2D
copying library\Box2D\Box2D.py -> build\lib.win-amd64-cpython-310\Box2D
copying library\Box2D\_init_.py -> build\lib.win-amd64-cpython-310\Box2D
creating build\lib.win-amd64-cpython-310\Box2D\b2
copying library\Box2D\b2\_init_.py -> build\lib.win-amd64-cpython-310\Box2D\b2
running build_ext
building 'Box2D.Box2D' extension
swigging Box2D\Box2D.i to Box2D\Box2D_wrap.cpp
swig.exe -python -c++ -IBox2D -small -O -includeall -ignoremissing -w201 -globals b2Globals -outdir library\Box2D -keyword -w511 -D_SWIG_KEYWORDS -o Box2D\Box2D_wrap.cpp Box2D\Box2D.i
Box2D\Box2D\Box2D.h(67) : Warning 302: Identifier 'b2Vec2' redefined by %extend (ignored),
Box2D\Box2D\Box2D.h(47) : Warning 302: %extend definition of 'b2Vec2'.
Box2D\Box2D\Box2D.h(158) : Warning 302: Identifier 'b2Vec3' redefined by %extend (ignored),
Box2D\Box2D\Box2D.h(168) : Warning 302: %extend definition of 'b2Vec3'.
Box2D\Box2D\Box2D.h(197) : Warning 302: Identifier 'b2Mat22' redefined by %extend (ignored),
Box2D\Box2D\Box2D.h(301) : Warning 302: %extend definition of 'b2Mat22'.
Box2D\Box2D\Box2D.h(271) : Warning 302: Identifier 'b2Mat33' redefined by %extend (ignored),
Box2D\Box2D\Box2D.h(372) : Warning 302: %extend definition of 'b2Mat33'.
Box2D\Box2D\Box2D.h(144) : Warning 506: Can't wrap varargs with keyword arguments enabled
Box2D\Box2D\Box2D.h(91) : Warning 509: Overloaded method b2Vec2::operator()(int32) effectively ignored,
Box2D\Box2D\Box2D.h(85) : Warning 509: as it is shadowed by b2Vec2::operator()(int32) const.
error: Microsoft Visual C++ 14.0 or greater is required. Get it with "Microsoft C++ Build Tools": https://visualstudio.microsoft.com/visual-cpp-build-tools/
[end of output]

note: This error originates from a subprocess, and is likely not a problem with pip.
ERROR: Failed building wheel for box2d-py
Running setup.py clean for box2d-py
Failed to build box2d-py
ERROR: ERROR: Failed to build installable wheels for some pyproject.toml based projects (box2d-py)

```

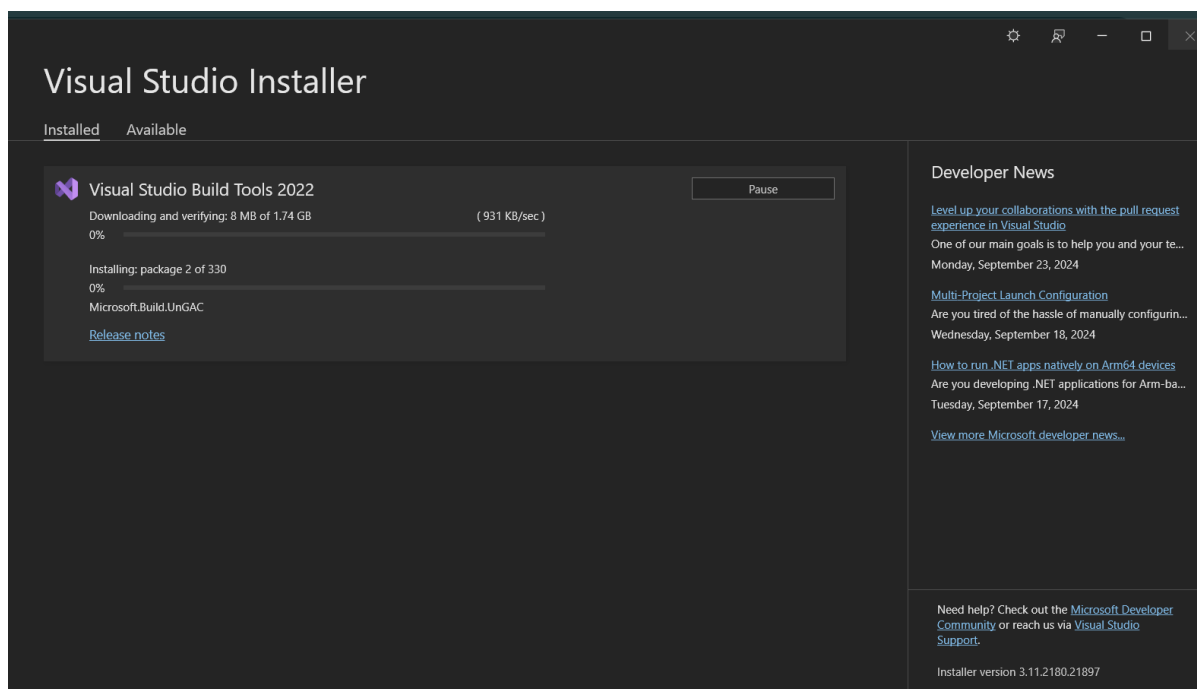
شکل ۱۰: خطای نصب box2d

در همان صفحه‌ی stackoverflow نیز مشکل مربوط به این قسمت تشریح شده و حل آن نیز مطابق توضیحات و همچنین متن خطا، با نصب Microsoft C++ Build Tools میسر است. با مراجعه به لینکی که در متن خطا اشاره شده است، ابزار مورد نیاز را دانلود کرده و فایل نصبی آن را اجرا می‌کنیم. (شکل ۱۱)



شکل ۱۱: دانلود و نصب ابزار مایکروسافت

در صفحات بعدی نصب آن لازم است Microsoft C++ Build Tools انتخاب شود و سپس صفحه‌ی زیر ظاهر می‌گردد.



شکل ۱۲: نصب ابزار مایکروسافت (فرایند دانلود و نصب)

پس از اتمام نصب ابزار مایکروسافت، نصب gymnasium را با دستور زیر انجام می‌دهیم.

```
(RLClass) C:\Users\asus>pip install gymnasium[all]
```

خروجی نصب نیز به صورت شکل ۱۳ است.

```
Successfully built box2d-py
Installing collected packages: swig, mpmath, box2d-py, urllib3, sympy, six, scipy, pyparsing, pycparser, packaging, opt-einsum, opencv-python, networkx, ml-dtypes, MarkupSafe, lz4, kiwisolver, imageio-ffmpeg, idna, fonttools, filelock, fasteners, decorator, cython, cycler, contourpy, colorama, charset-normalizer, certifi, tqdm, requests, python-dateutil, Jinja2, jaxlib, cffi, torch, proglog, mujoco-py, matplotlib, jax, moviepy
Successfully installed MarkupSafe-2.1.5 box2d-py-2.3.5 certifi-2024.8.30 cffi-1.17.1 charset-normalizer-3.3.2 colorama-0.4.6 contourpy-1.3.0 cycler-0.12.1 cython-0.29.37 decorator-4.4.2 fasteners-0.19 filelock-3.16.1 fonttools-4.54.1 idna-3.10 imageio-ffmpeg-0.5.1 jax-0.4.33 jaxlib-0.4.33 Jinja2-3.1.4 kiwisolver-1.4.7 lz4-4.3.3 matplotlib-3.9.2 ml-dtypes-0.5.0 moviepy-1.0.3 mpmath-1.3.0 mujoco-py-2.1.2.14 networkx-3.3 opencv-python-4.10.0.84 opt-einsum-3.4.0 packaging-24.1 proglog-0.1.10 pycparser-2.22 pyparsing-3.1.4 python-dateutil-2.9.0.post0 requests-2.32.3 scipy-1.14.1 six-1.16.0 swig-4.2.1 sympy-1.13.3 torch-2.4.1 tqdm-4.66.5 urllib3-2.2.3
```

شکل ۱۳: اتمام نصب gymnasium در ویندوز

۱.۳ اجرای کد نمونه در ویندوز

از کد زیر برای بررسی صحت نصب پکیج در محیط ویندوز استفاده می‌کنیم.

