به نام ایزد یکتا

مقدمه‌ای عملی بر پایتون

جهت ارئه برای درس مکاترونیک2

استاد: جناب آقای دکتر بادنوا

فاضل محمد علی پور

ارشد مکاترونیک

اسفند 1402

فهرست مطالب:

تاریخچه کوتاه:

زبان برنامه نویسی پایتون، در اواخر دهه ۱۹۸۰، توسط Guido Van Rossum محقق هلندی ابداع شد؛ Van Rossum در سال ۱۹۸۹ پیاده سازی پایتون را آغاز کرد و آن را جایگزین زبان ABC معرفی کرد.

نسخه اولیه و نخستین در سال 1991 تحت ورژن عددی 0.9 منتشر شد.

- نسخه 1 در ژانویه 1994 منتشر شد.
- نسخه 2.1 در سال 1995 منتشر شد.
- و ...

پایتون چیست؟:

پایتون (Python Programming Language) یک زبان برنامه‌نویسی دارای نحو (Syntax) ساده و سطح بالا (High-Level Programming Language) است. در عین حال، پایتون یک زبان همه منظوره (General Purpose)، «چندپلتفرمی» (Multi-Platform) و چند پارادایمی (Multi-Paradigm) است که از نوع داده پویا (Dynamic Data Type) پشتیبانی می‌کند.

زبانی ساده و قابل درک که به اندازه دیگر زبان‌های برنامه‌نویسی رقیب خود، قدرتمند باشد. زبانی متن-باز که همه علاقه‌مندان بتوانند در توسعه آن مشارکت کنند. زبانی که کدهای آن به اندازه متن انگلیسی قابل درک و خوانا باشند. زبانی که برای وظایف روزانه مناسب باشد و امکان توسعه برنامه را به سرعت و در زمان کوتاه فراهم کند.

به طور کل دو مدل(ورژن کلی) از پایتون موجود است ورژن 2 و ورژن 3، که عملا میتوان گفت دسته بندی؛ اما در حال حاضر صحبتی از دسته پایتون2 به میان نیست؛ تفاوت‌های اصلی بین پایتون ۲ و پایتون ۳ عبارتند از:

۱ .مدیریت رشته‌ها:(Strings Handling)

در پایتون ۳، رشته‌ها به صورت Unicode است که بهبود قابلیت اطمینان و کاربرد رشته‌ها را ارتقا داده است. در حالی که در پایتون ۲، استفاده از رشته‌های با بایت‌ها (Byte strings) و Unicode می‌تواند گیج‌کننده باشد.

۲ .تفاوت در عملگر تقسیم:(/)

در پایتون ۳، عملگر تقسیم (/) نتیجه‌ی نهایی را به عدد حقیقی (float) تبدیل می‌کند. اما در پایتون ۲، تقسیم (/) نتیجه‌ی نهایی را به نوع داده‌ی دیگری تبدیل می‌کند (صحیح یا حقیقی، بسته به نوع داده‌ای که در دسترس است).

۳ .تفاوت در دستور چاپ:

در پایتون ۳، دستور چاپ (print) به عنوان یک تابع عمل می‌کند و برای استفاده از آن باید پرانتز استفاده شود مانند ("print("Hello، در حالی که در پایتون ۲ این کار اختیاری بوده و پرانتز لازم نیست.

۴ .دیگر تغییرات:

پایتون ۳ تغییرات دیگری نیز در مورد نحوه‌ی مدیریت استثناءها(Exceptions) ، تعداد و نوع توابع داخلی، نحوه‌ی مدیریت توابع جدید و موارد دیگر داشته است.

***به طور کلی، پایتون ۳ بهبودهای زیادی از نظر کارایی، قابلیت اطمینان و کدپایه‌ی بهتری نسبت به پایتون ۲ دارد. به همین دلیل، در حال حاضر بسیاری از مجتمع‌ها، پروژه‌ها و برنامه‌نویسان از پایتون ۳ استفاده می‌کنند. همچنین، پایتون ۲ دیگر توسعه‌ی فعالی ندارد و پشتیبانی آن نیز در سال ۲۰۲۰ متوقف شده است، بنابراین استفاده از نسخه‌ی پایتون ۳ برای پروژه‌های جدید توصیه می‌شود .**[مسیر یادگیری پایتون](#).

تفاوت زبان‌های سطح بالا و سطح پایین و تفاوت زبان‌های کامپایل شونده و تفسیری1:

سطح بالا: برای درک تفاوت زبان برنامه نویسی سطح بالا و پایین بهتره که با ویژگی هر کدام به صورت مجزا آشنا بشیم. جالبه بدونید که زبان‌های سطح بالا از ترکیب عبارت‌های انگلیسی، نمادهای ریاضی و... برای ساختن و نوشتن دستورالعمل‌ها استفاده می‌کنن؛ به همین دلیل درک اون‌ها برای انسان راحت‌تره. در عین حال به دلیل استفاده از این عبارات باید از یک کامپایلر یا مفسر برای ترجمه زبان سطح بالا به زبان قابل فهم برای کامپیوتر استفاده کنیم؛ با توجه به این مسئله یکی از تفاوت زبان برنامه نویسی سطح بالا و پایین سرعت نه‌چندان بالای این زبان‌ها در مقایسه با زبان‌های سطح پایینه.(با توجه به پیشرفت روز افزاون پروسسورها و چیپ‌ها و تفاوت‌هایی که در نوع پردازش داده شاهد هستیم عملا اگر کار در مقیاس خیلی بزرگ باشد معمولا این موضوع حائز اهمیت می‌شود).

مجموع مشخصات این زبان‌ها رو می‌تونیم به صورت زیر خلاصه کنیم:

- امکان اجرا در پلتفرم‌های مختلف
- درک آسان
- استفاده‌ی گسترده
- دیباگ ساده
- تفسیر و کامپایل ساده در مقایسه با زبان سطح پایین
- برای ترجمه این زبان به کدهای خوانا توسط ماشین به یک کامپایلر یا مفسر نیاز داریم.
- اشغال بیشتر حافظه در مقایسه با زبان‌های سطح پایین
- محبوبیت بالا در سراسر جهان

جاوا، پایتون، Cو C++ نمونه‌های از زبان‌های سطح بالا هستند.

سطح پایین: از جمله تفاوت زبان برنامه نویسی سطح بالا و پایین می‌تونیم به ارتباط مستقیم زبان‌های سطح پایین با پردازنده‌ی سیستم و استفاده از کدهای باینری یا ماشین به جای کدهای انگلیسی اشاره کنیم. به همین دلیل درک چنین کدهایی برای انسان سخت‌تر و برای ماشین راحت‌تره؛ چرا که پردازنده به صورت مستقیم کدها رو پردازش می‌کنه. یکی از نمونه‌های زبان برنامه نویسی سطح پایین، زبان ماشینه که با ارسال کدهای ۰ و ۱ تلاش می‌کنه تا دستورالعمل‌ها رو مستقیماً به پردازنده منتقل کنه؛ اما به دلیل امکان بالای خطا در این زبان، مهندسان زبان دیگری به نام اسمبلی رو توسعه دادند که درکش برای انسان راحت‌تر از زبان ماشینه.

مجموع مشخصات زبان‌های سطح پایین رو می‌تونیم به شکل زیر بیان کنیم:

- دیباگ دشوار
- استفاده‌ی محدود
- عدم امکان اجرا روی پلتفرم‌های مختلف
- درک آسان توسط ماشین
- درک دشوار برای انسان
- برای ترجمه‌ی دستورالعمل‌ها به یک اسمبلر نیاز داریم.

تفاوت زبان‌های سطح بالا و سطح پایین و تفاوت زبان‌های کامپایل شونده و تفسیری2:

1. زبان‌های کامپایل شونده:

در زبان‌های کامپایل شونده، کد منبع برنامه توسط یک کامپایلر به زبان ماشین ترجمه می‌شود. این ترجمه یکباره صورت می‌گیرد و یک فایل اجرایی (باینری) ایجاد می‌شود که بعداً می‌تواند بدون نیاز به مجدداً ترجمه، در هر زمانی اجرا شود. مثال‌هایی از زبان‌های کامپایل شونده شامل C ، C++، و Java می‌شوند.

۲ .زبان‌های تفسیری:

در زبان‌های تفسیری، کد منبع به صورت مستقیم توسط یک مفسر یا مترجم به زبان ماشین ترجمه و اجرا می‌شود. این ترجمه در هنگام اجرا صورت می‌گیرد و نیازی به ایجاد یک فایل اجرایی جداگانه نیست. مثال‌هایی از زبان‌های تفسیری شامل Python، Ruby، و JavaScript می‌شوند.

