به نام او

آموزش برنامه‌نویسی

PYTHON 3.12

مقدمه‌ای بر اصول پایه‌ای پایتون
(ویرایش چهارم)

فابریزیو رومانو
هنریش کراکر

ترجمه: مهندس حسین یعسوبی
انتشارات پندار پارس

سرشناسه : رومانو، فابریتسیو، ۱۹۷۵- م.Romano, Fabrizio, 1975-

عنوان و نام پدیدآور : آموزش برنامه نویسی Python 3.12: مقدمه‌ای بر اصول پایه‌ای پایتون(ویرایش چهارم)/ فابریزیو رومانو، هنریش کراکر؛ ترجمه حسین یعسوبی.

مشخصات نشر : تهران : پندار پارس، ۱۴۰۴.

مشخصات ظاهری : ۵۷۲ ص.

شابک : 978-622-7785-52-4

وضعیت فهرست نویسی : فیپا

یادداشت : عنوان اصلی: Learn python programming : An in-depth introduction to the fundamentals of python, 4rd.ed, 2024.

عنوان دیگر : مقدمه‌ای بر اصول پایه‌ای پایتون(ویرایش چهارم).

موضوع : پایتون (زبان برنامه‌نویسی کامپیوتر) Python (Computer program language)

شناسه افزوده : گروگر، هاینریک، ۱۹۸۱- م. Kruger, Heinrich, 1981-

شناسه افزوده : یعسوبی، حسین، ۱۳۵۲ -، مترجم

رده بندی کنگره : ۷۳/QA۷۶

رده بندی دیویی : ۱۳۳/۰۰۵

شماره کتابشناسی ملی : ۱۰۲۶۳۵۷۳

انتشارات پندارپارس

دفتر فروش: انقلاب، ابتدای کارگرجنوبی، کوی رشتچی، شماره ۱۴، واحد ۱۶ **www.pendarepars.com**

تلفن: ۶۶۵۷۲۳۳۵ - تلفکس: ۶۶۹۲۶۵۷۸ همراه: ۰۹۱۲۲۴۵۲۳۴۸ **info@pendarepars.com**

نام کتاب	: **آموزش برنامه‌نویسی Python 3.12، مقدمه‌ای بر اصول پایه‌ای پایتون**
ناشر	: **انتشارات پندار پارس**
تالیف	: **فابریزیو رومانو، هنریش کراکر (Fabrizio Romano, Heinrich Kruger)**
ترجمه	: **حسین یعسوبی**
چاپ نخست	: **آبان ۱۴۰۴ شمارگان : ۲۰۰ نسخه چاپ، صحافی : وحید**
قیمت	: **۶۷۰.۰۰۰ تومان شابک : ۴-۵۲-۷۷۸۵-۶۲۲-۹۷۸**

سخن مترجم (ناشر)

کتابی که در دست دارید، چهارمین ویرایش از کتاب پایتون "فابریزیو رومانو" است که همچون نسخه ۳.۹، به‌کمک هنریش کراکر، آخرین به‌روزرسانی‌ها را در محتویات کتاب و کدهای آن پیاده کرده‌اند.

این کتاب نظر بسیاری از متخصصان را به خود جلب کرده است؛ چرا که برنامه‌نویسی پایتون را از سطح مبتدی آغاز کرده و تا سطح تقریبا پیشرفته ادامه داده و در یک فصل نیز به کتابخانه‌های علم داده پرداخته است. بنابراین اگر بگویم ترجمه بهترین کتاب آموزش برنامه‌نویسی پایتون موجود است، گزاف نگفته‌ام. با کمی گشت و گذار در وب، به این نتیجه خواهید رسید.

پایتون زبانی است که جای خود را به‌خوبی در همه زمینه‌های فنآورانه، دانشگاهی، کسب‌وکار و فضای مجازی باز کرده است و کمتر کسی است که این را نداند. بر خود لازم دانستم این اثر را به بهترین شکل ممکن ترجمه کنم تا شاید بتواند راهگشای برنامه‌نویسان و دانشجویان و حتی دانش‌آموزانی که قصد فراگیری یک زبان همه‌کاره را دارند باشد. هرچند، هیچ اثری به‌دور از اشکال نیست. کافیست بر ما منت نهید و نظرها و یا اشکال‌های احتمالی که از چشم ما به‌دور مانده را در صفحه این کتاب در سایت انتشارات پندارپارس مطرح فرمایید.

حسین یعسوبی

مدیر مسئول انتشارات پندار پارس

پاییز ۱۴۰۴

فهرست

پیش‌گفتار

با کمال افتخار، چهارمین ویرایش از کتابمان را تقدیم شما می‌کنیم. احساس می‌کنم همین دیروز بود، اما در اصل، ۱۰ سال پیش بود (۲۰۱۵). در طی چند هفته، کتاب جزو پرفروش‌ترین کتاب‌ها شد، و تا به امروز، خوانندگان آن با دوست‌داشتنی‌ترین پیام‌ها و ایمیل‌ها از سرتاسر جهان، مرا مورد لطف خود قرار داده و می‌دهند.

ده سال پیش، معنای برنامه‌نویس شدن، کار کردن با ابزارهایی مشخص، و پیاده‌سازی یک الگوی نمونه بود. امروز، دورنما متفاوت است. توسعه‌دهندگان گرایش دارند حتی تخصصی‌تر از پیش عمل کنند، و تمرکز بیشتری روی مواردی همچون APIها، و اپلیکیشن‌های توزیع‌شده وجود دارد. ما هم تلاش کرده‌ایم این گرایش‌های جاری را گلچین کنیم و با بهره‌گیری از تجربیات‌مان به‌عنوان توسعه‌دهنده‌هایی که در صنعتی که به‌سرعت به‌روز می‌شود کار می‌کنند، آنرا در بهترین لایه عملکردی ممکن ارائه دهیم.

هر ویرایش جدید، برخی از انواع تغییرات را به‌همراه داشت. فصل‌های منسوخ شده، حذف و فصل‌های جدیدی افزوده شد تا روش‌های مدرنی که نرم‌افزار در آن نوشته شده را منعکس کند.

در این ویرایش، سه فصل جدید افزوده شده است. نخستین آن، درباره عنوان اشاره‌گر نوع (type hinting) بحث می‌کند. این موضوع جدیدی در پایتون نیست، اما اکنون احساس می‌کنیم اجرای این کار به یک سنت جاافتاده تبدیل شده، طوری که بدون آن کتاب ناقص به نظر می‌رسید.

دومین مورد آن، سرفصل اپلیکیشن‌های CLI است، که گام‌های آن را جایگزین فصل قدیمی که درباره GUIها بود کردیم. بیشترین کاری که امروزه با یک کامپیوتر انجام می‌دهیم در یک مرورگر است، و اپلیکیشن‌های دسکتاپ زیادی با تکیه بر کامپوننت‌های مرورگر ایجاد یا بازنویسی شده‌اند، پس احساس کردیم که یک فصل کامل که به GUIهایی پرداخته که شاید کمی منسوخ شده باشد.

در پایان، سومین فصل نیز به مبحث برنامه‌نویسی رقابتی می‌پردازد.