```
"""
Course: Reinforcement Learning in Control
Semester: 4031 | Fall 2024
Student: STRH
Created on Thu, 26 Sept 2024, 19:01:00
HW1: Gym Installation on windows
"""

import gymnasium as gym

env = gym.make("Humanoid-v4", render_mode="human")
observation, info = env.reset(seed=42)
for _ in range(1000):
    action = env.action_space.sample()
    observation, reward, terminated, truncated, info = env.step(action)

    if terminated or truncated:
        observation, info = env.reset()

env.close()
```

در این کد با دستور `gym.make` محیط مدنظر را می‌سازیم، که در این بخش محیط‌های مختلف را امتحان کرده و خروجی هر یک را نشان می‌دهیم، ضمناً ذکر می‌کنیم که پردازش‌های هر محیط به صورت گرافیکی نمایش داده شود.

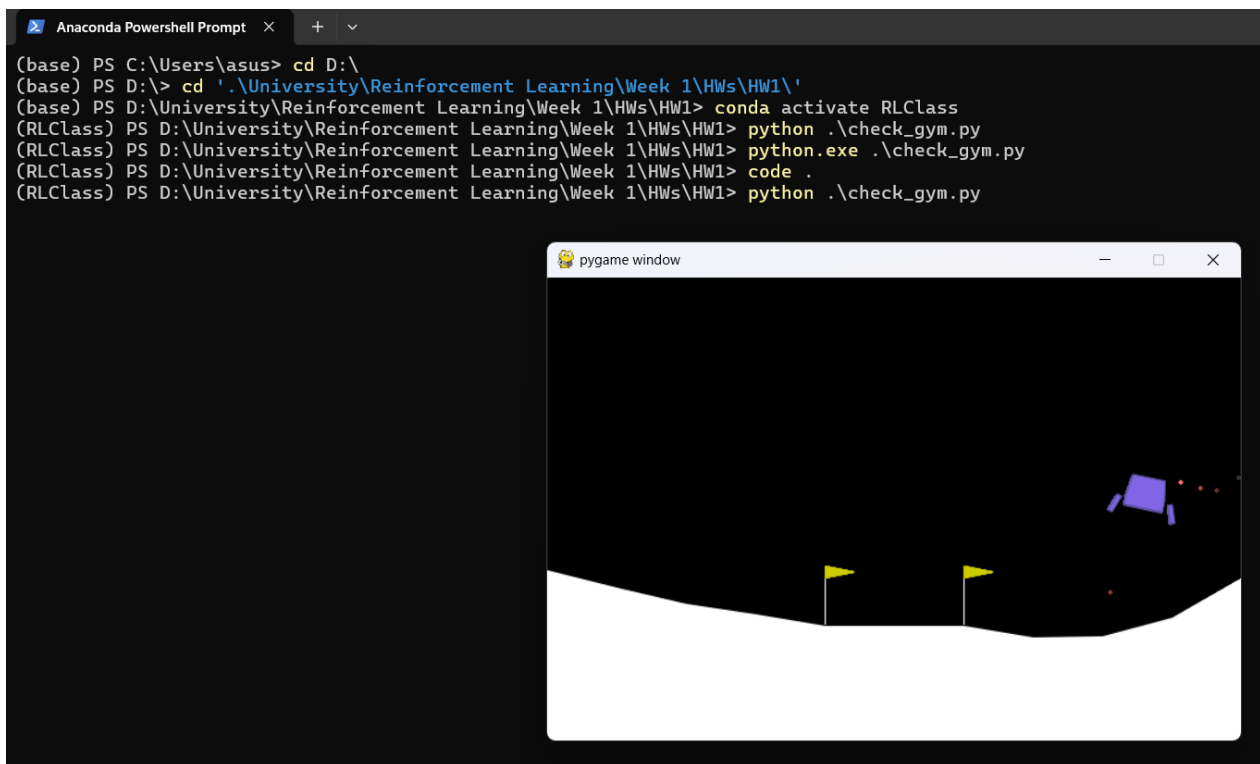
با دستور `env.reset` محیط آغاز به کار کرده (`initialize`) می‌شود یا می‌توان آن را `reset` کرد. عدد ۴۲ بخاطر این است که در اجراهای مختلفی که از یک محیط می‌گیریم نتایج یکسان حاصل شود. `Observation` حالت اولیه‌ی سیستم را بر می‌گرداند و `info` شامل اطلاعات اضافی در خصوص محیط است.

حلقه‌ی بعدی مربوط به اجرای شبیه سازی برای ۱۰۰۰ تلاش است. که در هر قدم عامل در محیط یک عملی را انجام می‌دهد. خروجی هر قدم که عملی انجام می‌شود، حالت جدید سیستم، میزان پاداش، مقداری منطقی که نشان دهنده اتمام `episode` است (`terminated`) و مقداری منطقی که نشان دهنده

اتمام episode است بخاطر محدودیت زمانی و یا هرچیزی بجز اتمام task و در نهایت مشابه قبل info شامل اطلاعات اضافی دیگری در خصوص محیط است.

سپس چک می‌کنیم اگر task به علت موفقیت در انجام و یا محدودیت زمانی و ... تمام شود، محیط مجدداً reset می‌شود. در نهایت پس از اتمام ۱۰۰۰ تلاش، محیط بسته می‌شود.

مطابق شکل زیر کدی که نوشتیم را اجرا می‌گیریم، خروجی زیر مربوط به محیط lunar lander است.



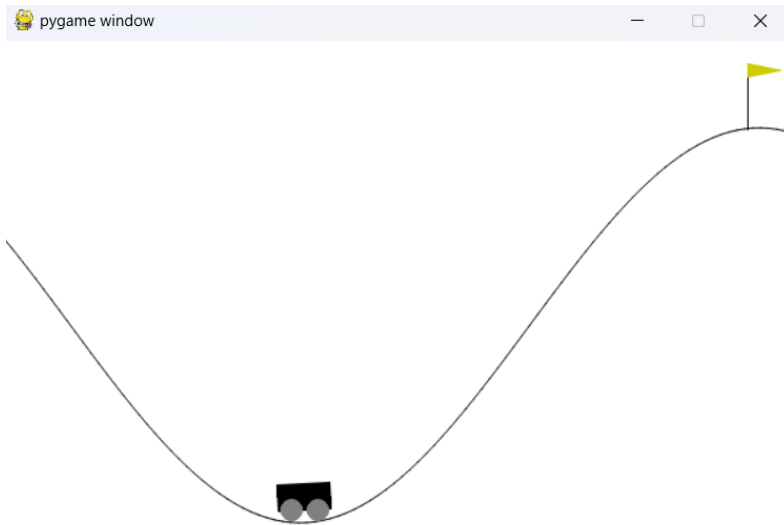
The screenshot shows an Anaconda Powershell Prompt window with the following commands and output:

```
(base) PS C:\Users\asus> cd D:\
(base) PS D:\> cd '..\University\Reinforcement Learning\Week 1\Hws\HW1\'
(base) PS D:\University\Reinforcement Learning\Week 1\Hws\HW1> conda activate RLClass
(RLClass) PS D:\University\Reinforcement Learning\Week 1\Hws\HW1> python .\check_gym.py
(RLClass) PS D:\University\Reinforcement Learning\Week 1\Hws\HW1> python.exe .\check_gym.py
(RLClass) PS D:\University\Reinforcement Learning\Week 1\Hws\HW1> code .
(RLClass) PS D:\University\Reinforcement Learning\Week 1\Hws\HW1> python .\check_gym.py
```

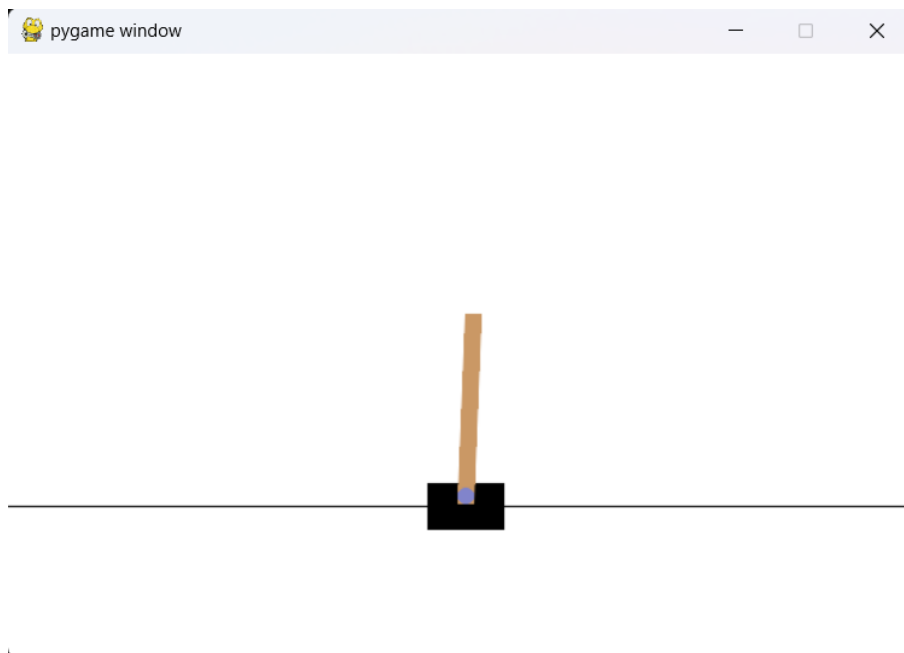
Below the terminal window, a Pygame window titled "pygame window" displays the Lunar Lander game. The game shows a blue lander on a black background, with a white ground line and two yellow flags at the bottom. The lander is positioned on the right side of the screen, and the flags are on the left.

شکل ۱۴: اجرای محیط lunar lander

خروجی محیط‌های mountain car، cart pole نیز به ترتیب در شکل‌های ۱۵ و ۱۶ قرار دارد.

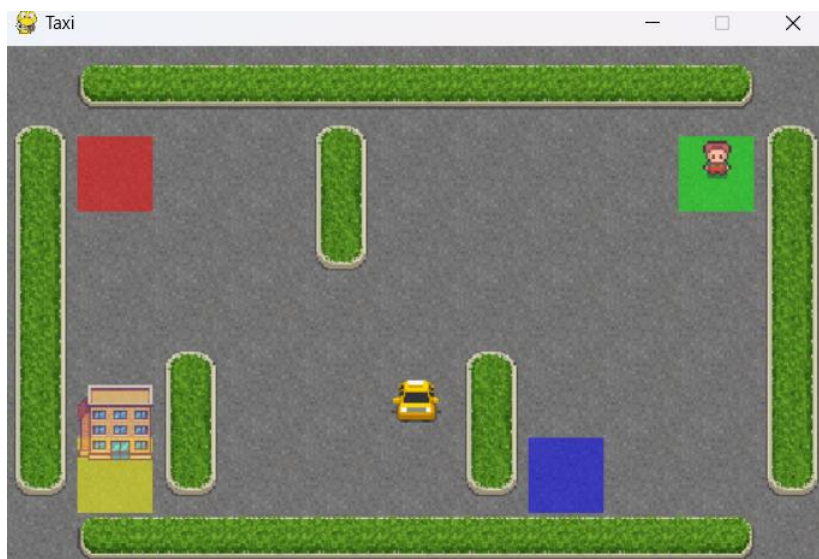


شکل ۱۵: خروجی محیط mountain car در ویندوز



شکل ۱۶: خروجی محیط cart pole در ویندوز

و در نهایت محیط taxi اجرا گردید.



شکل ۱۷: خروجی محیط taxi در ویندوز

فصل ۲: نصب در محیط colab

۲.۱ نصب در محیط colab

در این مطابق تصویر زیر دو محیط از gymnasium را بر روی colab نصب می‌کنیم.

```
[2] !pip install gymnasium[classic-control]

collecting gymnasium[classic-control]
  Using cached gymnasium-0.29.1-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from gymnasium[classic-control]) (1.26.4)
Requirement already satisfied: cloudpickle>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from gymnasium[classic-control]) (2.2.1)
Requirement already satisfied: typing-extensions>=4.3.0 in /usr/local/lib/python3.10/dist-packages (from gymnasium[classic-control]) (4.12.2)
collecting farama-notifications>=0.0.1 (from gymnasium[classic-control])
  Using cached Farama_Notifications-0.0.4-py3-none-any.whl.metadata (558 bytes)
Requirement already satisfied: pygame>=2.1.3 in /usr/local/lib/python3.10/dist-packages (from gymnasium[classic-control]) (2.6.0)
Using cached Farama_Notifications-0.0.4-py3-none-any.whl (2.5 kB)
Using cached gymnasium-0.29.1-py3-none-any.whl (953 kB)
Installing collected packages: farama-notifications, gymnasium
Successfully installed farama-notifications-0.0.4 gymnasium-0.29.1

[3] !pip install gymnasium[toy-text]

Requirement already satisfied: gymnasium[toy-text] in /usr/local/lib/python3.10/dist-packages (0.29.1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from gymnasium[toy-text]) (1.26.4)
Requirement already satisfied: cloudpickle>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from gymnasium[toy-text]) (2.2.1)
Requirement already satisfied: typing-extensions>=4.3.0 in /usr/local/lib/python3.10/dist-packages (from gymnasium[toy-text]) (4.12.2)
Requirement already satisfied: farama-notifications>=0.0.1 in /usr/local/lib/python3.10/dist-packages (from gymnasium[toy-text]) (0.0.4)
Requirement already satisfied: pygame>=2.1.3 in /usr/local/lib/python3.10/dist-packages (from gymnasium[toy-text]) (2.6.0)
```

شکل ۱۸: نصب gymnasium بر روی colab

همانطور که مشاهده می‌گردد نصب gymnasium در colab با موفقیت انجام شد.

۲.۲ اجرای کد نمونه در colab

برای این بخش نیز از کدی که در بخش نصب ویندوز به توضیح آن پرداختیم، استفاده می‌شود. با توجه به اینکه امکان نمایش خروجی محیط در colab فراهم نیست، دو خط به کد قبلی جهت ضبط خروجی اضافه شد. این کد به دلیل اینکه render_mode ابتدا human بود، خطا می‌دهد (شکل ۱۹) و لازم است برای ضبط خروجی render_mode آن به rgb_array تغییر کند.

```

ValueError                                Traceback (most recent call last)
<ipython-input-9-7e0d7dc6e283> in <cell line: 12>()
    10
    11 env = gym.make("CartPole-v1", render_mode="human")
--> 12 env = RecordVideo(env, "test.mp4", step_trigger = lambda episode_number: True)
    13 observation, info = env.reset(seed=42)
    14 for _ in range(1000):

/usr/local/lib/python3.10/dist-packages/gymnasium/wrappers/record_video.py in __init__(self, env, video_folder, episode_trigger, step_trigger, video_length, name_prefix, disable_logger)
    72
    73     if env.render_mode in {None, "human", "ansi", "ansi_list"}:
--> 74         raise ValueError(
    75             f"Render mode is {env.render_mode}, which is incompatible with"
    76             f" RecordVideo. Initialize your environment with a render_mode"

ValueError: Render mode is human, which is incompatible with RecordVideo. Initialize your environment with a render_mode that returns an image, such as rgb_array.

```

شکل ۱۹: خطای ضبط خروجی در colab

با اصلاح این مورد، کد زیر در محیط colab اجرا شد و خروجی به صورت ویدیو ذخیره گردید.

```

"""
Course: Reinforcement Learning in Control
Semester: 4031 | Fall 2024
Student: STRH
Created on Thu, 26 Sept 2024, 19:01:00
HW1
"""

import gymnasium as gym
from gymnasium.wrappers.record_video import RecordVideo

env = gym.make("CartPole-v1", render_mode="rgb_array")
env = RecordVideo(env, "test.mp4", step_trigger = lambda episode_number: True)
observation, info = env.reset(seed=42)
for _ in range(1000):
    action = env.action_space.sample()
    observation, reward, terminated, truncated, info = env.step(action)

    if terminated or truncated:
        observation, info = env.reset()

env.close()

```

شکل ۲۰: کد مورد استفاده برای اجرا در colab

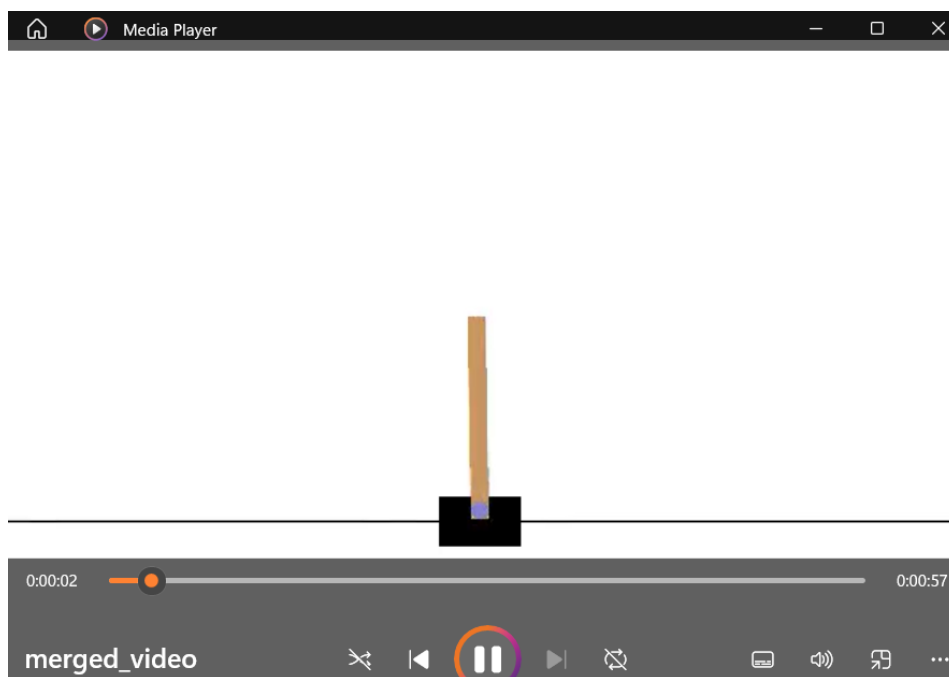
با توجه به اینکه خروجی در چندین ویدیو ذخیره شد، با استفاده از کدی دیگر تمام ویدیوها به یکدیگر متصل شدند و نتیجه اجرا در شکل ۲۲ موجود است.