***تفاوت اصلی بین این دو نوع زبان برنامه‌نویسی این است که در زبان‌های کامپایل شونده، کد برنامه ابتدا به زبان ماشین ترجمه می‌شود و سپس اجرا می‌شود، در حالی که در زبان‌های تفسیری، کد به صورت تفسیری و در هنگام اجرا ترجمه و اجرا می‌شود. همچنین، زبان‌های تفسیری معمولاً انعطاف‌پذیرتر و قابل خواندن‌تر هستند، در حالی که زبان‌های کامپایل شونده بهینه‌تر و سریع‌تر می‌توانند اجرا شوند.**

پارامتر	زبان سطح بالا	زبان سطح پایین
اصلی	اصلی‌ترین تفاوت برنامه نویسی سطح بالا و سطح پایین، کاربرد پسند بودن این زبان‌ها است که مدیریت، درک و دیباگ اون‌ها رو ساده‌تر کرده و باعث کاربرد گسترده‌تر اون‌ها شده.	درک زبان‌های سطح پایین یا سازگار با ماشین برای انسان دشواره؛ اما تفسیر اون‌ها به راحتی توسط ماشین انجام میشه.
سهولت اجرا	ساده	دشوار
فرآیند ترجمه	به یک کامپایلر یا مفسر برای ترجمه به کد ماشین نیاز دارند.	برای ترجمه‌ی مستقیم دستورالعمل‌های زبان ماشین به یک اسمبلر نیاز داریم.
کارآمدی حافظه	بسیار پایین: اشغال بیشتر حافظه در مقایسه با زبان‌های سطح پایین	بسیار بالا: اشغال حجم کوچکی از حافظه در مقایسه با زبان‌های سطح بالا
قابلیت انتقال	امکان جابه‌جایی بین دستگاه‌های مختلف	عدم امکان جابه‌جایی بین دستگاه‌ها
قابل فهم بودن	درک و یادگیری آسان توسط برنامه‌نویس‌های مختلف	درک و یادگیری دشوار
وابستگی به ماشین	این زبان‌ها به ماشین وابسته نیستند.	این زبان‌ها به ماشین وابسته هستند و درک آن‌ها توسط کاربران عادی دشواره
دیباگ کردن	آسان	دشوار
کاربرد	بسیار رایج هستند و به صورت گسترده استفاده میشن.	در حال حاضر چندان رایج نیستند.
سرعت اجرا	از جمله تفاوت زبان برنامه نویسی سطح بالا و پایین سرعت اجرای پایین این زبان به دلیل ترجمه کدها است.	سرعت اجرای بالا
نیاز به سخت‌افزار	برنامه نویسی با این زبان‌ها به دانش سخت‌افزاری نیاز نداره.	پیش نیاز نوشتن برنامه با این زبان‌ها، آشنایی با سخت‌افزارها است و این یکی از مهم‌ترین تفاوت برنامه نویسی سطح بالا و سطح پایین محسوب میشه.
سهولت تغییر و اصلاح	دشوار: هر عبارت ممکنه که دستورالعمل‌های مختلفی رو اجرا کنه.	ساده: میشه عبارت رو به صورت مستقیم به دستورالعمل تبدیل کرد.
مثال	COBOL، BASIC، Perl، پایتون، جاوا و...	زبان ماشین و اسمبلی

کاربردها و دلایل کلی برای یادگیری پایتون(دلایل یادگیری):

سینتکس واضح و خوانا:

Pythonدارای سینتکس یا نحوی فوق‌العاده تمیز و ساده است که آن را بسیار خوانا و قابل‌درک می‌کند. این سادگی، هم مبتدیان و هم توسعه‌دهندگان با تجربه را قادر می‌سازد تا به طور یکپارچه در پروژه‌های پیچیده همکاری کنند و کار گروهی مؤثر را ارتقا دهند.

مقیاس‌پذیری:

مقیاس‌پذیری زبان برنامه‌نویسی پایتون یکی از قابلیت‌های مهم این زبان است که آن را برای مشاغل در هراندازه مناسب می‌کند. شرکت‌های مشهوری مانند گوگل، اسپاتیفای، نتفلیکس و اینستاگرام از قابلیت‌های پایتون برای ساخت برنامه‌هایی استفاده کرده‌اند که بدون زحمت حجم عظیمی از ترافیک و تعاملات کاربر را مدیریت می‌کنند.

تطبیق‌پذیری:

یکی از ویژگی‌های برجسته پایتون تطبیق‌پذیری باورنکردنی آن است. برخلاف برخی از زبان‌های برنامه‌نویسی که محدود به دامنه‌های خاص هستند، پایتون کاربردهای عملی را در دامنه‌های مختلف پیدا می‌کند. این زبان در توسعه برنامه‌های وب و تلفن همراه، برنامه‌های بازی، راه‌حل‌های درجه یک سازمانی، پلتفرم‌های تجارت الکترونیک و زمینه‌های پیشرفته مانند یادگیری ماشینی و هوش مصنوعی برتری دارد. **منبع پیشنهادی برای نصب :**آموزش نصب پایتون در انواع سیستم عامل

اکوسیستم غنی:

محبوبیت پایتون اکوسیستم وسیع و پررونقی از کتابخانه‌ها و چارچوب‌ها را پرورش داده است. این ابزارهای از پیش‌ساخته شده به طور قابل‌توجهی فرآیندهای توسعه را تسریع می‌کنند و به توسعه‌دهندگان اجازه می‌دهند تا از راه‌حل‌های موجود استفاده کنند و در زمان و تلاش صرفه‌جویی کنند.

انجمن و پشتیبانی:

پایتون دارای یک جامعه پر جنب‌وجوش و فعال از توسعه‌دهندگان است که به بهبود مستمر آن کمک می‌کنند. این امر پشتیبانی مداوم، به‌روزرسانی‌های مکرر و منابع فراوانی را در دسترس توسعه‌دهندگان در هر سطح مهارت قرار می‌دهد.

استقلال پلتفرم:

رویکرد Python استقلال از پلتفرم پایتون به برنامه‌ها امکان می‌دهد بدون تغییر بر روی پلتفرم‌های مختلف به طور یکپارچه اجرا شوند. این سازگاری بین پلتفرم، استقرار را ساده می‌کند و نیاز به توسعه پلتفرم خاص را کاهش می‌دهد.

نمونه‌سازی سریع: نحو مختصر و رسا پایتون به توسعه‌دهندگان اجازه می‌دهد تا ایده‌ها و مفاهیم را به سرعت نمونه‌سازی کنند. توانایی آن برای تبدیل وظایف پیچیده به کد ظریف، چرخه‌های توسعه سریع را ارتقا می‌دهد و زمان ورود به بازار را تسریع می‌کند.

یادگیری آسان: یادگیری نسبتاً آسان پایتون آن را به یک انتخاب عالی برای مبتدیان تبدیل می‌کند. خوانایی و طراحی شهودی آن، فرآیند نصب روان را برای توسعه‌دهندگان مشتاق تسهیل کرده و آن‌ها را قادر می‌سازد تا شروع به ساخت برنامه‌های کاربردی مفید در اوایل سفر یادگیری خود کنند.

پشتیبانی از توسعه تست محور: سادگی دستور کدنویسی پایتون، آن را برای اتخاذ شیوه‌های توسعه آزمایش محور مساعد می‌کند. این امر توسعه‌دهندگان را تشویق خواهد کرد تا قبل از اجرای ویژگی‌ها، آزمایش‌هایی روی آن انجام دهند که منجر به برنامه‌های قوی‌تر و قابل‌اعتمادتر می‌شود.

نحو ساده: نحو و در واقع دستورات ساده موجود در زبان برنامه‌نویسی پایتون این زبان را به گزینه مناسبی برای فراگیری برنامه‌نویسی به طور خاص برای افرادی که تازه وارد دنیای برنامه‌نویسی شده‌اند مبدل کرده است. همچنین، افرادی که از دیگر رشته‌های علمی نیاز به یادگیری یک زبان برنامه‌نویسی برای کاربردهای مربوط به رشته خود دارند نیز می‌توانند به راحتی پایتون را بیاموزند. مطالعه مطلب «برنامه نویسی پایتون برای مبتدیان – به زبان ساده» در این راستا به افراد مبتدی پیشنهاد می‌شود.