بقیه فصل‌ها نیز به‌روز شده است تا منعکس‌کننده آخرین موارد افزوده شده به زبان باشد، و طوری بهبود یافته که حتی ساده‌تر و روان‌تر ارائه داده شود، درحالی‌که هنوز مثال‌های جالبی برای خوانندگان پیشنهاد می‌دهد.

جوهره و روح کتاب هنوز دست‌نخورده باقی‌مانده است. یعنی احساس نمی‌شود کتاب پایتون دیگری است. این بیشتر درباره برنامه‌نویسی است. چون تلاش شده است تاحد امکان اطلاعات بیشتری را پوشش دهد و گاهی حتی محدودیت شمارش صفحه اجازه این را نمی‌داد و مجبور شدیم خواننده را به منابع خارجی ارجاع دهیم.

این نسخه طوری طراحی شده که ماندگار باشد. مفاهیم و اطلاعات را در مسیری توضیح می‌دهد که باید تا حد امکان، در گذر زمان، پایدار و بدون تغییر بماند. بنابرین کار، تفکر، و جلسات زیادی داشتیم تا به این خواسته‌ها برسیم.

همچنین تفاوت چشمگیری با نسخه‌ی اول خود دارد. بسیار پخته‌تر، حرفه‌ای‌تر شده و تمرکز بیشتری بر زبان دارد و کمی از پروژه‌ها فاصله گرفته است. ما معتقدیم که خط تعادل میان این دو بخش به‌درستی ترسیم شده است.

لازم است به سختی کار کنید. کدها برای دانلود در دسترس شماست و توصیه می‌کنیم با آنها کار کنید، آزمایش کنید، تغییر دهید، بسط دهید، بشکنید، و به چیزهایی دست یابید. لازم است آنها را از GitHub بردارید. دوست داریم موارد حیاتی را توسعه دهید. می‌خواهیم شما در کدنویسی، مستقل و صاحب اختیار باشید. امیدواریم در هر کجا هستید، این کتاب به کمک شما بیاید و دست‌کم، به برنامه‌نویس بهتری تبدیل شوید.

هدفم این بود که این کتاب، تنها درباره زبان نباشد بلکه درباره برنامه‌نویسی باشد. در واقع، هنر برنامه‌نویسی، چند دیدگاه را دربر می‌گیرد و زبان، تنها یکی از آنهاست. جنبه قاطع دیگر برنامه‌نویسی، استقلال است. توانایی متوقف نکردن خودتان وقتی به دیوار بلندی برخورد می‌کنید و نمی‌دانید برای حل مشکل موجود، چه کنید. کتابی برای آموزش آن وجود ندارد و فکر کردم به‌جای اینکه سعی کنم این جنبه را آموزش دهم، با آزمایش کردن، خوانندگان را در این باره آموزش دهم.

کتاب برای چه کسانی مفید است

پایتون، مشهورترین زبان آموزشی مقدماتی در دانشگاه‌های تراز بالای علوم رایانه در ایالات متحده است، پس اگر تازه وارد دنیای برنامه‌نویسی شده‌اید یا اگر تجربه اندکی در برنامه‌نویسی با هر زبانی دارید و دوست دارید در این راه گام بگذارید، این زبان و این کتاب، آن چیزی است که به آن نیاز دارید.

اگر پیش‌تر با پایتون یا زبان دیگری کار کرده‌اید، باز هم این کتاب برایتان مفید است، هم به‌عنوان یک مرجع بنیادی و اصلی و هم برای ارائه تجربه‌ها و پیشنهادهای بسیار گسترده‌ای که در دو دهه اخیر، به‌دست آمده و در این کتاب در اختیارتان می‌گذاریم.

محتویات کتاب

فصل ۱، معرفی کوتاه پایتون؛ با مفاهیم بنیادی برنامه‌نویسی پایتون آشنا می‌شوید. شما را تا بالا آوردن و اجرای پایتون روی سیستم‌تان راهنمایی می‌کنیم و با چند دستور آن آشنا خواهید شد.

فصل ۲، Data Typeهای درون–ساخت؛ انواع داده‌های پایتون که به‌شکل پیش‌ساخته در آن موجود است را معرفی می‌کند. پایتون دارای مجموعه بسیار گسترده‌ای از انواع داده‌های بومی و محلی است و این فصل، با یک مثال کوتاه، هر یک از آنها را توصیف می‌کند.

فصل ۳، شروط و تکرار؛ می‌آموزد با بازرسی شرط‌ها، اعمال منطق، و پیاده‌سازی حلقه‌ها، چگونه جریان کد را کنترل کنیم.

فصل ٤، توابع، ساخت بلوک‌های کد؛ شیوه نوشتن توابع را آموزش می‌دهد. توابع، کلیدهایی برای استفاده دوباره از کدها و کاهش زمان دیباگ کردن است و عموما برای نوشتن بهتر کد است.

فصل ٥، خلاصه‌لیست‌ها و مولدها؛ دیدگاه‌های عملیاتی برنامه‌نویسی پایتون را به شما معرفی می‌کند. این فصل، شیوه نوشتن خلاصه‌لیست‌ها و مولدها را می‌آموزد که ابزارهای مفیدی است که می‌توان برای افزایش سرعت کد و ذخیره حافظه، از آنها استفاده کرد.

فصل ٦، OOP، Decorators، و Iterators؛ مبانی برنامه‌نویسی شئ‌گرا با پایتون را آموزش می‌دهد. مفاهیم کلیدی و همه پتانسیل‌های این پارادایم را نشان می‌دهد. همچنین، یکی از محبوب‌ترین ویژگی‌های پایتون، یعنی دکوراتورها را نشان می‌دهد. در آخر نیز مفاهیم تکرارشدنی‌ها یا همان تکرارکننده‌ها را پوشش می‌دهد.

فصل ۷، استثناها و مدیران محتوا؛ مفهوم استثناها را معرفی می‌کند که بیانگر خطاهایی است که در برنامه رخ می‌دهد و چگونگی رسیدگی به آنها را مطرح می‌کند. همچنین، با مفهوم مدیرها آشنا می‌شوید که در کار با منابع، بسیار مفید است.

فصل ۸، ماندگاری فایل‌ها و داده‌ها؛ می‌آموزد چگونه با فایل‌ها، جریان‌ها، فرمت‌های تبادل داده‌ها و دیتابیس‌ها سروکار داشته باشیم.

فصل ۹، رمزنگاری و توکن‌ها؛ به مفاهیم امنیت، هش‌ها، رمزنگاری، و توکن‌ها می‌پردازد که برای نوشتن نرم‌افزار امن، الزامی است.

فصل ۱۰، آزمایش؛ روش‌های اصلی تست کد را با ارائه مثال‌هایی از نحوه پیاده‌سازی آن نشان می‌دهد تا به کدی قوی‌تر، سریع‌تر، و قابل اتکاتر برسیم.

فصل ۱۱، دیباگ‌کردن و رفع اشکال؛ روش‌های اصلی دیباگ‌کردن کد را با ارائه مثال‌هایی از نحوه پیاده‌سازی آن نشان می‌دهد.

فصل ۱۲، مقدمه‌ای بر Type Hinting، شما را با سینتکس و مفاهیم اصلی نوع‌گذاری آشنا می‌کند. نوع‌گذاری در سال‌های اخیر، مشهور و مشهورتر شده است، زیرا هم زبان و هم ابزارهایی که بخشی از زیست‌بوم آن است را غنی می‌کند.