```
import os
from natsort import natsorted
from moviepy.editor import VideoFileClip, concatenate_videoclips

# Folder containing video files
file_paths = "/content/test"
video_clips = natsorted([os.path.join(file_paths, f) for f in os.listdir(file_paths) if f.endswith('.mp4')])
clips = [VideoFileClip(clip) for clip in video_clips]
final_clip = concatenate_videoclips(clips)
output_path = "/content/merged_video.mp4"
final_clip.write_videofile(output_path, codec='libx264', audio_codec='aac')

print(f"Video saved to {output_path}")
```

شکل ۲۱: کد مورد استفاده برای چسباندن ویدیوها



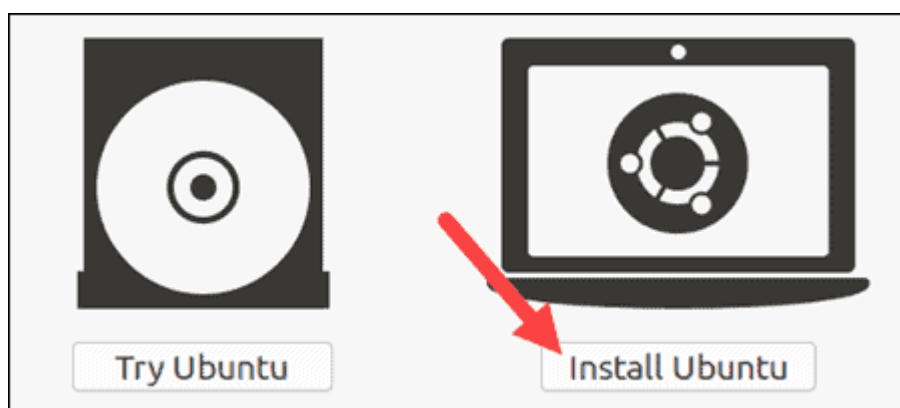
شکل ۲۲: خروجی محیط colab

فصل ۳: نصب در محیط لینوکس

۳.۱ نصب لینوکس

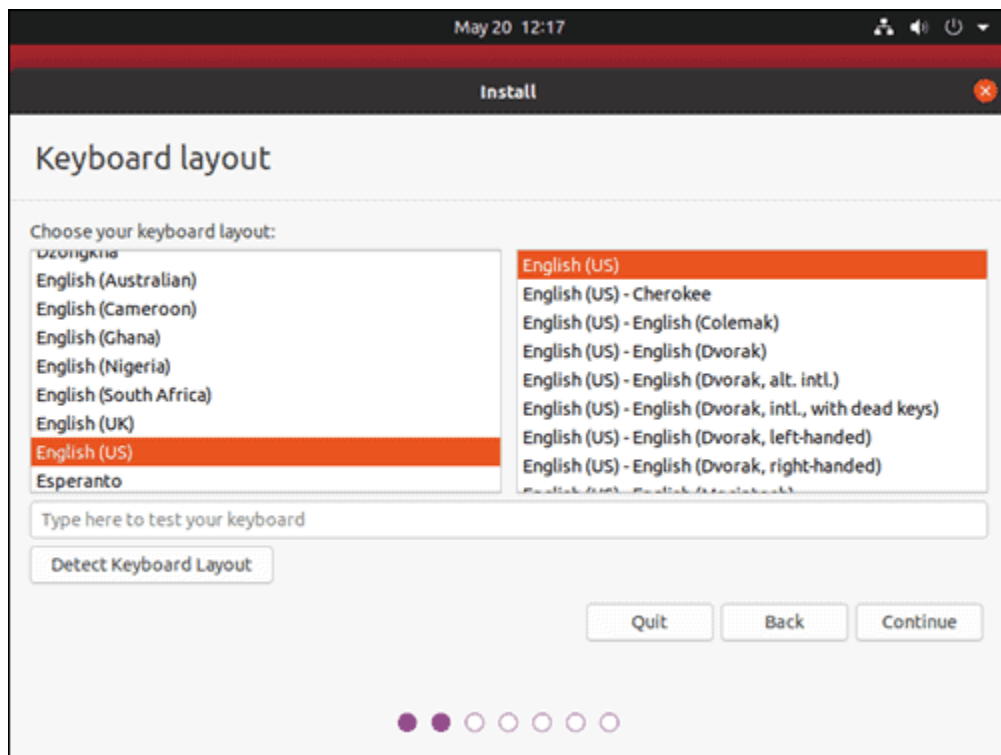
با استفاده از virtual box و یا به صورت dual boot میتوان سیستم عامل لینوکس را در کنار ویندوز نصب کرد. با توجه به اینکه سیستم عامل لینوکس (Ubuntu 22.04) از قبل بر روی سیستم نصب است (به صورت dual boot) به طور خلاصه به نصب نسخه 20.04 اشاره می‌گردد. (نصب نسخه‌ها تفاوتی ندارند) ابتدا لازم است با استفاده از disk management یکی از درایوهایی که حجم خالی دارد، یا همه‌ی آن را اگر خالی است به حالت unallocated دریاوریم. بر روی درایو مدنظر کلیک راست کرده و shrink را انتخاب می‌کنیم، میزان حجمی که لازم داریم را وارد کرده و سپس همه‌ی درایو یا بخشی از آن (بستگی به حجم انتخابی دارد) به صورت unallocated در می‌آید، که در ادامه لینوکس بر روی آن نصب خواهد شد. برای نصب لینوکس ابتدا لازم است فایل iso آن از سایت رسمی دانلود شود. سپس با استفاده از نرم افزار rufus یک فلش خالی را bootable می‌کنیم، که بتوان با استفاده از آن سیستم عامل را بر روی سیستم نصب کرد.