همه‌منظوره بودن: پایتون یک زبان برنامه‌نویسی همه‌منظوره است. این یعنی از پایتون می‌توان برای کاربردهای گوناگون و در حوزه‌های مختلف استفاده کرد. از زبان پایتون می‌توان برای توسعه وب، توسعه اپلیکیشن‌های دسکتاپ و موبایل، برنامه‌نویسی محاسباتی، هوش مصنوعی (و یادگیری ماشین)، علم داده و بسیاری از دیگر زمینه‌ها استفاده کرد. شایان توجه است که افراد در رشته‌های مختلف علمی، می‌توانند از پایتون برای پیاده‌سازی‌های مربوط به زمینه فعالیت خود استفاده کنند. قابلیت مهمی که در بحث همه‌منظوره بودن پایتون نباید از آن چشم پوشید این است که همه‌منظوره بودن پایتون موجب می‌شود تا فرد یک‌بار یک زبان برنامه‌نویسی را بیاموزد و از آن در حوزه‌های مختلف کاری استفاده کند. همچنین، پایتون تا حد خوب و قابل توجهی فرد را از اینکه طی یک پروژه برنامه‌نویسی از زبان‌های مختلف برای منظورهای مختلف استفاده کند، بی‌نیاز می‌سازد.

چندسکویی: پایتون یک زبان برنامه‌نویسی چندپلتفرمی یا چندسکویی است؛ این یعنی می‌توان از پایتون برای برنامه‌نویسی در پلتفرم‌های گوناگون دسکتاپ و موبایل استفاده کرد. در واقع، می‌توان با پایتون برای سیستم عامل‌های گنو/لینوکس، ویندوز، مک و یونیکس و برنامه‌نویسی پلتفرم‌های موبایل مانند اندروید و iOS استفاده کرد. البته، در حال حاضر پایتون خیلی گزینه مناسبی برای برنامه‌نویسی موبایل نیست و پیشگام‌های این حوزه کاتلین و جاوا هستند.

چندپارادایمی: پایتون از پارادایم‌های برنامه‌نویسی گوناگون پشتیبانی می‌کند. از جمله این موارد می‌توان به برنامه‌نویسی خطی، تابعی و شی‌گرا اشاره کرد. این امر پایتون را به گزینه مناسبی برای پروژه‌های گوناگون مبدل می‌کند. امروزه،پارادایم برنامه‌نویسی شی‌گرا محبوب‌ترین و پراستفاده‌ترین پارادایم برنامه‌نویسی است که از آن در پروژه‌های گوناگونی استفاده می‌شود. پشتیبانی از این پارادیم برنامه‌نویسی بسیار پر کاربرد، یک مزیت کلیدی برای هر زبان برنامه‌نویسی محسوب می‌شود. حال آنکه پایتون از پارادایم‌های محبوب دیگری مانند برنامه‌نویسی خطی و تابعی پشتیبای می‌کند و همین، پایتون را به زبانی قدرتمند و مناسب برای پروژه‌های ریز و درشت در حوزه‌های گوناگون مبدل می‌کند.

کتابخانه‌های شخص ثالث متعدد و متنوع: پایتون از کتابخانه‌های متعددی بهره می‌برد که در حوزه‌های گوناگون از وب گرفته تا هوش مصنوعی و یادگیری ماشین و همچنین، علم داده، قابل استفاده هستند. وجود کتابخانه‌های قدرتمندی مانند نام‌پای (NumPy)، سای‌پای (SciPy)، پانداس (Pandas)، کرس (Keras)، سایکیت‌لرن (Scikit-Learn)، بوکه (Bokeh)، تنسورفلو (Tensorflow) و دیگر موارد، پایتون را به زبان یکه‌تاز حوزه هوش مصنوعی و علم داده مبدل کرده است.

و ...

کاربردها و دلایل کلی برای یادگیری پایتون-(کاربردها)2:

۱ .**هوش مصنوعی و یادگیری ماشین:** پیاده‌سازی یک الگوریتم تشخیص تصاویر در پایتون با استفاده از کتابخانه TensorFlow.توضیح :در این مثال، ممکن است بخواهید یک مسئله پردازش تصویر را حل کنید. با استفاده از پایتون و کتابخانهTensorFlow ، می‌توانید یک مدل یادگیری عمیق (Deep Learning) برای تشخیص الگوها و اشیاء مختلف در تصاویر ایجاد کنید.

۲ .**علم داده:** آنالیز داده‌های مالی و پیش‌بینی روند بازار سهام با استفاده از Pandas و Matplotlib .توضیح :در اینجا، با استفاده از پایتون و کتابخانه‌های موردنیاز مانند Pandas برای آنالیز داده‌ها و Matplotlib برای رسم نمودارها، می‌توانید داده‌های مالی را بخوانید، تحلیل کنید و الگوریتم‌های پیش‌بینی را پیاده‌سازی کنید.

۳ .**طراحی و توسعه اپلیکیشن‌های دسکتاپ و موبایل:** توسعه یک برنامه دسکتاپ مدیریت پروژه با استفاده از فریم‌ورک PyQt.توضیح :در اینجا، شما می‌توانید با استفاده از پایتون و فریم‌ورک‌های موردنیاز برای توسعه برنامه‌های دسکتاپ مدیریت پروژه، اپلیکیشن‌های موبایل و حتی اپلیکیشن‌های وب مدیریت کنید.

۴ .**توسعه بازی‌های کامپیوتری:** توسعه یک بازی ماجراجویی ساده با استفاده از کتابخانه Pygame .توضیح :در این مثال، می‌توانید از پایتون و کتابخانه Pygame برای ساخت بازی‌های کامپیوتری استفاده کنید. این کتابخانه به شما امکان می‌دهد گرافیک‌های دوبعدی را به راحتی رسم کرده و بازی‌های مختلف را ایجاد کنید.

۵ .**امنیت سایبری و شبکه:** پیاده‌سازی یک ابزار بررسی امنیت شبکه با استفاده از کتابخانهScapy .توضیح :در این مثال، می‌توانید از پایتون و کتابخانه Scapy برای ایجاد ابزارهای بررسی امنیت شبکه مختلف استفاده کنید. این ابزارها می‌توانند برای شناسایی و اصلاح آسیب‌پذیری‌های شبکه و دفاع در برابر حملات مختلف مورد استفاده قرار گیرند.

۶ .**اسکریپت نویسی و اتومیشن:** نوشتن یک اسکریپت برای خودکارسازی پردازش فایل‌های متنی. توضیح :در این مثال، می‌توانید از پایتون برای نوشتن اسکریپت‌هایی که عملیات خودکار و اتوماسیون تکالیف مرتبط با سیستم‌ها و فایل‌ها را انجام می‌دهند، استفاده کنید. توضیح :با استفاده از پایتون، می‌توانید اسکریپت‌هایی برای خودکارسازی تکالیف مختلفی ایجاد کنید، از جمله پردازش فایل‌های متنی، مدیریت فایل‌ها، پشتیبان‌گیری و انجام عملیات متعدد دیگر.

۷ .**محاسبات علمی و عددی:** حل یک معادله دیفرانسیل به کمک کتابخانه‌یSciPy .توضیح :در اینجا، می‌توانید از پایتون برای حل مسائل علمی و عددی استفاده کنید. با استفاده از کتابخانه‌ی SciPy و ابزارهای دیگر مانند NumPy وMatplotlib ، می‌توانید معادلات دیفرانسیل، انتگرالی و سایر مسائل ریاضی و علمی را حل کنید و نتایج را تجسم کنید.

۸ .**برنامه‌نویسی اینترنت اشیا:** توسعه یک برنامه کنترل خانه‌هوشمند با استفاده از پروتکل MQTT و کتابخانه‌ی Paho.توضیح :در اینجا، می‌توانید از پایتون برای ارتباط با دستگاه‌های مختلف اینترنت اشیاء (IoT) و کنترل آن‌ها استفاده کنید. این شامل کنترل دستگاه‌های خانه‌هوشمند، سنسورها، وسایل نورپردازی و غیره است.

۹ .**برنامه‌نویسی سیستم‌های توکار:** توسعه یک سیستم خودکارسازی برای مدیریت سیستم‌های شبکه با استفاده از کتابخانه Fabric.توضیح :در اینجا، می‌توانید از پایتون برای ایجاد اسکریپت‌های خودکارسازی برای مدیریت سیستم‌ها و شبکه‌ها استفاده کنید. این شامل مدیریت پکت‌ها، نصب و پیکربندی نرم‌افزارها، ویرایش فایل‌های پیکربندی و غیره می‌شود.