فصل ۱۳، علم داده؛ چند مفهوم کلیدی و یک ابزار بسیار ویژه را معرفی می‌کند، Jupyter Notebook.

فصل ١٤، مقدمه‌ای بر توسعه API؛ اصول بنیادی توسعه API را با استفاده از فریمورک FastAPI بیان می‌دارد.

فصل ۱۵، اپلیکیشن‌های CLI؛ به معرفی اپلیکیشن‌های اینترفیس خط فرمان می‌پردازد. اینها در یک کنسول یا ترمینال ران می‌شوند و روش مرسوم و طبیعی است که توسعه‌دهندگان، چندین ابزار شخصی خودشان را روزانه در آن بنویسند.

فصل ١٦، پکیج‌کردن برنامه‌های پایتون؛ شما را با فرایند آماده‌سازی یک پروژه برای انتشار آشنا می‌سازد و نشان می‌دهد چگونه نتیجه را روی Python Package Index (PyPI) بارگزاری کنید.

فصل ۱۷، چالش‌های برنامه‌نویسی؛ با نشان دادن نحوه حل دو مشکل از وب‌سایت Advent of Code، شما را مفهوم برنامه‌نویسی رقابتی آشنا می‌کند.

بهره‌برداری بیشتر از کتاب

با دنبال کردن مثال‌های کتاب، آموزش بهتری می‌گیرید. به این منظور، به یک رایانه، یک ارتباط اینترنت، و یک مرورگر نیاز دارید. **کتاب برای نسخه ۳.۱۲ پایتون نوشته شده، اما بیشتر بخش‌های آن با هر نسخه جدید Python 3.* باید کار کند.** راهنمای نصب پایتون روی سیستم‌عامل هم ارائه شده است. رویه‌های نصب همیشه درحال تغییر است، پس لازم است به به‌روزترین راهنمای نصب که در وب یافت می‌شود مراجعه شود. همچنین شیوه نصب کتابخانه‌های اضافه به‌کار رفته در فصل‌های مختلف را توضیح داده‌ایم و درصورت برخورد با هر مشکلی حین نصب آنها، پیشنهادهایی نیز ارائه کرده‌ایم. برای تایپ کد، نیاز به هیچ ویرایشگر خاصی نیست؛ هرچند، پیشنهاد می‌شود آنهایی که در مثال‌های پیش رو معرفی شده است را مدنظر داشته باشید تا متناسب با محیط کدنویسی ما باشد. در فصل یک پیشنهادهایی برای این موضوع ارائه شده است.

دانلود فایل‌های کد مثال‌ها

فایل‌های کد مثال‌های کتاب را از سایت انتشارات پندار پارس (صفحه این کتاب، برگه سورس‌کد و ضمائم) بردارید. همچنین، در GitHub به نشانی زیر نیز موجود است:

https://github.com/PacktPublishing/Learn-Python-Programming-Fourth-Edition

همچنین در نشانی زیر، مجموعه‌ای از کتاب‌ها و ویدئوهای مرتبط با کتاب را برایتان گذاشته‌ایم:

https://github.com/PacktPublishing/

ضمنا توجه داشته باشید که ورودی و خروجی‌های خط فرمان (کامندلاین) را به‌صورت زیر، یعنی با سه فلش مقابلشان، آورده‌ایم:

```
>>> import sys
>>> print(sys.version)
```

فصل ۱

معرفی کوتاه پایتون

به اختصار، برنامه‌نویسی کامپیوتری یا همان کدنویسی، به رایانه می‌گوید با استفاده از یک زبانی که می‌شناسد، کاری را انجام دهد. رایانه‌ها ابزارهای بسیار مفیدی هستند، اما شوربختانه نمی‌توانند برای خودشان فکر کنند. آنها نیاز دارند هر چیزی به آنها گفته شود: چگونه یک کار انجام شود، چگونه برای تصمیم‌گیری برای مسیری که باید دنبال شود، شرطی را بررسی کند، چگونه داده‌هایی که از یک وسیله (دیوایس) می‌آید رسیدگی شود، مانند شبکه یا یک دیسک، و وقتی مورد پیش‌بینی نشده‌ای رخ دهد چه واکنشی نشان داده شود؛ مثلا عملیاتی شکست بخورد یا گم شود.

به سبک‌ها و زبان‌های گوناگونی می‌توان کدنویسی کرد. چه بسیاری از آنها سخت و پیچیده هستند. هر کسی می‌تواند چگونگی کدنویسی را بیاموزد و شما نیز می‌توانید. اما، اگر می‌خواستید شاعر شوید آیا نوشتن به تنهایی کافی بود؟ مجبور بودید یک سری مهارت کسب کنید و تلاش بیشتری انجام دهید و طبع شعر هم که زیربنای کار است.

در آخر، همه چیز بستگی به این دارد که قصد دارید در این جاده حرکت کنید. کدنویسی، کنار هم چیدن یک سری دستور که کار کند نیست. چیزی خیلی بیشتر از این است.

کد خوب، باید کوتاه، سریع، زیبا و باسلیقه باشد، خواندن و فهم آن آسان باشد، بسط و تغییرش ساده باشد، پیمایش و تعمیرش ساده باشد، و به آسانی آزمایش شود. نوشتن کدی که هم‌زمان، همه این موارد کیفی را داشته باشد نیاز به زمان و کسب تجربه بالا دارد، اما خبر خوب این است که با خواندن این کتاب، نخستین گام به سمت این ایده‌آل‌ها را خواهید پیمود و تردیدی ندارم که از پس آن بر می‌آیید. در حقیقت، هرکسی می‌تواند، همه ما همیشه درحال برنامه‌نویسی هستیم، تنها از آن آگاه نیستیم. به مثال زیر توجه کنید:

فرض کنید می‌خواهید یک فنجان قهوه دم کنید. برای این کار به یک فنجان، یک ظرف قهوه‌دان، یک قاشق، آب، و کتری نیاز دارید. حتی اگر این کار را بلد نباشید، سعی می‌کنید مقداری داده را ارزیابی کنید. مطمئن می‌شوید آب در کتری هست و زیر کتری روشن است، فنجان تمیز است، و قهوه کافی در قهوه‌دان موجود است. سپس، آب را جوش می‌آورید و شاید در این حین، مقداری قهوه در فنجان بریزید. وقتی آب جوش آماده شد، آنرا در فنجان می‌ریزید و آنرا با قاشق به‌هم می‌زنید.

برنامه‌نویسی این مثال چگونه است؟

ما منابع را گردآوری کردیم (کتری، قهوه، آب، قاشق، و فنجان) و برخی شرایط مربوط به آنها را فراهم کردیم (زیر کتری را روشن کردیم، تمیز بودن فنجان، و وجود داشتن قهوه). سپس دو کار را آغاز کردیم (جوش آوردن آب و ریختن قهوه در فنجان)، و هنگامی که هر دوی آنها انجام شد، با ریختن آب در فنجان و هم زدن، رویه را به پایان رساندیم.