برای اینکه نصب انجام شود لازم است سیستم را restart کرده، در منوی bios سیستم اولویت boot را به فلش بدهیم، سپس پس از بالا آمدن سیستم وارد مراحل نصب لینوکس می‌شویم.



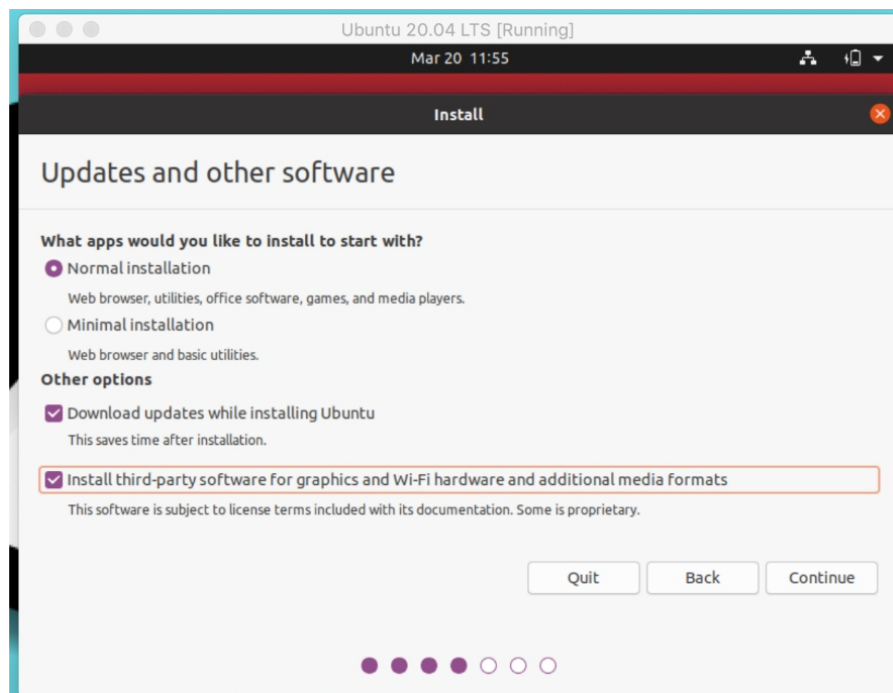
شکل ۲۳: آغاز نصب لینوکس

در هنگام نصب، ابتدا زبان باید انتخاب شود که آن را انگلیسی انتخاب می‌کنیم.



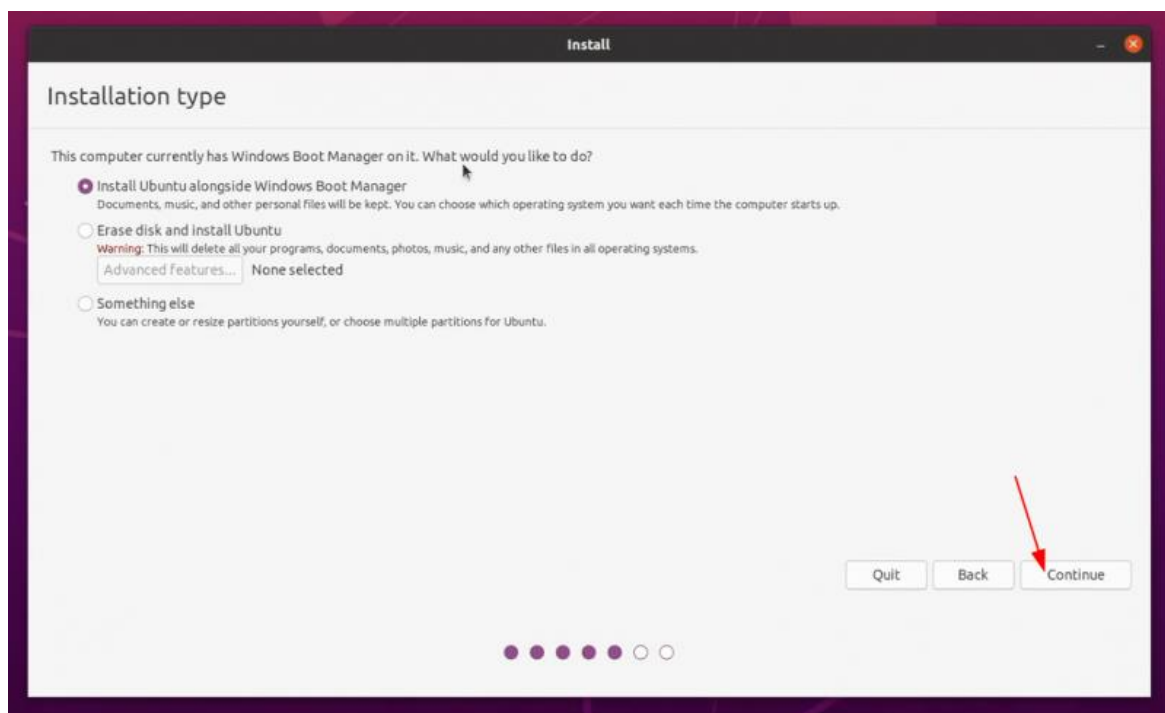
شکل ۲۴: انتخاب زبان

سپس ادامه‌ی نصب را مطابق شکل ۲۵ پیش می‌بریم.



شکل ۲۵: انتخاب نوع نصب

حال لازم است انتخاب کنیم که می‌خواهیم لینوکس در کنار ویندوز نصب شود. این بخش مهم و حیاتی است و نباید گزینه‌ی اشتباهی انتخاب شود.



شکل ۲۶: انتخاب نحوه نصب

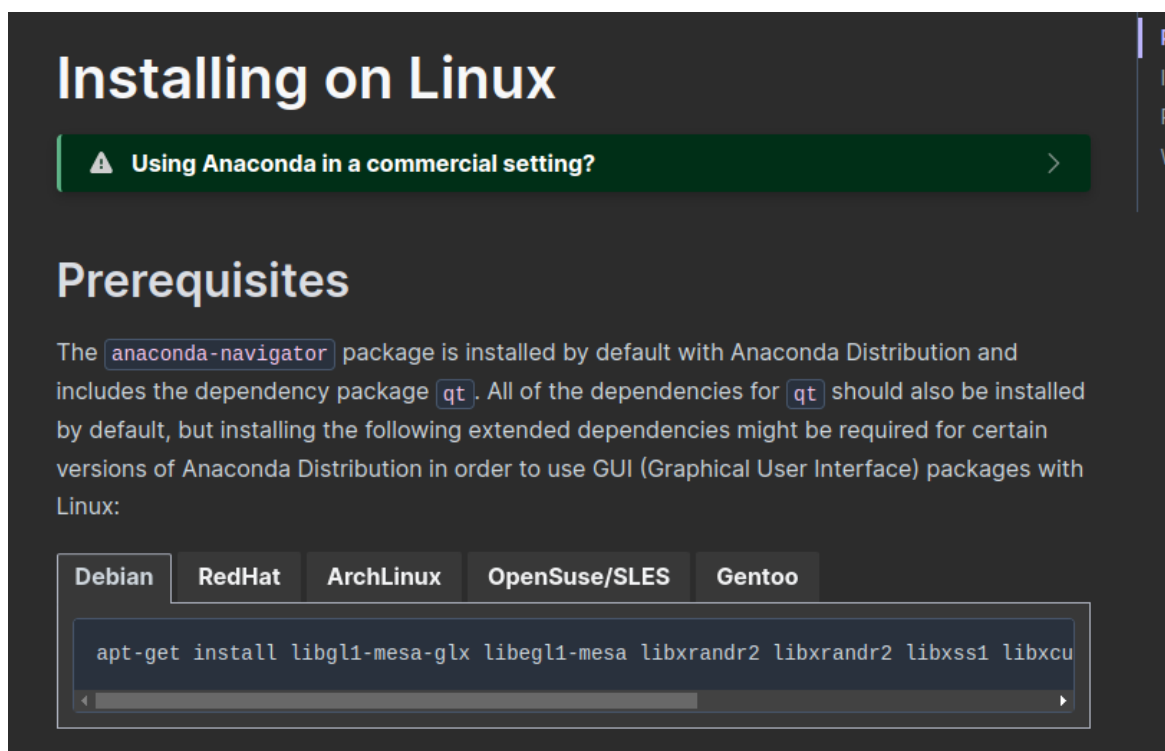
در ادامه باید لوکیشن و مشخصات کاربر وارد شود، ادامه‌ی فرایند نصب نکته‌ی خاصی ندارد و از اشاره به آن اجتناب می‌شود.

۳.۲ نصب anaconda در لینوکس

پیش‌تر برای توسعه‌ی برنامه‌های هوش مصنوعی، anaconda را بر روی لینوکس نصب کرده بودیم و در اینجا فقط به نحوه‌ی نصب آن اشاره می‌شود.

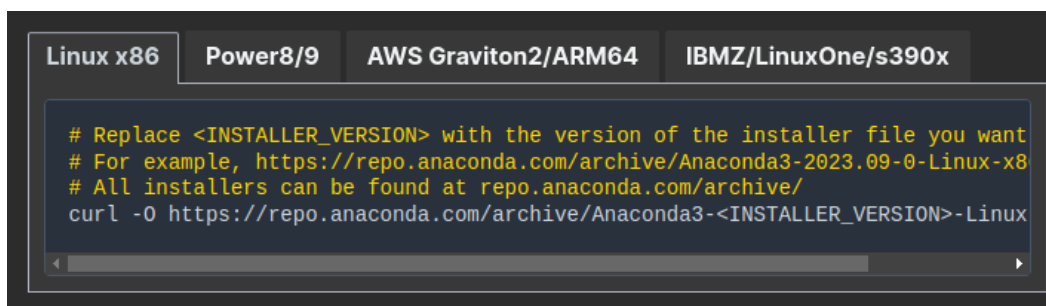
بدین منظور لازم است به سایت آن‌ا‌کوندا مراجعه کرده و راهنمای نصب آن را مشاهده کنیم و گام به گام با آن پیش برویم.

ابتدا لازم است پیش‌نیازهای نصب آناکوندا نصب شوند. (شکل ۲۷)

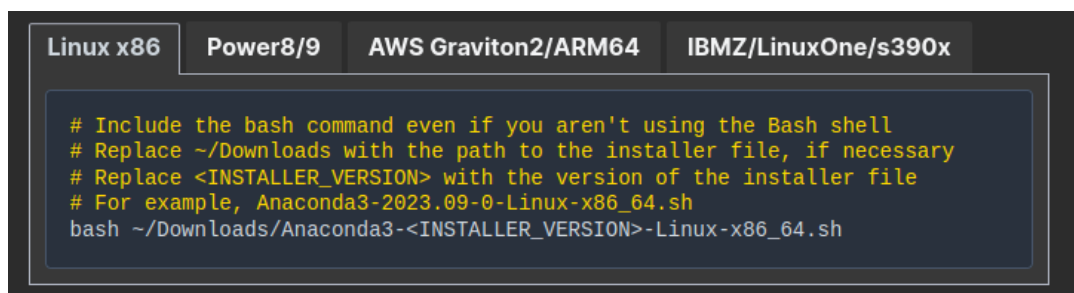


شکل ۲۷: راهنمای نصب آناکوندا، نصب پیش‌نیازها

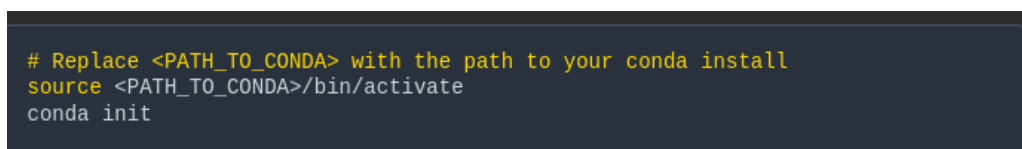
سپس با استفاده از دستورات زیر در ترمینال لینوکس، ابتدا آناکوندا را دانلود کرده و با دستور bash آن را نصب می‌کنیم.



شکل ۲۸: دانلود آناکوندا

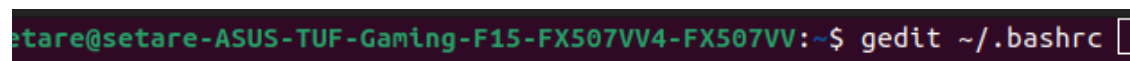


شکل ۲۹: نصب آناکوندا



شکل ۳۰: اتمام نصب

پس از نصب با استفاده از ترمینال فایل `./bashrc` را باز کرده و خطوطی را به آن اضافه می‌کنیم، که محیط `base` آناکوندا همیشه فعال باشد و نیازی نباشد که همیشه با باز کردن ترمینال جدید آن را `source` کنیم. (شکل ۳۱)



حال همیشه محیط `base` فعال خواهد بود.