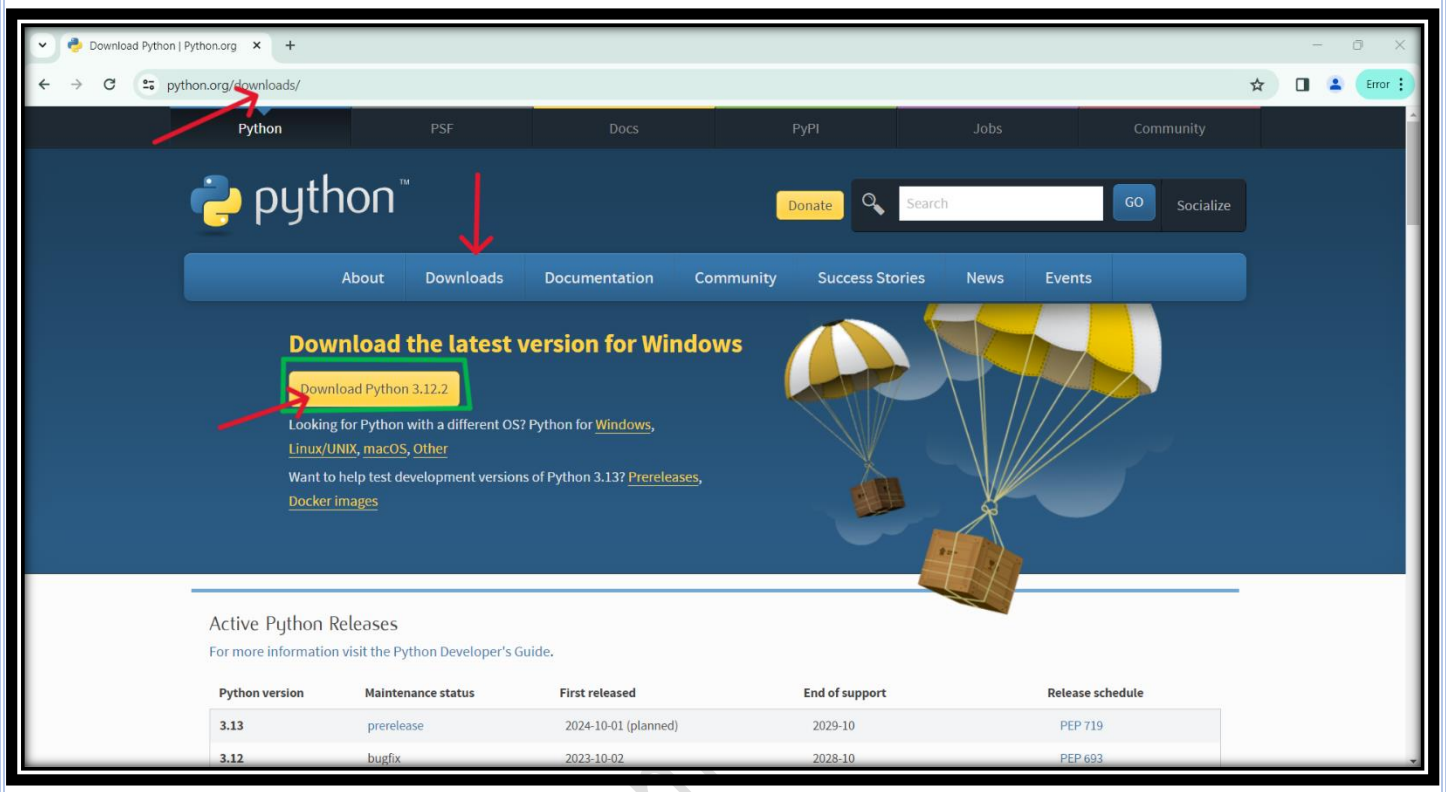
۱۰ .**حوزه مالی و فاین‌تک:** پیاده‌سازی الگوریتم‌های مالی برای پیش‌بینی بازار با استفاده از کتابخانهPandas .توضیح :در اینجا، می‌توانید از پایتون برای آنالیز داده‌های مالی و ایجاد الگوریتم‌های پیش‌بینی برای روند بازار و سرمایه‌گذاری استفاده کنید. این شامل تحلیل سرمایه‌گذاری، پیش‌بینی قیمت‌ها و روند بازار، مدیریت ریسک و غیره می‌شود.

و ...

آماده سازی محیط کار و آغاز کدنویسی:

الف) نصب Python:

به سادگی با سرچ کردن خوده کلمه "Python" در مرورگر و مراجعه با اولین سایت یعنی سایت python.org و رفتن به بخش دانلود میتوان فایل و ورژن مورد نیاز و سازگار با سیستم عامل را دانلود و به راحتی نصب کرد.



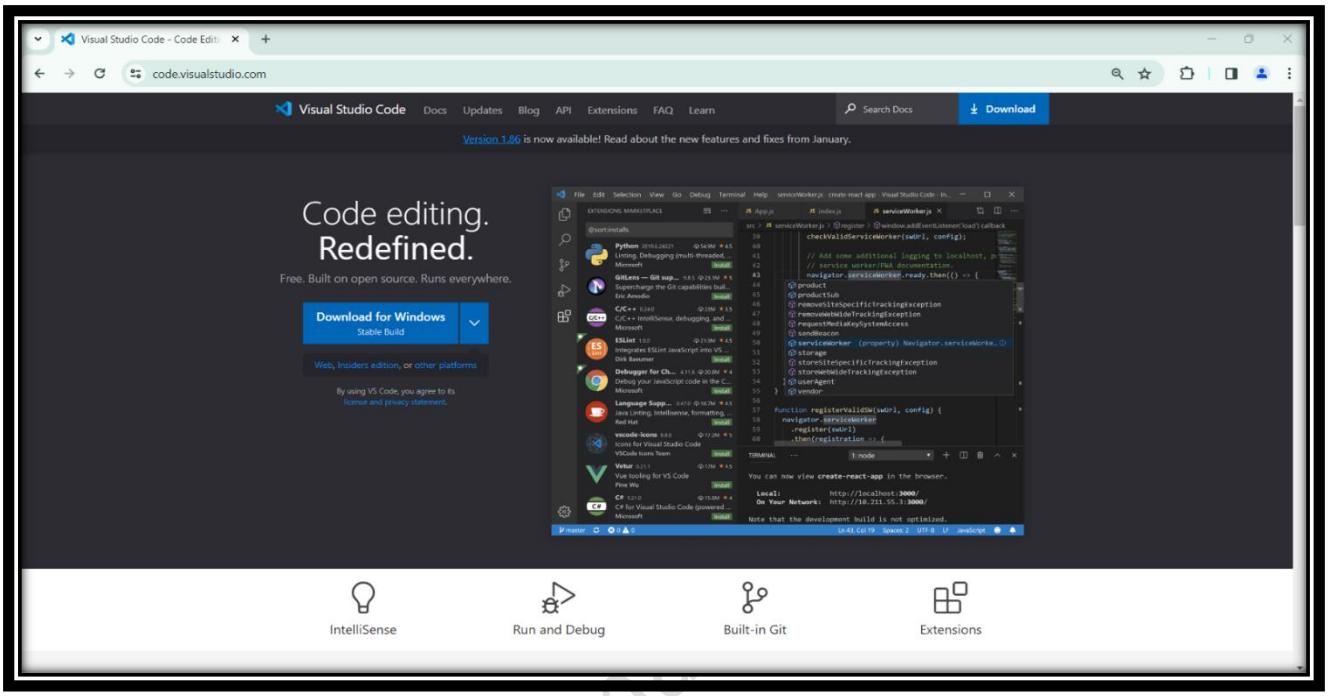
همان گونه که در تصویر بالا قابل مشاهده هست خوده سایت براساس سیستم عامل شما اخرین و بهترین نسخه را پیشنهاد می‌دهد، درقسمت‌های پایین‌تر نسخه‌های دیگر تحت عنوان "Releases" می‌توان یافت و بنا به نیاز دانلود و نصب کرد؛ به طور اگر این راه، راه انتخابی شما نباشد به راحتی میتوان از Microsoft Store آخرین نسخه پایتون راه دانلود کنید، یا اگر در سیستم عامل‌هایی هستید که کرنل آنها مبتنی بر لینوکس باشد با دستورهای دریافت مخازن مانند apt در اوبونتو یا ...:

```
┌──(root💀adm-VirusStove)-[~]
└─# apt-get install python3.6
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'libpython3.6-stdlib' for regex 'python3.6'
Note, selecting 'python3.6-2to3' for regex 'python3.6'
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
[SF] Your changes will be lost.
[SF] https://www.thc.org/segfault/faq/#lost
```

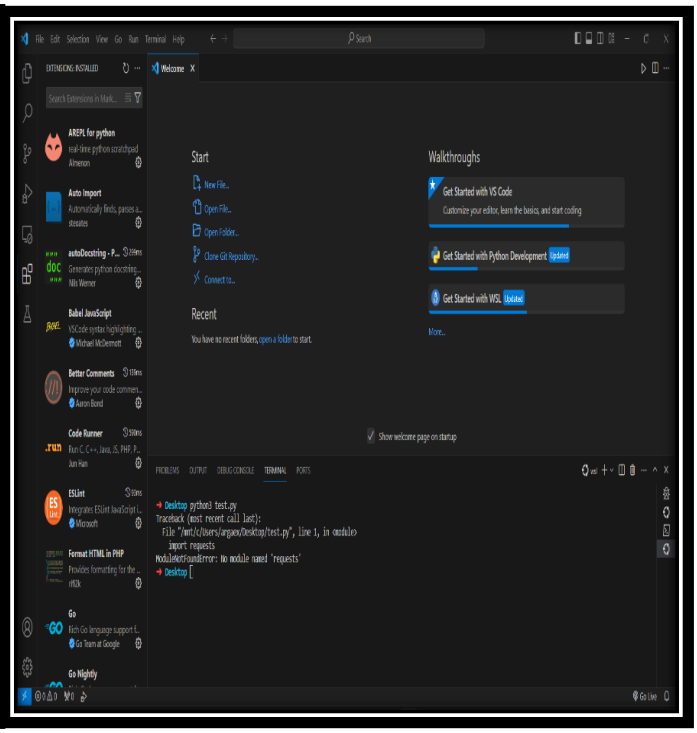
ب) آماده سازی و دانلود محیط کدنویسی یا نصب IDE(Integrated Development Environment):