می‌توانید آنرا ببینید؟ تنها یک کارکرد سطح بالای یک برنامه قهوه را توضیح دادیم. کار دشواری نبود اما این همان کاری است که مغز ما هر روز انجام می‌دهد: ارزیابی شرط‌ها، تصمیم برای انجام کارها، انجام وظایف، تکرار برخی از آنها، و توقف در برخی نقاط. اشیاء را تمیز کن، آنها را سرجایشان بگذار، و غیره.

همه آن چیزی که لازم است اکنون بیاموزید این است که همه کارهایی که به‌شکل خودکار در زندگی روزمره خود انجام می‌دهید را خرد کنید تا یک رایانه بتواند برخی از آنها را واقعا درک کند. و لازم است یک زبان را برای ساختن آن، به خوبی فراگیرید.

این کتاب برای همین منظور است. نحوه انجام این کار را خواهیم گفت و تلاش می‌کنیم با تعریف چند مثال ساده اما متمرکز (از نوع مورد علاقه‌ام)، این کار را انجام دهیم.

در این فصل، موارد زیر را پوشش می‌دهیم:

- ویژگی‌ها و زیست‌بوم پایتون
- راهنمایی‌هایی برای شیوه اجرا و کار با پایتون و محیط‌های ویژه
- شیوه اجرای برنامه‌های پایتون
- شیوه سازماندهی کد و مدل اجرایی پایتون

معرفی کوتاه پایتون

هنگامی که به آموزش کدنویسی می‌پردازم دوست دارم به جهان واقعی ارجاع دهم؛ باور دارم که حقایق موجود در جهان واقعی، به شناخت بهتر مفاهیم کمک می‌کند. هرچند، اینک زمان آن رسیده که کمی جدی‌تر باشیم و کدنویسی را از زاویه فنی‌تر ببینیم.

هنگامی که کدی را می‌نویسیم، دستوری را برای انجام کارهایی به رایانه می‌دهیم. اتفاق در کجا می‌افتد؟ در خیلی جاها: حافظه رایانه، درایوهای سخت، کابل‌های شبکه، CPU و غیره. این یک جهان کامل است که بیشتر وقت‌ها بیانگر زیرمجموعه‌ای از جهان واقعی است.

اگر یک قطعه کد نرم‌افزاری بنویسید که امکان خرید آنلاین لباس را به مردم بدهد، مجبورید افراد واقعی، لباس‌های واقعی، برندهای واقعی، اندازه و غیره را درون مرزهایی از یک برنامه، ارائه کنید.

پس برای انجام این کار، نیاز به ایجاد و رسیدگی به اشیاء در برنامه‌ای که می‌نویسید دارید. هر شخص می‌تواند یک شئ باشد. هر ماشین، یک شئ است. یک جفت جوراب هم یک شئ است. خوشبختانه، پایتون اشیاء را به خوبی می‌شناسد.

دو ویژگی اصلی که هر شئ دارد، **مشخصه‌ها** (properties) و **متدها** (methods) است. اجازه دهید یک شخص را به‌عنوان مثال شئ درنظر بگیریم. نوعا در یک برنامه رایانه‌ای، افراد را به‌عنوان مشتری یا کارمند معرفی می‌کنید. مشخصه‌هایی که برای آنها درنظر می‌گیرید چیزهایی مانند نام، SSN، سن، آیا دارای گواهی‌نامه رانندگی هستند، نشانی ایمیل، جنسیت، و غیره است. در یک برنامه رایانه‌ای، همه داده‌هایی که به‌منظور استفاده از یک شئ برای رسیدن به هدف خود نیاز دارید را ذخیره می‌کنید. اگر درحال کدنویسی یک وب‌سایت فروش لباس هستید، شاید بخواهید قد و وزن افراد را هم در کنار دیگر مقیاس‌های مشتریان خود ذخیره کنید تا بتوانید لباس‌های مناسب را به آنها پیشنهاد دهید. پس **مشخصه‌ها، ویژگی‌های یک شئ هستند**. همیشه از آنها استفاده می‌کنیم: می‌توانید آن قلم را به من بدهید؟___ کدام یکی؟___ قلم مشکی را. در اینجا ما از مشخصه مشکی یک قلم برای شناسایی آن استفاده کردیم.

متدها چیزهایی هستند که یک شئ می‌تواند انجام دهد. من شخصا متدهایی مانند صحبت کردن، راه رفتن، خوابیدن، خوردن، خواندن و نوشتن و غیره دارم. **هر آن چیزی که بتوانم انجام دهم می‌تواند به عنوان متدهای اشیائی دیده شود که مرا تعریف می‌کند**.

پس اکنون که می‌دانید اشیاء چیست و اینکه آنها متدهایی که می‌توانید اجرا کنید و مشخصه‌هایی که می‌توانید رسیدگی کنید را عرضه می‌کنند، آماده آغاز کدنویسی هستید. کدنویسی در حقیقت، به‌سادگی مدیریت آن اشیائی است که در زیرمجموعه جهانی که ما در حال بازتولید در نرم‌افزارمان هستیم زندگی می‌کنند. اشیاء را می‌توان به دلخواه خود، ایجاد، استفاده، استفاده مجدد و حذف کرد.

با توجه به فصل مدل داده‌ای در مستند رسمی پایتون:

"اشیاء، تجرد داده‌های پایتون هستند. همه داده‌ها در یک برنامه پایتون، توسط اشیاء یا ارتباط میان آنها بیان می‌شوند."

در فصل ۶ نگاه دقیق‌تری به اشیاء پایتون می‌اندازیم. اینک لازم است بدانید که **هر شیئی در پایتون دارای یک شناسه یا ID، یک نوع، و یک مقدار است**.

ID یک شئ پس از ایجاد هرگز تغییر نمی‌کند. این یک شناساننده یکتا برای آن است و پایتون برای بازیابی شئ وقتی می‌خواهیم از آن استفاده کنیم در پشت صحنه از آن استفاده می‌کند.

نوع یا type هم به هیچ وجه تغییر نمی‌کند. نوع می‌گوید چه عملیاتی توسط شئ پشتیبانی می‌شود و چه مقادیر ممکنی می‌تواند به آن تخصیص یابد. در فصل ۲ مهم‌ترین انواع داده پایتون را خواهیم دید.

مقدار یا value می‌تواند تغییر کند یا نکند. اگر بتواند تغییر کند، به آن شئ تغییرپذیر، ناپایدار، یا mutable می‌گوییم؛ وگرنه به آن تغییرناپذیر، پایدار، یا immutable می‌گوییم.

چگونه می‌توان از یک شئ استفاده کرد؟ البته که یک نام به آن می‌دهیم! با نام‌گذاری آن می‌توانیم هر بار برای بازیابی و استفاده از آن، از آن نام استفاده کنیم. در یک وضعیت عمومی‌تر، اشیائی همچون اعداد، رشته‌ها (متن)، کلکسیون‌ها و غیره، با یک نام مرتبط هستند. معمولا می‌گوییم که این نام، نام یک متغیر است. **متغیر را مانند یک جعبه‌ای ببینید که برای نگهداری داده‌ها می‌توانید استفاده کنید.**

اشیاء، داده‌ها را توصیف می‌کنند. اینک چه؟ خب باید از آنها استفاده کنیم، درست است؟ شاید بخواهیم آنها را از طریق یک ارتباط شبکه‌ای انتقال دهیم یا در یک دیتابیس ذخیره کنیم. شاید آنها را روی یک صفحه وب نمایش دهیم یا آنها را درون یک فایل بنویسیم. به منظور این کار لازم است با پر کردن یک فرم کاربری، یا فشار یک دکمه، یا بازکردن یک صفحه وب و انجام یک جست‌وجو، واکنش نشان دهیم. این کارها را با اجرای کد خود، ارزیابی شرط‌ها برای انتخاب اینکه کدام اجزا اجرا شود، چند بار، و در چه شرایطی، عملی می‌کنیم.