```

118
119 export PATH="/usr/local/bin:$PATH"
120 export PATH="/usr/local/cuda-12.1/bin${PATH:+:${PATH}}"
121 export LD_LIBRARY_PATH="/usr/local/cuda-12.1/lib64${LD_LIBRARY_PATH:+:${LD_LIBRARY_PATH}}"
122
123 # >>> conda initialize >>>
124 # !! Contents within this block are managed by 'conda init' !!
125 __conda_setup="$(('/home/setare/anaconda3/bin/conda' 'shell.bash' 'hook' 2> /dev/null)"
126 if [ $? -eq 0 ]; then
127     eval "$__conda_setup"
128 else
129     if [ -f "/home/setare/anaconda3/etc/profile.d/conda.sh" ]; then
130         . "/home/setare/anaconda3/etc/profile.d/conda.sh"
131     else
132         export PATH="/home/setare/anaconda3/bin:$PATH"
133     fi
134 fi
135 unset __conda_setup
136 # <<< conda initialize <<<
137
138 source /opt/ros/humble/setup.bash
139 source /usr/share/colcon_argcomplete/hook/colcon_argcomplete.bash

```

شکل ۳۱: اضافه کردن خطوط مرتبط با آناکوندا به bashrc

در محیط base قبلا gymnasium را نصب کرده بودیم ولی برای اطمینان ابتدا چک می‌کنیم که gymnasium نصب است یا خیر. (شکل ۳۲)

```

(base) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~$ python
Python 3.11.7 (main, Dec 15 2023, 18:12:31) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import gymnasium
>>> 

```

شکل ۳۲: چک نصب gymnasium

با توجه به اینکه پکیج از قبل نصب است، برای طی کردن مراحل نصب در محیط لینوکس، مجدد gymnasium را در یک محیط مجازی که قبلا برای توسعه کارهای بینایی ماشین ساخته بودیم، نصب می‌کنیم.

محیط را مطابق شکل ۳۳ فعال کرده و اول چک می‌کنیم که پکیج نصب است یا خیر، که نصب نیست و می‌توان فرایند نصب آن را در محیط لینوکس مجدد انجام داد.

```
(base) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~$ python
Python 3.11.7 (main, Dec 15 2023, 18:12:31) [GCC 11.2.0] on linux
(base) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~$ conda activate Tracking
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~$ python
Python 3.10.13 (main, Sep 11 2023, 13:44:35) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import gymnasium'
      File "<stdin>", line 1
        import gymnasium'
              ^
SyntaxError: unterminated string literal (detected at line 1)
>>> □
```

شکل ۳۳: چک نصب gymnasium در محیط بینایی

با استفاده از دستور زیر پکیج را با تمام محیط‌هایش نصب می‌کنیم.

```
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~$ pip install gymnasium[all]
```

فرایند نصب در لینوکس مانند ویندوز با خطا مواجه شد (شکل ۳۴)، که این خطا بخاطر محیط‌های box2d است و لازم است مجدداً در لینوکس نیز swig نصب شود. با استفاده از دستور زیر نصب آن را انجام می‌دهیم.

```
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~$ conda install swig
```

```

setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV: ~
$ pip install gymnasium
Collecting gymnasium
  Downloading gymnasium-0.29.1-py3-none-any.whl (953 kB)
Collecting ale-py-0.8.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.7 MB)
  Downloading ale-py-0.8.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.7 MB)
Collecting decorator-4.4.2-py2.py3-none-any.whl (9.2 kB)
Collecting fasteners-0.19-py3-none-any.whl (18 kB)
Collecting glfw-2.7.0-py2.py27.py3.py38.py39.py310.py311.py312.py313.py314.py315.py36.py37.py38-none-manylinux2014_x86_64.whl (211 kB)
Collecting inageto_ffmpeg-0.5.1-py3-none-manylinux2010_x86_64.whl (26.9 MB)
  Downloading inageto_ffmpeg-0.5.1-py3-none-manylinux2010_x86_64.whl (26.9 MB)
Collecting ml_dtypes-0.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.5 MB)
  Downloading ml_dtypes-0.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.5 MB)
Collecting proglog-0.1.10-py3-none-any.whl (6.4 kB)
Collecting opt_einsum-3.4.0-py3-none-any.whl (71 kB)
Collecting pygame-2.6.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (14.0 MB)
  Downloading pygame-2.6.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (14.0 MB)
Collecting pyopengl-3.1.7-py3-none-any.whl (2.4 MB)
  Downloading pyopengl-3.1.7-py3-none-any.whl (2.4 MB)
Collecting importlib_resources-6.4.5-py3-none-any.whl (36 kB)
  Downloading importlib_resources-6.4.5-py3-none-any.whl (36 kB)
Building wheels for collected packages: box2d-py, moviepy
Building wheel for box2d-py (setup.py) ... error
error: subprocess-exited-with-error

python setup.py bdist_wheel did not run successfully.
exit code: 1
[so lines of output]
Using setuptools (version 68.2.2).
running bdist_wheel
running build
running build_py
creating build/lib.linux-x86_64-cpython-310
creating build/lib.linux-x86_64-cpython-310/Box2D
copying library/Box2D/Box2D.py -> build/lib.linux-x86_64-cpython-310/Box2D
copying library/Box2D/_init_.py -> build/lib.linux-x86_64-cpython-310/Box2D
creating build/lib.linux-x86_64-cpython-310/Box2D/b2
copying library/Box2D/b2/_init_.py -> build/lib.linux-x86_64-cpython-310/Box2D/b2
running build_ext
building 'Box2D.Box2D' extension
swigging Box2D/Box2D.t to Box2D/Box2D_wrap.cpp
swig -python -c++ -IBox2D -small -O -includeall -ignoremissing -w201 -globals b2Globals -outdir library/Box2D -keyword -xS11 -D_SWIG_KMARGS -o Box2D/Box2D_wrap.cpp Box2D/Box2D.t
error: command 'swig' failed: No such file or directory
[end of output]

note: This error originates from a subprocess, and is likely not a problem with pip.
Running setup.py clean for box2d-py
Building wheel for moviepy (setup.py) ... done
Created wheel for moviepy: filename=moviepy-1.0.3-py3-none-any.whl size=110721 sha256=7b55fbcfc5488c1daee5de7a5c1e6dec8e06a75a39d25e13fb120d340bce95
Stored in directory: /home/setare/.cache/pip/wheels/9e/32/2d/e10123b088fbc02fed53cc18c8a171d3c87479ed045a7c1
Successfully built moviepy
Failed to build box2d-py
WARNING: You are using pip version 23.0.1, but the recommended pip version is 23.1.2.
(Python 3.10.13)
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV: ~$ conda install swig

```

شکل ۳۴: خطا در نصب gymnasium در محیط لینوکس

پس از نصب swig مجدد دستور نصب gymnasium را اجرا می‌کنیم، مطابق خروجی زیر نصب با موفقیت انجام شد.