خوده پایتون به صورت جداگانه در زمان نصب یک محیط کدنویسی و دیباگینگ دارد به نام IDLE، اما به دلیل اینکه خیلی قابل کاستوم شدن و ارور هندلینگ نیست برنامه نویسان از محیط‌های کدنویسی مثل VsCode, Sublime, pycharm و ... استفاده میکنند یا بعضا از کد ادیتورهای آنلاین استفاده میکنند، اما به طور محیط ساده و بسیار کارآمد برای معرفی VsCode بهترین گزینه است.(code.visualstudio.com)

برای دانلود کافی است "vscode" را سرچ کنید، به اولین سایت مراجعه کرده و به اولین سایت مراجعه فرمایید:



محیط آن به شکل زیر است:



کتابخانه‌ها در پایتون:

کتابخانه پایتون مجموعه ای از ماژول ها است که شامل توابع و کلاس هایی است که می تواند توسط برنامه های دیگر برای انجام وظایف مختلف استفاده شوند؛ تعریف خیلی ساده ماژول، باید گفت ماژول مجموعه کدی است که برای ساخت یک ساختار پیچیده مورد استفاده قرار می گیرد، که در برنامه‌های دیگر قرار است استفاده شود حلا یه به صورت بخشی از پروسه یا به صورت ماژولار؛ از ماژول ها می توان در برنامه های مختلف استفاده کرد و آن برنامه را حرفه ای تر کرد و فیچرهایی به آن اضافه کرد و کاستوم کرد؛ کتابخانه های پایتون را می توان در سه گروه دسته بندی کرد.

- گروه اول: کتابخانه هایی که به صورت پیش فرض در پایتون وجود دارند.
- گروه دوم: کتابخانه هایی که شما آن ها را ایجاد کرده اید.
- گروه سوم: کتابخانه هایی مانند PyPI که توسط منابع خارجی یا سایر افراد ایجاد شده است.

چند کتابخانه پر استفاده:

1) Requests :یکی از محبوب‌ترین کتابخانه‌های عمومی پایتون، کتابخانه Requests است که هدف آن آسان و خواناتر کردن درخواست‌های HTTP برای انسان است. تحت مجوز Apache2 و نوشته شده در پایتون، Requestsعملاً استانداردی است که توسط توسعه‌دهندگان برای ایجاد درخواست‌های HTTP در پایتون استفاده می‌شود. کاربرد اساسی این کتابخانه در ادامه آورده شده است.

2) Pillow: کتابخانه تصویربرداری پایتون (Python Imaging Library | PIL) «یک کتابخانه رایگان پایتون است که امکان پردازش تصویر را به مفسر پایتون اضافه می‌کند. به زبان سادهPIL ، اجازه دستکاری، باز کردن و ذخیره فرمت‌های مختلف فایل‌های تصویری را در پایتون می‌دهد .Pillow توسط الکس کلارک و سایر همکارانش ایجاد شده است. کاربرد اساسی این کتابخانه در ادامه آورده شده است.

3) Scrapy: فریمورک رایگان و منبع باز پایتونی است که به‌طور گسترده برای تجزیه و تحلیل خودکار اطلاعات در وب و تعدادی وظایف دیگر از جمله تست خودکار و داده کاوی استفاده می‌شود. کاربرد اساسی این کتابخانه در ادامه آورده شده است.

4) Asyncio: بسیاری از توسعه‌دهندگان پایتون در سراسر جهان از کتابخانه asyncio برای نوشتن کد همزمان از سینتکس «انتظار/ همگام‌سازی (async /await) «استفاده می‌کنند. کاربرد اساسی این کتابخانه در ادامه آورده شده است.

5) OpenCV :به عنوان یک کتابخانه پایتون، OpenCVاز توابع مختلفی تشکیل شده است، که آن را به ابزاری عالی برای برنامه‌های بینایی کامپیوتری بلادرنگ تبدیل می‌کند. این کتابخانه بسیار موثر می‌تواند ورودی‌های بصری متنوعی را، نه فقط از تصاویر بلکه از داده‌های ویدیویی نیز پردازش کند .OpenCV می‌تواند به شناسایی چهره‌ها، دست خط و اشیاء بپردازد. کاربرد اساسی این کتابخانه در ادامه آورده شده است.

6) NumPy: یک کتابخانه برای زبان برنامه نویسی پایتون (Python) است. با استفاده از این کتابخانه امکان استفاده از آرایه‌ها و ماتریس‌های بزرگ چند بعدی فراهم می‌شود. هم‌چنین می‌توان از تابع‌های ریاضیاتی سطح بالا بر روی این آرایه‌ها استفاده کرد.

7) Pandas :پانداس(Pandas) یک کتابخانه متن‌باز (Open Source) «با گواهینامه BSD است که کارایی بالا، ساختاری با قابلیت استفاده آسان و ابزارهای تحلیل داده برای» زبان برنامه‌نویسی پایتون (Python Programming Language) را فراهم می‌کند. در واقع، می‌توان گفت پانداس یک کتابخانه قدرتمند برای تحلیل، پیش‌پردازش (PreProcessing) و بصری‌سازی (Visualization)داده‌ها است.

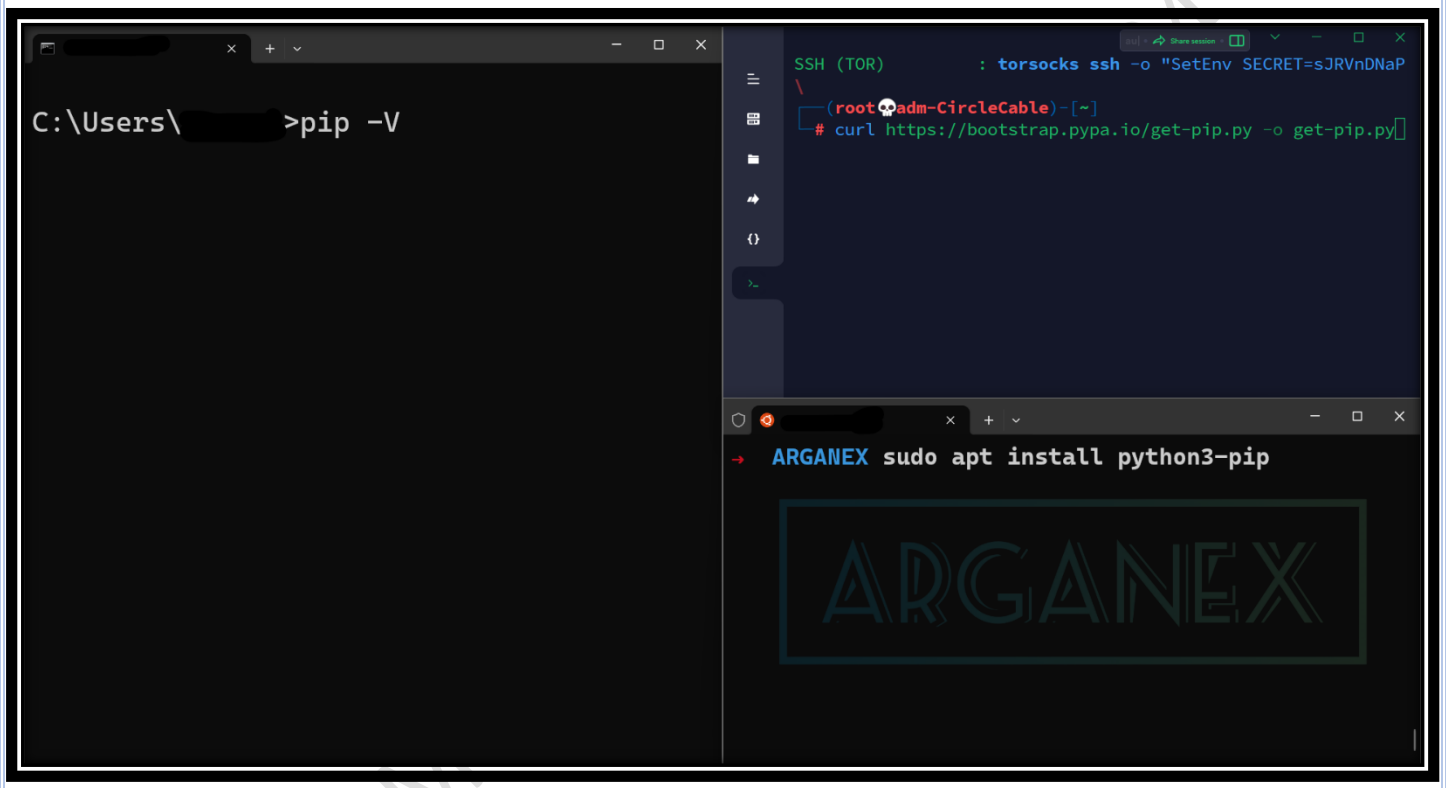
نحوه نصب و استفاده از کتابخانه‌ها:

نصب کتابخانه‌های خارجی توسط PIP در محیطی ترمینالی مثل CMD یا هر پوسته متصل و متعامل با سیستم عامل انجام میشود، یا به صورت دانلود مستقیم پکیج انجام میشود یا واسطه‌هایی مثل Anaconda که همان کار PIP را انجام میدهد.