و برای انجام همه اینها، به‌شکل اساسی نیاز به زبان داریم. در اینجاست که پایتون به کارمان می‌آید. پایتون زبانی است که در این کتاب برای دستور دادن به رایانه در انجام کاری برایمان، استفاده خواهیم کرد.

مقدمه چینی کافیست، اجازه دهید وارد گود شویم.

ورود به پایتون

پایتون، تولید شگفت‌انگیز Guido Van Rossum، یک ریاضی‌دان و دانشمند هلندی علوم رایانه است که تصمیم گرفت با پرژه‌ای که در کریسمس ۱۹۸۹ آغاز شد به جهانیان هدیه کند. این زبان در حدود سال ۱۹۹۱ به‌شکل عمومی منتشر شد و امروز، یکی از برترین زبان‌های برنامه‌نویسی در کل دنیاست.

من وقتی ۷ سال داشتم برنامه‌نویسی را روی یک کومودور VIC-20 آغاز کردم که بعدها با برادر بزرگترش یعنی کومودور ۶۴ جایگزین شد. آن زبان، بیسیک بود. سپس به سراغ زبان پاسکال، اسمبلی، C، C++، جاوا، جاوااسکریپت، ویژوال بیسیک، PHP، ASP، ASP.NET، C# و دیگر زبان‌های کوچکی که حتی به‌خاطر نمی‌آورم رفتم، اما تنها وقتی با پایتون آشنا شدم احساس کردم که گمگشده خود را یافته‌ام.

وقتی همه اجزای بدن شما در حال نعره زدن است، و می‌گوید این را بخر! این برایمان بهترین است!

این مرا یاد روزی انداخت که خواستم از آن استفاده کنم. سینتکس آن کمی متفاوت از آنچه استفاده کرده بودم بود، اما پس از گذراندن احساس عجیب آغازین (مانند داشتن لباس‌های نو)، احساس کردم عاشقش شده‌ام. عمیقا. اجازه دهید بگویم چرا.

درباره پایتون

پیش از اینکه وارد جزئیات کار شویم به حسی بپردازم که چرا فردی می‌خواهد از پایتون استفاده کند (پیشنهاد می‌کنم صفحه پایتون در ویکیپدیا را برای جزئیات بیشتر بخوانید).

به نظر من، پایتون دارای نقاط قوت زیر است:

قابل‌حمل بودن

پایتون در هر جایی اجرا می‌شود و حمل‌ونقل یک برنامه از لینوکس به ویندوز یا مکینتاش، معمولا تنها در حد موضوع فیکس کردن مسیرها و تنظیمات است. پایتون برای قابل حمل بودن طراحی شده و در پشت رابط کاربری، مراقب تغییرات ناگهانی سیستم‌عامل مشخص می‌باشد تا مجبور نباشید سختی کدنویسی مربوط به یک پلتفرم مشخص دیگر را به جان بخرید.

انسجام

پایتون به‌شدت منطقی و منسجم است. می‌توانید ببینید که توسط یک دانشمند خبره علوم رایانه طراحی شده است. اگر متدی را نشناسید، بیشتر وقت‌ها کافیست تنها حدس بزنید چگونه آن متد فراخوانی می‌شود.

شاید اکنون به اهمیت آن پی نبرید، به‌ویژه اگر مبتدی باشید، اما این یک ویژگی اصلی آن است. یعنی کمتر موجب آشفتگی افکارتان و به‌هم ریختگی و زیرورو کردن اسنادتان می‌شود و کمتر نیاز به رجوع به مغزتان حین کدنویسی می‌شود.

بهره‌وری توسعه‌دهنده

مطابق با نظر Mark Lutz (در ویرایش پنجم کتاب Learning Python انتشارات اورلی)، یک برنامه پایتون، معمولا یک پنجم تا یک سوم از اندازه محیط کد جاوا یا C++ را می‌گیرد. یعنی اجرای سریع‌تر برنامه. و سریع‌تر بودن عالی است. سریع‌تر بودن یعنی یک پاسخ سریع‌تر به بازار. کد کمتر نه تنها به معنای نوشتن کد کمتر است بلکه کد کمتری باید خوانده شود (و کدزن‌های حرفه‌ای، بسیار بیشتر از آنکه کدنویسی کنند کدخوانی می‌کنند)، کد کمتری تعمیر و نگهداری شود و پالایش شود.

جنبه مهم دیگر این است که پایتون بدون نیاز به گردآوری زمان‌بر و طولانی و به‌هم پیوستگی گام‌ها، اجرا می‌شود، بنابراین مجبور نیستید منتظر دیدن نتایج کار خود بمانید.

یک کتابخانه وسیع

پایتون دارای یک کتابخانه استاندارد عظیم باورنکردنی است. اگر جواب کارتان را هم ندهد، جامعه پایتون در سرتاسر جهان دارای کتابخانه‌های تنومند شخص‌سومی برای نیازهای شخصی است که می‌توان آزادانه (رایگان) در Python Package Index (PyPI) به آن دسترسی داشت. هنگامی که کدهای پایتون را می‌نویسید و متوجه می‌شوید که نیاز به یک ویژگی (فیچر) مشخصی دارید، در بیشتر موارد دست‌کم یک کتابخانه موجود است که آن ویژگی، از پیش برای شما پیاده‌سازی شده است.

کیفیت نرم‌افزار

پایتون به‌شدت روی خوانا بودن، انسجام، و کیفیت متمرکز است. یکنواختی زبان، امکان خوانا بودنش را افزایش می‌دهد و امروزه، بسیار مهم است که برنامه‌نویسی، یک تلاش جمعی باشد تا یک تلاش انفرادی، و به‌شکل گروهی انجام پذیرد. جنبه مهم دیگر پایتون، طبیعت ذاتی چند الگویی آن است. از آن می‌توان به‌عنوان یک زبان اسکریپتی استفاده کرد، اما می‌توان از سبک‌های شئ‌گرایی، دستوری، و برنامه‌نویسی فانکشنال نیز بهره‌برداری کرد. در کل، استعدادش بالاست.

یکپارچگی نرم‌افزار

جنبه مهم دیگر پایتون این است که می‌تواند با بسیاری دیگر از زبان‌ها توسعه داده شده و یکپارچه شود که به معنای این است که حتی وقتی یک شرکت درحال استفاده از زبان دیگری به‌عنوان ابزار اصلی کارش است، پایتون می‌تواند وارد شود و به‌عنوان یک عامل چسب، میان برنامه‌های کاربردی پیچیده‌ای که به هر روشی نیاز به صحبت با همدیگر دارند، عمل کند. این نوعی از یک مبحث پیشرفته است اما در جهان واقعی، این ویژگی بسیار مهم است.