```

Successfully installed ale-py-0.8.1 box2d-py-2.3.5 cython-0.29.37 decorator-4.4.2 etils-1.9.4 farana-notifications-0.0.4 fasteners-0.19 glfw-2.7.0 gymnasium-0.29.1 inageto-ffmpeg-0.5.1 importlib-resources-6.4.5 jax-0.4.33 jaxlib-0.4.33 ml-dtypes-0.5.0 moviepy-1.0.3 mujoco-py-2.1.2.14 numpy-1.26.4 opt-einsum-3.4.0 proglog-0.1.10 pygame-2.6.0 pyopengl-3.1.7 shimmy-0.2.1 swig-4.2.1
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV: ~$ python

```

شکل ۳۵: نصب موفقیت آمیز gymnasium در لینوکس

مطابق شکل ۳۶، چک می‌کنیم که آیا نصب موفقیت آمیز بوده یا خیر.

```

(base) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV: ~/Reinforcement Learning/Week 1/HWs/HW1$ conda activate Tracking
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV: ~/Reinforcement Learning/Week 1/HWs/HW1$ python
Python 3.10.13 (main, Sep 11 2023, 13:44:35) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> import gymnasium
>>>
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV: ~/Reinforcement Learning/Week 1/HWs/HW1$

```

شکل ۳۶: چک نصب gymnasium در محیط لینوکس

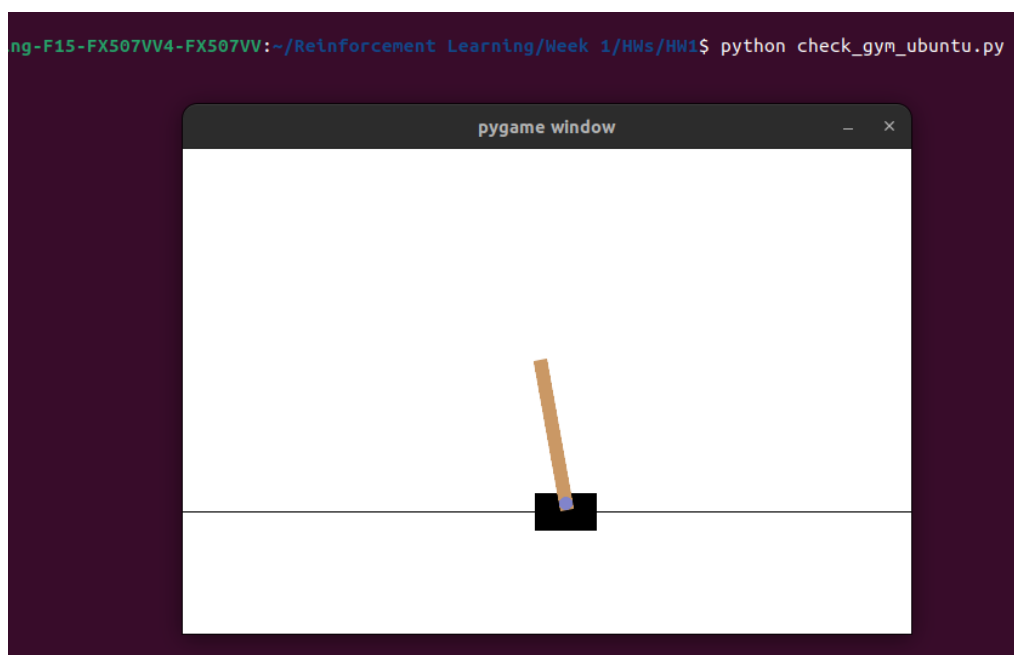
به صورت زیر فایل کد را ایجاد می‌کنیم.

```
>>>
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~/Reinforcement Learning/Week 1/HWs/HW1$ touch check_gym_ubuntu.py
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~/Reinforcement Learning/Week 1/HWs/HW1$ code .
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~/Reinforcement Learning/Week 1/HWs/HW1$
```

شکل ۳۷: ایجاد کد

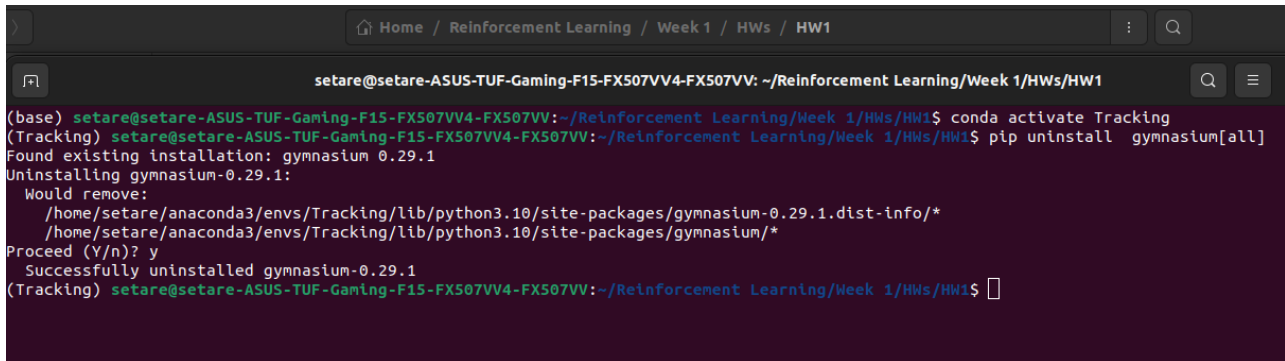
۳.۳ اجرای کد نمونه در اینوکس

مشابه همان کدی را که در ویندوز نوشتیم اینجا می‌نویسیم و اجرا می‌کنیم. خروجی به صورت زیر است که نمایانگر اجرای صحیح کد و نصب کامل پکیج است.



شکل ۳۸: خروجی gymnasium و محیط cart pole در لینوکس

با توجه به عدم سازگاری سایر پکیج‌های محیط tracking با gymnasium و حیاتی بودن آن‌ها، پکیج gymnasium از این محیط حذف و ورژن سایر کتابخانه‌ها را به حالت قبل بازگرداندیم، و زین پس در همان محیط base از gymnasium که قبلاً بر روی آن نصب شده بود، استفاده می‌کنیم.



```
setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV: ~/Reinforcement Learning/Week 1/HWs/HW1
(base) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~/Reinforcement Learning/Week 1/HWs/HW1$ conda activate Tracking
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~/Reinforcement Learning/Week 1/HWs/HW1$ pip uninstall gymnasium[all]
Found existing installation: gymnasium 0.29.1
Uninstalling gymnasium-0.29.1:
  Would remove:
    /home/setare/anaconda3/envs/Tracking/lib/python3.10/site-packages/gymnasium-0.29.1.dist-info/*
    /home/setare/anaconda3/envs/Tracking/lib/python3.10/site-packages/gymnasium/*
Proceed (Y/n)? y
  Successfully uninstalled gymnasium-0.29.1
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~/Reinforcement Learning/Week 1/HWs/HW1$
```

شکل ۳۹: حذف gymnasium

فصل ۴: اجرای mountain car

۴.۱ اجرای کد mountain car

در این بخش محیط مجازی tracking را غیرفعال کرده به محیط base می‌رویم و مطابق دستور زیر کد پایتون را ایجاد می‌کنیم.