خوده PIP چیست؟:

نصب کننده بسته پایتون (به انگلیسی: Python Installs Packages) (مخفف انگلیسی: pip) یک سامانه مدیریت بسته است که به زبان پایتون نوشته شده و برای نصب و مدیریت بسته‌های نرم‌افزاری مورد استفاده قرار می‌گیرد. این برنامه به یک مخزن آنلاین بسته‌های عمومی متصل می‌شود که فهرست بسته پایتون (PyPI) نامیده می‌شود.

چطور کار میکنه؟: (نصب)

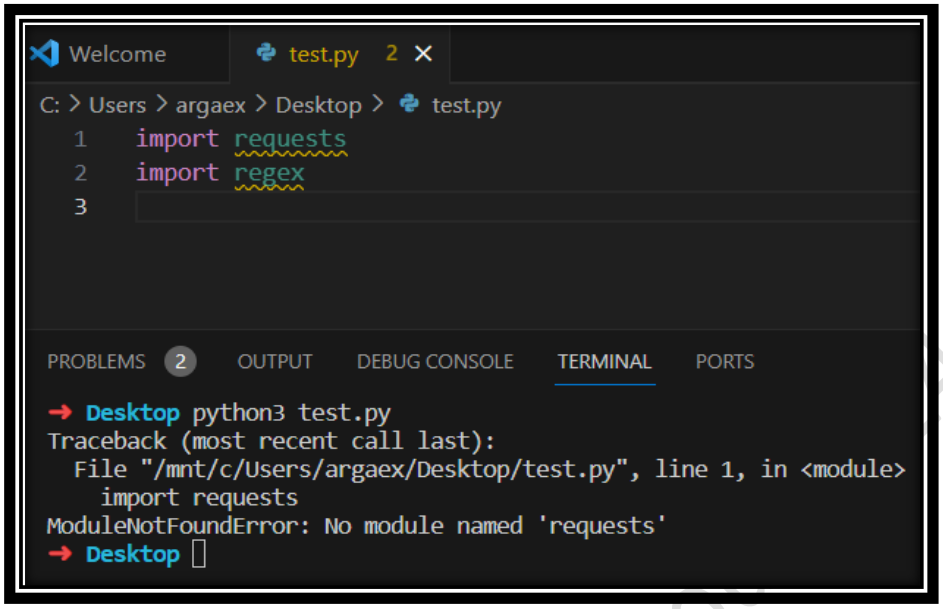


چطور از کتابخونه‌ها استفاده کنیم؟:

ابتدا اینکه برخی از کتابخانه‌های پایتون یا ماژول به صورت پیشفرض و built-in فقط نیاز به فراخوانی در زمان نیاز، دارند، مانند Print.

اما برای دیگر کتابخانه‌ها:

اگر به تصویر توجه کنید نحوه وارد کردن کتابخانه‌های زیر را مشاهده میکنید، ولی در زمان اجرا اروری مبنی بر اینکه چنین کتابخانه‌هایی موجود نمیباشند را مشاهده میکنید که دلیل آن عدم نصب کتابخانه‌ها است بر روی این سیستم.



برای نصب کتابخانه‌های دلخواه و مورد نیاز:

۱. نصب یک کتابخانه خاص: `pip install library_name`

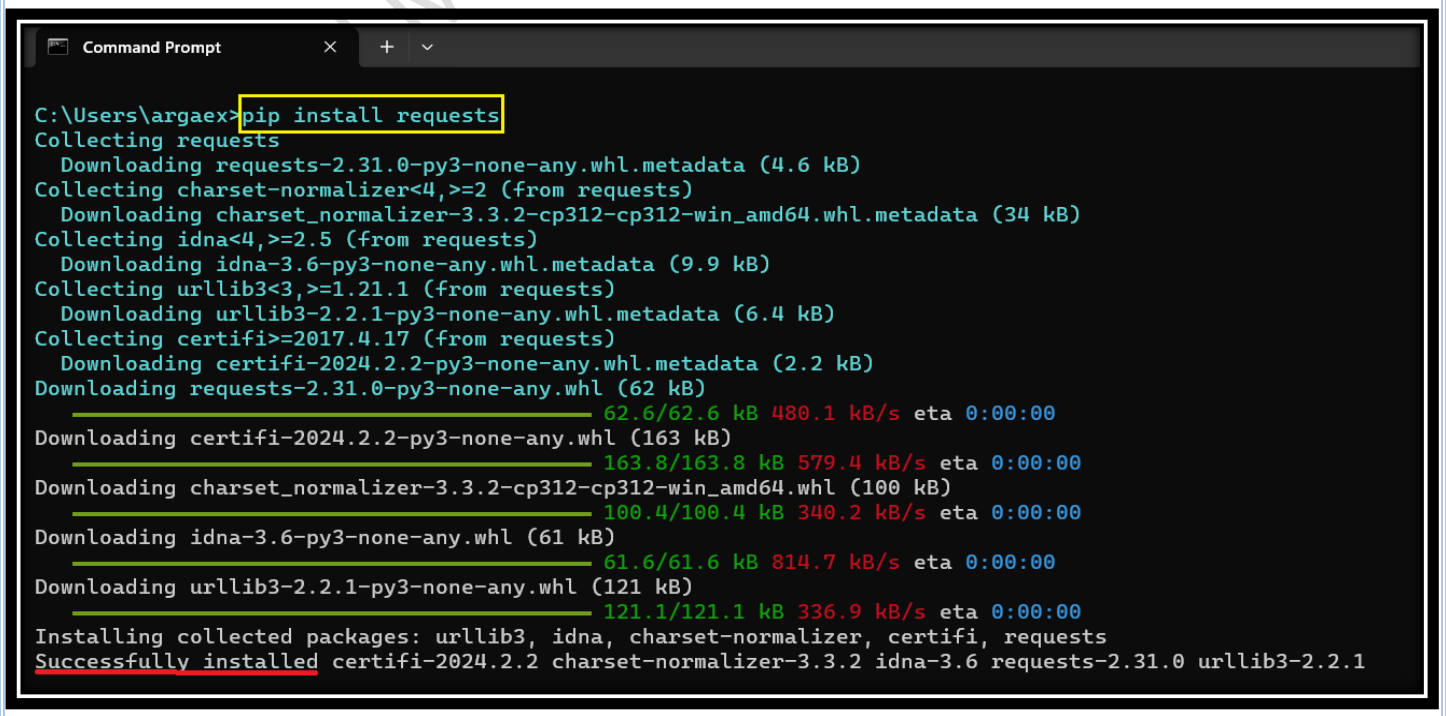
۲. نصب یک نسخه خاص از کتابخانه: `pip install library_name==version_number`

۳. نصب آخرین نسخه از کتابخانه: `pip install –upgrade library_name`

۴. نصب کتابخانه‌ها از یک فایل requirements: `pip install -r requirements.txt`

در فایل requirements.txt شما می‌توانید لیستی از کتابخانه‌های موردنیاز خود را به صورت زیر قرار دهید:

`library1==1.0.0 library2>=2.1.0 library3` در اینجا، library1 با نسخهٔ 1.0.0، library2 با نسخهٔ 2.1.0 یا بالاتر، و library3 با آخرین نسخه نصب خواهند شد.



شروع کردن مباحث پایتون:

اولین کد: نمایش یک **متغیر** یا هر نوع داده‌ای در خروجی.