علم داده

امروزه پایتون تقریبا معروف‌ترین (اگر نگوییم قطعا) زبان به‌کار رفته در فیلدهای علم داده، زبان ماشین، و هوش مصنوعی است. بنابراین دانش پایتون برای افرادی که می‌خواهند دستی در این فیلدها داشته باشند الزامی است.

رضایت‌مندی و لذت

کار با پایتون، جذاب و سرگرم‌کننده است. من می‌توانم ۸ ساعت کدنویسی کنم و خوشحال و راضی، شرکت را ترک کنم و این مغایر با تقلای برنامه‌نویس‌های دیگری است که مجبورند تحمل کنند، زیرا از زبان‌هایی استفاده می‌کنند که فاقد همین مقدار ساختارهای داده‌ای و دستورهای خوش طرح هستند. تردید نکنید که پایتون، کدنویسی را بسان سرگرمی می‌کند و این، انگیزه و کیفیت کارتان را بالا می‌برد.

اینها جنبه‌های اصلی است که چرا پایتون را به هرکسی پیشنهاد می‌کنم. البته، ویژگی‌های فنی و پیشرفته بسیاری هست که می‌توان درباره آنها صحبت کرد که جای آنها در این پیش‌گفتار نیست و فصل به فصل که جلو برویم خواهید دید.

موانع چیست؟

شاید تنها مانعی که کسی بتواند در پایتون بیابد که ربطی به سلیقه‌های شخصی ندارد، سرعت اجرایش است. نوعا، پایتون آهسته‌تر از برادران کامپایل شده‌اش است. پیاده‌سازی استاندارد محصولات پایتون، وقتی یک برنامه‌کاربردی را اجرا می‌کنید، یک نسخه‌ی کامپایل شده از سورس کد به‌نام بایت کد است (با پسوند pyc.)، که سپس توسط مفسر پایتون اجرا می‌شود. مزیت این رویه در قابل حمل بودن آن است که ما بهای این کاهش سرعت ناشی از این حقیقت که پایتون مانند زبان‌های دیگر، در سطح ماشین کامپایل نمی‌شود را می‌پردازیم.

هرچند، امروزه سرعت پایتون به‌ندرت یک مشکل تقلی می‌شود، و پهنه کاربردش توجهی به این ویژگی منفی آن ندارد. اتفاقی که می‌افتد این است که در زندگی واقعی، هزینه سخت‌افزار، یک مشکل دائمی نیست، و معمولا دست‌یابی به سرعت با کارهای موازی‌سازی، به‌اندازه کافی آسان است. افزون بر این، بسیاری از برنامه‌نویسان، بخش وسیعی از زمان انتظار را صرف تکمیل عملیات I/O می‌کنند؛ بنابراین سرعت اجرای خام، اغلب یک عامل دوم در بازدهی کلی به‌شمار می‌رود.

گفتنی است که توسعه‌دهندگان اصلی پایتون در سال‌های اخیر تلاش زیادی کرده‌اند تا عملیات مربوط به رایج‌ترین ساختارهای داده را با سرعت بیشتری اجرا کنند. این تلاش، در برخی موارد، بسیار موفق بوده است، و قدری این مشکل را رفع کرده است.

در مواردی که سرعت بسیار حیاتی است، یکی می‌تواند روی پیاده‌سازی‌های سریع‌تر پایتون همچون PyPy سوئیچ کند، که به‌طور متوسط، افزایش سرعت چهار–لا را با پیاده‌سازی تکنیک‌های کامپایل پیشرفته، ارائه می‌دهد. این همچنین امکان نوشتن اجزای بحرانی بازدهی کد را در سایر زبان‌ها همچون C یا C++ و تعامل آن با کد پایتون می‌دهد. کتابخانه‌هایی مانند pandas و NumPy (که استفاده مرسومی برای انجام علم داده در پایتون دارند) از چنین تکنیک‌هایی استفاده می‌کنند.

توجه: چند پیاده‌سازی متفاوت دیگر از زبان پایتون وجود دارد. در این کتاب، از پیاده‌سازی ارجاعی استفاده می‌کنیم که به CPython هم معروف است. فهرست سایر روش‌ها را در نشانی زیر ببینید:

https://www.python.org/download/alternatives/

اگر آن استدلال به‌اندازه کافی خوب نبود می‌توانید همواره درنظر بگیرید که پایتون برای هدایت بک‌اند سرویس‌هایی همچون Spotify و Instagram به‌کار رفته است که بازدهی، امر نگران کننده‌ای است. با این حال، پایتون کار خودش را به بهترین شکل ممکن انجام داده و می‌دهد.

امروزه چه افرادی از پایتون استفاده می‌کنند؟

هنوز قانع نشده‌اید؟ اجازه دهید نگاهی به شرکت‌هایی که امروزه از پایتون استفاده می‌کنند بیاندازیم: گوگل، یوتیوب، دراپ‌باکس، یاهو!، Zope Corporation، Industrial Light & Magic، Walt Disny Feature Animation، Blender 3D، Pixar، NASA، NSA، Red Hat، Nokia، IBM، Netflix، Yelp، Intel، Cisco، HP، Qualcomm و JPMorgan Chase تنها چند مورد از آنهاست.

حتی بازی‌هایی همچون Battlefield 2، Civilization IV و QuArK با استفاده از پایتون پیاده‌سازی شده‌اند.

پایتون در مفاهیم پیچیده‌ای همچون برنامه‌نویسی سیستمی، برنامه‌نویسی وب و API، برنامه‌های کاربردی GUI، بازی‌سازی و روباتیک، نمونه‌سازی سرعت، تعامل و یکپارچه‌سازی سیستم، علم داده، برنامه‌های کاربردی دیتابیس، و موارد دیگر کاربرد دارد. چندین دانشگاه معتبر نیز پایتون را به عنوان زبان اصلی در رشته علوم کامپیوتر تدریس می‌کنند.

تنظیم محیط

در سیستم من (MacBook Pro)، این آخرین نسخه پایتونی است که دارم:

```
>>> import sys
>>> print(sys.version)
3.12.2 (main, Feb 14 2024, 14:16:36) [Clang 15.0.0 (clang-1500.1.0.2.5)]
```

همان‌گونه که می‌بینید یک نسخه 3.12.2 است که در دوم اکتبر ۲۰۲۳ انتشار یافته است. متن آغازین بالا کمی کد پایتون است که در یک کنسول تایپ شده است. در مورد این کمی جلوتر صحبت خواهیم کرد.

همه مثال‌های این کتاب با استفاده از پایتون ۳.۱۲ اجرا خواهد شد. اگر دوست دارید مثال‌ها را دنبال کنید و سورس‌کد را دانلود کنید، مطمئن شوید که از همین نسخه استفاده می‌کنید.

نصب پایتون

واقعا هرگز به داشتن یک بخش setup در یک کتاب متقاعد نشده‌ام، باوجودی که مجبورید حتما فرایند نصب را انجام دهید. بیشتر وقت‌ها، میان زمانی که نویسنده، دستورها را می‌نویسد و زمانی که شما آنها را پیاده‌سازی می‌کنید ماه‌ها می‌گذرد. و این بستگی به شانس شما دارد. یک نسخه تغییر می‌کند و شاید مواردی از آن با روشی که در کتاب آمده، کار نکند. خوشبختانه ما وب را داریم، پس برای کمک به نصب و اجرای پایتون، تنها به اهداف و نکات مهم فرایند نصب می‌پردازم.