```
>>> exit()
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~/Reinforcement Learning/Week 1/HWs/HW1$ touch mountain_car.py
(Tracking) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~/Reinforcement Learning/Week 1/HWs/HW1$ conda deactivate
(base) setare@setare-ASUS-TUF-Gaming-F15-FX507VV4-FX507VV:~/Reinforcement Learning/Week 1/HWs/HW1$
```

شکل ۴۰: بازگشت به base و ساخت کد پایتون

خطوط کد نیز به صورت شکل ۴۱ می‌باشد.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

"""
Course: Reinforcement Learning in Control
Semester: 4031 | Fall 2024
Student: STRH
Created on Thu, 26 Sept 2024, 20:49:00
HW1: Test Mountain Car Environment
"""

import gymnasium as gym
from alive_progress import alive_bar

env = gym.make("MountainCar-v0", render_mode="human")
observation, info = env.reset(seed=42)
with alive_bar(10000) as bar:
    for _ in range(10000):
        action = env.action_space.sample()
        observation, reward, terminated, truncated, info = env.step(action)

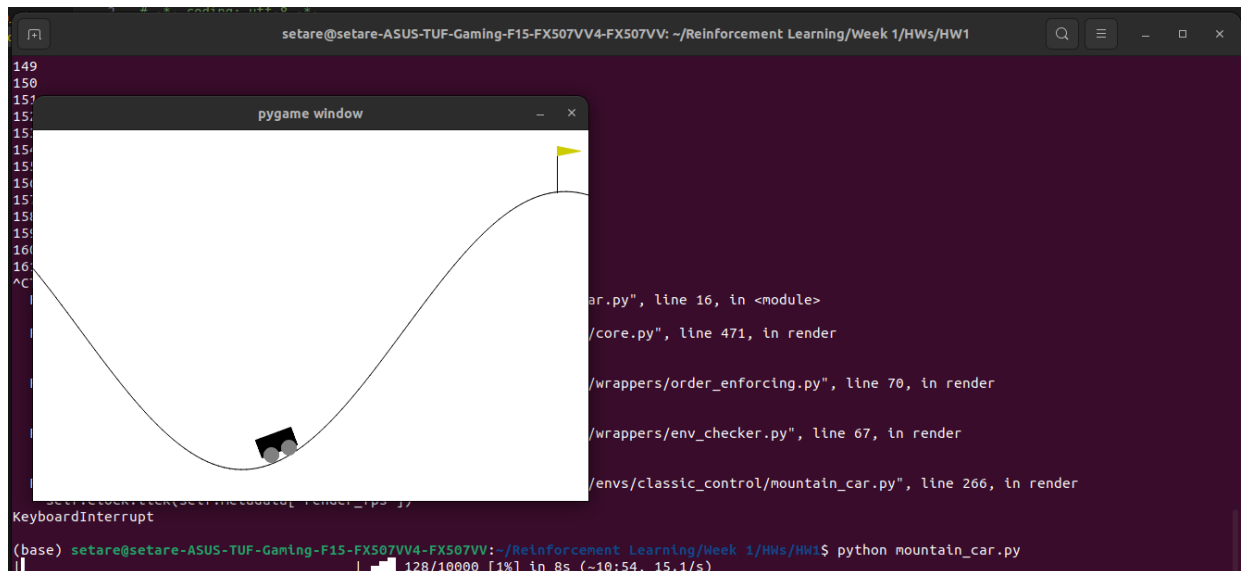
        if terminated or truncated:
            observation, info = env.reset()

    bar()

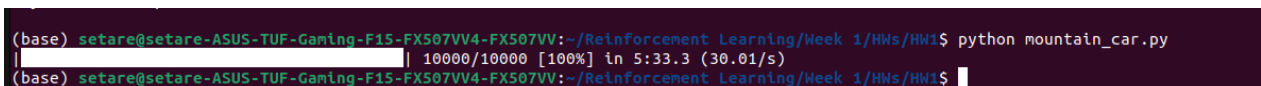
env.close()
```

شکل ۴۱: کد اجرای mountain car

روند کد مانند کدهای قبلی است که استفاده کردیم و توضیحات کامل آن در بخش اجرای کد نمونه بر روی ویندوز موجود است، در اینجا فقط تعداد iteration ها به ۱۰۰۰۰ افزایش یافته است و یک bar نیز اضافه کردیم تا در ترمینال فرایند اجرای عمل‌های تصادفی را مشاهده کنیم.



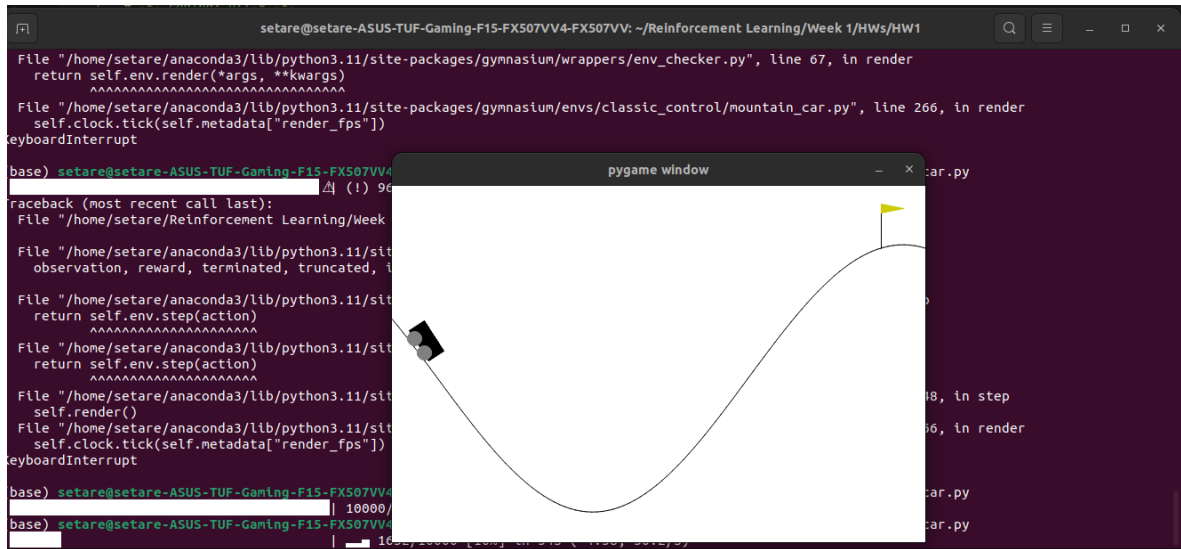
شکل ۴۲: خروجی ۱۰۰۰۰ عمل تصادفی در محیط لینوکس



شکل ۴۳: اتمام اجرای کد

محیط mountain car continuous را که action ها در آن به صورت continous است، برخلاف قبلی که گسسته است، نیز اجرا گرفتیم و خروجی مطابق تصویر ۴۴ است.

اجرای کد mountain car

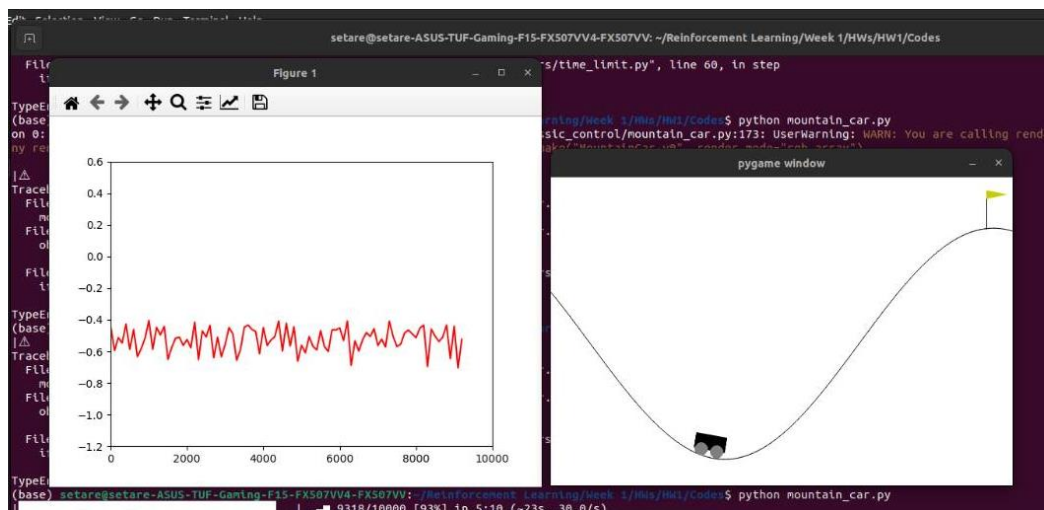


شکل ۴۴: اجرای مجدد mountain car حالت continuous

در این بخش کد mountain car را کمی تغییر دادیم، ابتدا به آن نمودار interactive اضافه کردیم، که مولفه‌ی افقی مکان ماشین را برحسب تعداد episodeها در هنگام پیش‌روی حلقه‌ی اجرای برنامه به صورت فعال رسم کند و برای داشتن کدی بهتر آن را به صورت شیء گرا نوشتیم.

خروجی به صورت زیر خواهد بود:

(نرخ به روزرسانی نمودار هر ۱۰۰ episode یکبار است)



شکل ۴۵: خروجی mountain car در هنگام اجرا

ویدیوی کوتاهی از روند اجرای این کد تهیه شده است که در فولدر videos قرار گرفته است (نرخ آپدیت نمودار در این ویدیو هر ۱۰ اپیزود یک بار است).

تمامی فایل‌های تمرین نیز در [یک ریپازیتوری](#) موجود است که پس از اتمام مهلت تمرین به صورت public قابل نمایش خواهد بود.