Welcome | # Example 1: Simple print statement Untitled-1 | Extension: Python

```python
# Example 1: Simple print statement
print("Hello, world!")

# Example 2: Concatenating strings with print
name = "John"
age = 30
print("My name is", name, "and I am", age, "years old.")

# Example 3: Formatting output using f-strings (Python 3.6+)
product = "apple"
price = 0.5
quantity = 3
print(f"I want to buy {quantity} {product}s for ${price * quantity:.2f}.")

# Example 4: Printing with newline characters
print("First line")
print("Second line")

# Example 5: Printing without newline characters
print("First line", end="")
print("Second line")

# Example 6: Printing with custom separator
print("apple", "banana", "orange", sep=", ")

# Example 7: Printing with different separators and ending
print("apple", "banana", "orange", sep=", ", end="!\n")

```

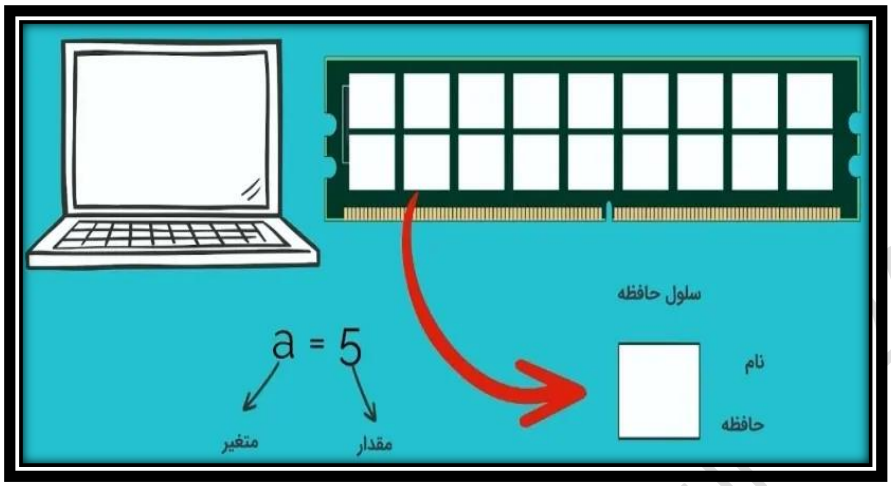
خروجی کد بالا:

..rgaex/Desktop

```
→  Desktop python3 Simple\ print\ statement.py
Hello, world!
My name is John and I am 30 years old.
I want to buy 3 apples for $1.50.
First line
Second line
First lineSecond line
apple, banana, orange
apple, banana, orange!
→  Desktop
```

تعریف متغیر:

متغیر (Variable) یک موقعیت نام‌گذاری شده است که برای ذخیره‌سازی داده‌ها در حافظه مورد استفاده قرار می‌گیرد. در واقع، می‌توان به متغیرها در پایتون به عنوان ظرف‌هایی نگریست که داده‌هایی را نگه‌داری می‌کنند که بعدا از طریق برنامه‌نویسی قابل تغییر هستند؛ هر متغیر داری مقدار یا value هست که مقادیر در پایتون تایپ‌های متفاوتی دارند.



برای نام گذاری یک متغیر در پایتون یک سری قوانین کلی هست:

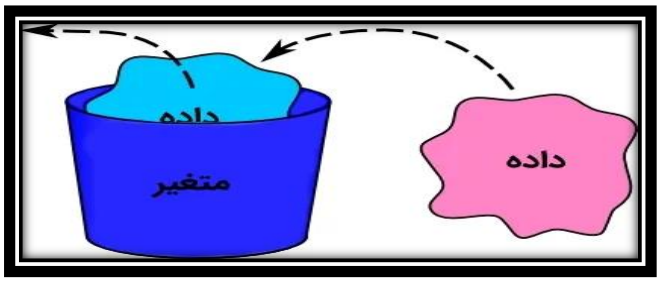
- ترکیب کاراکتر: نام متغیر به طور انحصاری می‌تواند حروف(a-z or A-Z)، اعداد و آندرلاین (_) را شامل شود.
- محدودیت اولیه :شروع نام متغیر با عدد غیرمجاز است.
- نام متغیر حساس به حروف بزرگ و کوچک است: بین my_variable و My_Variable تفاوت وجود دارد.
- نمی‌تواند شامل کاراکترهای غیرمجاز مانند . ,$,^ ,?, #باشد.
- نمی‌توان از کلمات رزرو شده در پایتون برای نام متغیر استفاده کرد.نام متغیر نباید دارای فضای خالی (spaces) باشد

برای افزایش بیشتر وضوح و انطباق برخی دیگر از قواعد نام‌گذاری به صورت موارد زیر هستند:

- توصیف مختصر: نام متغیر بهتر است معنی‌دار و مختصر باشند.
- استفاده از کوچک: بهتر است از حروف کوچک برای نام متغیر استفاده شود.
- انسجام چندکلمه‌ای: اگر تعیین نام متغیر شامل چندین کلمه است، بهتر است با آندرلاین کلمات از همدیگر جدا شوند. مثال: my_very_long_varible_name = 42

```
website = "soltan.com"
print(website) = > soltan.com

website = "mamad.com"
print(website) = > mamad.com
```



مثال بی ربط: (~_~)

عملا تعویض آب یا مایع درون پارچ با ریختن آب یا مایع بیشتر دیگر، باعث تعویض محتویات داخل پارچ می شود.

تایپ و انواع متغیرها Data Types :

Python has the following data types built-in by default, in these categories:

Text Type:	`str`
Numeric Types:	`int`, `float`, `complex`
Sequence Types:	`list`, `tuple`, `range`
Mapping Type:	`dict`
Set Types:	`set`, `frozenset`
Boolean Type:	`bool`
Binary Types:	`bytes`, `bytearray`, `memoryview`
None Type:	`NoneType`

داده	توضیح	
عددی (Numeric)	integer	شامل اعداد مثبت و منفی صحیح می باشد.
	float	شامل اعداد اعشاری می باشد.
	complex	شامل اعداد مختلط می باشد. نوع داده مختلط نوع غیر قابل تغییری است که یک جفت float را نگهداری می کند که یک بخش آن نشان دهنده قسمت حقیقی و یک بخش آن نشان دهنده قسمت موهومی عدد مختلط است. مانند (۳ + ۷.۱j)
رشته ای (String)	به مجموعه ای از کاراکترها که بین دو علامت کوتیشن یا دابل کوتیشن قرار گرفته باشند، اطلاق می شود.	
لیست (List)	مجموعه ای از آیتم ها هستند که بین دو علامت [] قرار گرفته و با علامت کاما (,) از هم جدا شده اند.	
تاپل (Tuple)	مجموعه ای از آیتم ها هستند که بین دو علامت () قرار گرفته و با علامت کاما (,) از هم جدا شده اند.	
دیکشنری (Dictionary)	مجموعه ای از آیتم ها هستند که به صورت کلید و مقدار بوده، بین دو علامت {} قرار گرفته، و با علامت کاما (,) از هم جدا شده اند.	
boolean	شامل دو مقدار true یا false می باشد.	

در پایتون کافیست که فقط نام متغیر را نوشته و به وسیله علامت مساوی یک مقدار به آن اختصاص دهیم:

```
1 intVar     = 10
2 floatVar   = 12.5
3 boolVar    = True
4 StringVar  = "Hello World!"
5 listVar    = [1,5,8]
6 tupleVar   = ("Python","Programming","Beginner")
7 dictionaryVar = {'Name': 'jack','family': 'Scalia','Age':7}
8
```

```
# چاپ کردن متغیرها:
print("intVar = {0}" .format(intVar))
print("floatVar = {0}" .format(floatVar))
print("boolVar  = {0}" .format(boolVar))
print("StringVar= {0}" .format(StringVar))
print("listVar= {0}" .format(listVar))
print("tupleVar= {0}" .format(tupleVar))
print("dictionaryVar= {0}" .format(dictionaryVar))
```

خروجی:

```
intVar        = 10
floatVar      = 12.5
boolVar       = True
StringVar     = Hello World!
listVar       = [1, 5, 8]
tupleVar      = ('Python', 'Programming', 'begginer')
dictionaryVar = {'Name': 'jack', 'family': 'Scalia', 'Age': 7}
```

در پایتون میتوان تایپ‌ها را به هم تبدیل کرد:

در زبان پایتون امکان تبدیل یک نوع به نوع دیگر وجود دارد که اصطلاحاً به آن Type Casting گفته می شود. پایتون دارای مجموعه ای از توابع از پیش تعریف شده است، که می توانند مقادیر را از یک نوع به نوع دیگر تبدیل کنند. در جدول زیر به برخی از این توابع اشاره شده است:

تابع	کاربرد
int(x)	x را به نوع صحیح تبدیل می کند.
long(x)	x را به نوع long تبدیل می کند.
float(x)	x را به نوع اعشار تبدیل می کند.
str(x)	x را به نوع رشته تبدیل می کند.
tuple(x)	x را به نوع tuple تبدیل می کند.
list(x)	x را به نوع list تبدیل می کند.
set(x)	x را به نوع set تبدیل می کند.
dict(x)	یک tuple که به صورت کلید/مقدار است را به نوع دیکشنری تبدیل می کند.
chr(x)	یک نوع صحیح را به کاراکتر تبدیل می کند
ord(x)	یک کاراکتر را به کد اسکی معادلش تبدیل می کند.