می‌دانم که بیشتر خوانندگان ممکن است ترجیح دهند کتاب به‌شکل راهنمای گام به گام باشد. شک دارم این روش، کارشان را آسان‌تر کند، پس به‌شدت باور دارم که اگر می‌خواهید کار با پایتون را آغاز کنید باید نخستین تلاشتان در جهت آشنایی با زیست‌بوم آن باشد. این بسیار مهم است و موجب افزایش اعتماد به نفس شما در فصل‌های پیش رو خواهد شد. هر کجا هم که گیر کردید، دوست خوبمان گوگل را فراموش نکنید.

تنظیم مفسر پایتون

فرایند نصب پایتون روی رایانه شما بستگی به سیستم‌عامل‌تان دارد. پیش از همه، پایتون به احتمال زیاد، از پیش در تقریبا هر توزیع لینوکسی به‌شکل پایه‌ای نصب و به‌شکل کامل، با آن یکپارچه شده است. اگر دارای سیستم‌عامل مک هستید، احتمال دارد پایتون۳ از پیش روی آن باشد، اما اگر از ویندوز استفاده می‌کنید به احتمال زیاد، نیاز به نصب آن دارید.

توجه: صرف‌نظر از اینکه پایتون از پیش روی سیستم‌تان نصب است، لازم است مطمئن شوید که نسخه ۳.۱۲ نصب شده باشد. برای تشخیص این مسئله، عبارت python –version یا python3 –version را در پرامپت فرمان تایپ کنید.

دریافت پایتون و کتابخانه‌های مورد نیاز و اجرای آن نیاز به کمی کار دستی دارد. به‌نظر می‌رسد لینوکس و مک، سیستم‌عامل‌های کاربرپسندتری برای برنامه‌نویسان پایتون باشند؛ اما ویندوز، بیشترین تلاش را می‌طلبد.

کار را از وب‌سایت رسمی پایتون به نشانی https://www.python.org آغاز کنید که میزبان مستندات رسمی پایتون و بسیاری منابع مفید دیگر است. زمانی را به بررسی آن تخصیص دهید.

منابع نصب مفید

وب‌سایت پایتون میزبان اطلاعات مفیدی نسبت به نصب پایتون روی سیستم‌عامل‌های مختلف است. برای هر سیستم‌عاملی به صفحه مربوط مراجعه کنید.

ویندوز و مک:

- https://docs.python.org/3/using/windows.html
- https://docs.python.org/3/using/mac.html

برای لینوکس، به لینک‌های زیر رجوع کنید:

- https://docs.python.org/3/using/unix.html
- https://ubuntuhandbook.org/index.php/2023/05/install-python-3-12-ubuntu/

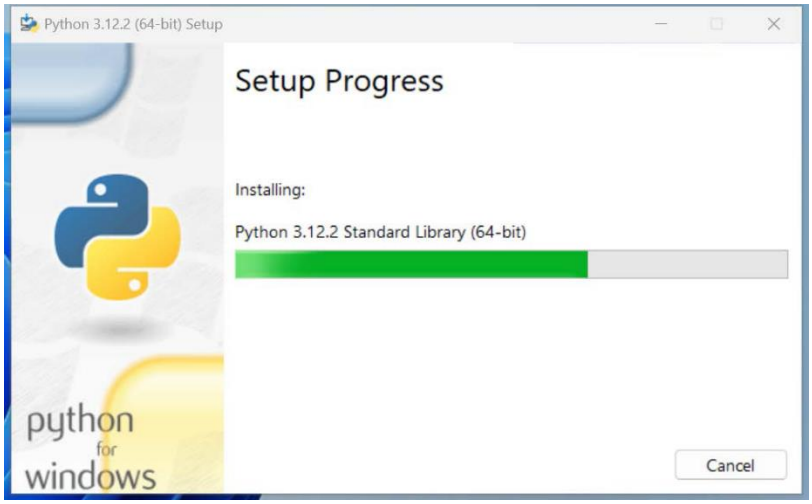
نصب پایتون روی ویندوز

برای‌مثال، این رویه‌ای برای نصب پایتون روی ویندوز است. نصب‌کننده متناسب با CPU کامپیوتر خود را از سرتیتر https://www.python.org/downloads/ دانلود کنید.

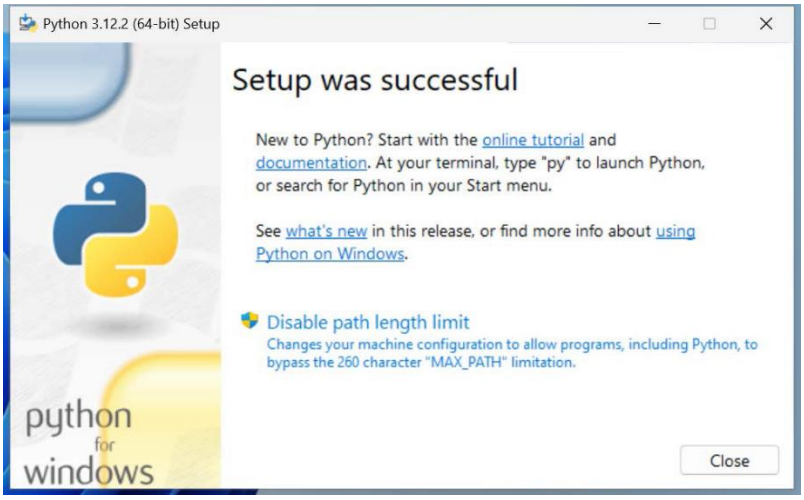
با این کار، برای آغاز فرایند نصب، می‌توانید روی آن دابل کلیک کنید.

پیشنهاد می‌کنیم نصب پیش‌فرض را انتخاب کنید، و گزینه Add python.exe to PATH را تیک نزنید تا از برخورد با سایر نسخه‌های پایتونی که ممکن است توسط سایر کاربران روی ماشین نصب شده باشد جلوگیری شود. برای درک بهتر مجموعه راهنماها، به لینک مندرج در پاراگراف قبلی رجوع کنید.

به محض کلیک روی Install Now، فرایند نصب آغاز می‌شود:



پس از پایان فرایند نصب، به صفحه پایانی هدایت خواهید شد:



روی Close کلیک کنید تا مراحل نصب پایان یابد.

اکنون که پایتون را روی سیستم خود نصب کردید، یک پرامپت فرمان باز کنید و با تایپ py، پایتون را در پوسته تعاملی (Python interactive shell) باز کنید. این فرمان، آخرین نسخه از پایتون نصب شده روی ماشین را انتخاب خواهد کرد. هنگام نگارش این کتاب، ۳.۱۲ آخرین نسخه پایتون بود. اگر نسخه جدیدتری را نصب دارید می‌توانید با فرمان py -3.12 آنرا تعیین کنید.

برای باز کردن پرامپت فرمان در ویندوز، به منوی Start ویندوز بروید و در کادر سرچ، عبارت cmd را تایپ کنید تا ترمینال بالا آغاز به کار کند. به‌صورت جایگزین، می‌توانید از Powershell نیز استفاده کنید.

نصب پایتون روی macOS

فرایند نصب در macOS، همانند آن در ویندوز است. به‌محض دانلود نصب‌کننده مناسب برای ماشین، گام‌های نصب را کامل کنید و سپس با رفتن به Application | Utilities| Terminal، یک ترمینال را آغاز کنید. یا اینکه از طریق Homebrew نصب کنید.

با ورود به پنجره ترمینال، می‌توانید python را تایپ کنید. اگر که نسخه اشتباهی بالا بیاید می‌توانید آنرا تغییر دهید و نسخه را با python3 یا python3.12 مشخص کنید.

نصب پایتون روی لینوکس

فرایند نصب پایتون روی لینوکس معمولا کمی پیچیده‌تر از ویندوز و مک است. بهترین کار این است که اگر روی یک ماشین لینوکسی هستید اقدام به جست‌وجوی آنلاین جدیدترین مجموعه گام‌ها برای توزیع کنید. احتمالا این

از توزیعی به توزیع دیگر کاملا متفاوت باشد و ذکر مثالی که همه توزیع‌ها را پوشش دهد امکان‌پذیر نیست. برای راهنمایی به لینک Useful installation resources بروید.

کنسول پایتون

از واژه کنسول به‌شکل مبادله‌ای برای اشاره به کنسول لینوکس، پرامپت خط فرمان ویندوز یا Powershell، و ترمینال مکینتاش استفاده خواهم کرد. همچنین با فرمت پیش‌فرض لینوکس به پرامپت کامند–لاین (یا همان خط–فرمان سریع) اشاره خواهم کرد، مانند این:

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

اگر با این دستور آشنا نیستید لطفا کمی وقت برای یادگیری مبانی کارکرد کنسول بگذارید. به اختصار، پس از نماد $ معمولا دستوری می‌آید که باید تایپ کنید. به بزرگی و کوچکی واژگان و فاصله‌ها دقت کنید که بسیار مهم است.

پس از باز شدن کنسول، python را در پرامپت تایپ کنید و مطمئن شوید پوسته تعاملی پایتون نمایش می‌یابد. برای خروج، ( ) exit را تایپ کنید. فراموش نکنید که اگر سیستم‌عامل‌تان دارای سایر نسخه‌های پایتونی که از پیش نصب شده است باشد، شاید مجبور شوید نسخه python3 یا pothon3.12 را مشخص کنید.

توجه: معمولا برای اشاره به پوسته تعاملی پایتون، از واژه ساده شده‌ی **کنسول پایتون** استفاده می‌کنیم.

پس از اجرای پایتون، باید عبارات زیر را ببینید (برخی جزئیات برپایه نسخه و OS تغییر می‌کند):

```
$ python
Python 3.12.2 (main, Feb 14 2024, 14:16:36)
[Clang 15.0.0 (clang-1500.1.0.2.5)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

اینک که پایتون نصب شده و می‌توانید آنرا اجرا کنید زمان آن است که مطمئن شوید ابزار دیگری دارید که با آن مثال‌های این کتاب را دنبال کنید: یک محیط مجازی یا virtual environment.

درباره محیط‌های مجازی

حین کار با پایتون، استفاده از محیط‌های مجازی، بسیار مرسوم است. اجازه دهید کمی توضیح دهیم که اینها چیست و چرا به اینها نیاز داریم. با یک مثال ساده، توضیح می‌دهیم.

پایتون را روی سیستم نصب می‌کنید و کار با یک وب‌سایت برای Client X را آغاز می‌کنید. یک پوشه پروژه می‌سازید و شروع به کدنویسی می‌کنید. در کنار آن، برخی کتابخانه‌ها را نیز نصب می‌کنید؛ مانند فریمورک

Django که در فصل ۱۴ به آن خواهیم پرداخت. فرض کنید نسخه Django 4.2 را برای Project X نصب می‌کنید.

اکنون وب‌سایت شما آنقدر خوب است که مشتری دیگری به‌نام Y می‌گیرید. او می‌خواهد وب‌سایت دیگری بسازید، پس Project Y را آغاز می‌کنید و به همان روش، دوباره نیاز به نصب جنگو دارید. تنها مشکلی که وجود دارد این است که نسخه جنگو اکنون به 5 ارتقاء یافته است و نمی‌توانید آن را روی سیستم خود نصب کنید زیرا جایگزین نسخه پیشین آن که برای Project X نصب کرده بودید خواهد شد. نمی‌خواهید ریسک معرفی مشکلات ناسازگاری را بکنید، بنابراین دو انتخاب دارید: یا درگیر نسخه‌ای شوید که هم‌اکنون روی سیستم دارید، یا آنرا به‌روز کنید و مطمئن شوید نخستین پروژه، باز به‌شکل کامل و صحیح با نسخه جدید کار می‌کند.

اگر صادق باشیم، هیچ کدام از این انتخاب‌ها چندان خوشایند نیست. قطعا نیست. اینجاست که راه‌حل مجازی‌سازی به میان می‌آید: محیط‌های مجازی!

محیط‌های مجازی، محیط‌های ایزوله شده پایتون هستند که هر کدام، پوشه‌ای است حاوی همه اجرا شدنی‌های لازم برای استفاده از پکیج‌هایی که یک پروژه پایتون لازم دارد (فعلا پکیج‌ها را مانند کتابخانه درنظر بگیرید).

پس برای ایجاد یک محیط مجازی برای Project X، همه وابستگی‌ها را نصب کنید و سپس یک محیط مجازی برای پروژه Y بسازید و همه وابستگی‌هایش را بدون کوچکترین نگرانی نصب کنید، زیرا هر کتابخانه‌ای که بعدا نصب می‌کنید، در انتهای مرزهای محیط مجازی مناسب می‌نشیند. در این مثال، پروژه X کتابخانه Django 4.2 را نگه می‌دارد، درحالی که پروژه Y، کتابخانه Django 5.0 را نگه می‌دارد.

توجه: بسیار مهم است که هرگز کتابخانه‌ها را مستقیما در سطح سیستم نصب نکنید. مثلا لینوکس برای وظایف و عملیات گوناگونی به پایتون تکیه دارد و اگر با سیستم نصب پایتون بخواهید آزارش دهید، ریسک به‌هم ریختگی تعامل کل سیستم را بالا خواهید برد (حدس بزنید برای این فرد چه اتفاقی می‌افتد...). پس این را به‌عنوان یک قانون مدنظر داشته باشید که پیش از رفتن به تخت‌خواب، مسواک بزنید: *همیشه، همیشه هنگام آغاز یک پروژه جدید، یک محیط مجازی بسازید.*

برای ساخت یک محیط مجازی روی سیستم‌تان، از چند روش مختلف می‌توانید استفاده کنید. به‌طوری که پایتون ۳.۵، روش ساخت یک محیط مجازی که از ماژول venv استفاده می‌کند را پیشنهاد داده است. در مستند رسمی پایتون به نشانی (https://docs.python.org/3/library/venv.html) می‌توانید آنرا ببینید.

نکته: مثلا اگر از یک توزیع لینوکس دبین–محور استفاده می‌کنید، لازم است ماژول venv را پیش از اینکه بتوانید از آن استفاده کنید با فرمان زیر نصب کنید:

```
$ sudo apt-get install python3.12-venv
```