برای مثال:

```
x = 9.99
convertToInt = int(x)

x = 9
convertToFloat = float(x)

x =  'a'
convertToAscii = ord(x)

x = 97
convertToUnichar = chr(x)

x = (1, 2, 3, 4, 5)
convertToList = list(x)

x = [1, 2, 3, 4, 5]
convertToTuple = tuple(x)

x = (('one', 1), ('two', 2), ('tree', 3))
convertToDictionary = dict(x)

print(convertToInt)
print(convertToFloat)
print(convertToAscii)
. . . . . .
```

```
9
9.0
97
a
[1, 2, 3, 4, 5]
(1, 2, 3, 4, 5)
{'one': 1, 'two': 2, 'tree': 3}
```

در خط 1 یک متغیر از نوع اعشار تعریف و در خط 2 با استفاده از تابع int() آن را به نوع صحیح تبدیل کرده ایم، در این تبدیل بخش اعشار از بین می رود و خروجی عدد 9 می شود. در خط 19 هم یک نوع tuple که آیتم های آن به صورت کلید مقدار هستند را به نوع dictionaryتبدیل کرده ایم. برای به دست آوردن نوع یک متغیر هم می توان از تابع type() استفاده کرد. به مثال زیر توجه کنید:

```
x = 10.0
print(type(x))
<class 'float'>
```

همانطور که مشاهده می کنید، نتیجه نمایش کلمه float است. یعنی متغیر ما از نوع float می باشد .کافیست که نام متغیر را به تابع type()بدهیم.

عملگرها در پایتون:

عملگرها (Operators) سمبل‌های خاصی در پایتون هستند که پردازش‌های حسابی و منطقی را انجام می‌دهند.

```
>>> 2+3
5
```

عملگرهای حسابی:

عملگرهای حسابی (Arithmetic Operators) برای انجام پردازش‌های ریاضی مانند جمع، تفریق، ضرب و دیگر موارد استفاده می‌شود؛ در جدول زیر، کلیه عملگرهای حسابی موجود در پایتون ارائه و عملکرد آن‌ها همراه با مثالی شرح داده شده است.

عملگر	شرح عملکرد	مثال
+	جمع	x + y
-	تفریق	x - y
*	ضرب دو عمل‌وند	x * y
/	تقسیم کردن	x / y
%	محاسبه باقی‌مانده تقسیم	x % y (باقی‌مانده x/y)
//	خارج قسمت صحیح	x // y
**	x رساندن متغیر y به توان	x**y (x شده y به توان)

```
1   x = 15
2   y = 4
3
4   # Output: x + y = 19
5   print('x + y =',x+y)
6
7   # Output: x - y = 11
8   print('x - y =',x-y)
9
10  # Output: x * y = 60
11  print('x * y =',x*y)
12
13  # Output: x / y = 3.75
14  print('x / y =',x/y)
15
16  # Output: x // y = 3
17  print('x // y =',x//y)
18
19  # Output: x ** y = 50625
20  print('x ** y =',x**y)
```

```
x + y = 19
x - y = 11
x * y = 60
x / y = 3.75
x // y = 3
x ** y = 50625
```

عملگرهای مقایسه:

عملگرهای مقایسه (Comparison Operators) برای انجام مقایسه مقادیر مورد استفاده قرار می‌گیرند. متناسب با شرط، خروجی مقایسه برابر با True یا False خواهد بود.

مثال	شرح عملکرد	عملگر
x > y	بزرگ‌تر است از	<
x < y	کوچک‌تر است از	>
x == y	برابر است با تایپ و مقدار	==
x != y	نامساوی	=!
x >= y	بزرگ‌تر یا مساوی	=<
x <= y	کوچک‌تر یا مساوی	=>

```python
x = 10
y = 12

# Output: x > y is False
print('x > y  is',x>y)

# Output: x < y is True
print('x < y  is',x<y)

# Output: x == y is False
print('x == y is',x==y)

# Output: x != y is True
print('x != y is',x!=y)

# Output: x >= y is False
print('x >= y is',x>=y)

# Output: x <= y is True
print('x <= y is',x<=y)
```

```
[Running] python
x > y  is False
x < y  is True
x == y is False
x != y is True
x >= y is False
x <= y is True
```

عملگرهای منطقی:(Boolean)

عملگرهای منطقی(Logical Operators) در واقع or ، and و not هستند.

عملگر	شرح عملکرد	مثال
and	در صورتی که هر دو (True)باشند، (True)است.	x and y
or	در صورتی (True)است که یکی از عمل‌وندها (True)باشد.	x or y
not	در صورتی (True)است که (False)باشد.	not x

```
1  x = True
2  y = False
3
4  # Output: x and y is False
5  print('x and y is',x and y)
6
7  # Output: x or y is True
8  print('x or y is',x or y)
9
10 # Output: not x is False
11 print('not x is',not x)
```

```
[Running] python
x and y is False
x or y is True
not x is False
```

عملگرهای بیتی:

عملگرهای بیتی (Bitwise Operators) روی عمل‌وندهایی از نوع رشته یا ارقام دودویی (Binary) کار می‌کنند. همانطور که از نام این نوع عملگرها مشخص است، پردازش را بیت به بیت انجام می‌دهند.

عملگر	شرح عملکرد	مثال
&	AND بیتی (و بیتی)	x& y = 0 (0000 0000)
|	OR بیتی (یا بیتی)	x | y = 14 (0000 1110)
~	Bitwise NOT (نقیض بیتی)	~x = -11 (1111 0101)
^	XOR بیتی	x ^ y = 14 (0000 1110)
<<	شیفت به راست بیتی	x>> 2 = 2 (0000 0010)
>>	شیفت به چپ بیتی	x<< 2 = 40 (0010 1000)

مطالعه بیشتر، لینک.

توابع در پایتون:

تابع در پایتون گروهی از عبارت‌های مرتبط است که یک کار مشخص را انجام می‌دهند. توابع کمک می‌کنند تا برنامه به بخش‌های کوچک‌تر و دانه‌بندی شده‌ای ماژولار Modular شکسته شود. هرچه برنامه بزرگ و بزرگ‌تر شود، تابع‌ها به سازمان‌یافته‌تر و قابل مدیریت شدن آن کمک می‌کنند. علاوه بر این، توابع مانع از تکرار برنامه‌نویسی برای یک کار واحد می‌شوند و کد را قابل استفاده مجدد می‌کنند. (برای مطالعه بیشتر.)

```
1. def function_name(parameters):
2.     """docstring"""
3.     statement(s)
```

کلیدواژه def علامت آغاز سرآیند (Header) تابع استپ؛ نام تابع (function_name)که به صورت یکتا، تابع را مشخص کند. نام‌گذاری تابع در پایتون از قواعدی مشابه با قواعد نام‌گذاری شناساگرها (Identifiers) تبعیت می‌کند. پارامترها (آرگومان‌ها) که به وسیله آن‌ها، مقادیر به تابع پاس داده می‌شوند. آرگومان‌ها اختیاری هستند. یک علامت نقل قول یا دو نقطه (:)برای تعیین پایان هدر. یک دستور return اختیاری برای بازگرداندن یک مقدار از تابع؛ مثالی از تابع در پایتون:

```
1.def greet(name):
2.     """This function greets to
3.     the person passed in as
4.     parameter"""
5.     print("Hello, " + name + ". Good morning!")
```

def.py > ...

```
1  def add():
3      Example usage:
4          sum_result = add_two_numbers()
5          if sum_result is not None:
6          print("The sum is:", sum_result)
7      '''
8
9      try:
10         num1 = float(input("Enter the first number: "))
11         num2 = float(input("Enter the second number: "))
12         total = num1 + num2
13         return total
14         #print(total)
15     except ValueError:
16         print("Please enter valid numbers.")
17
18 print(add())
```

انواع تابع در پایتون:

تابع در پایتون به سه نوع تقسیم می شود:

1. توابع داخلی که به آنها Built-in Functions گفته میشود.
2. توابع ساخته شده توسط کاربر که User-Defined Functions نامیده میشوند.
3. توابع ناشناس که با نام Anonymous functions نیز شناخته میشوند.

توابع داخلی پایتون:

توابع داخلی یا built-in functions به توابعی گفته میشود که عملکرد آنها به صورت پیشفرض در مفسر پایتون قرار گرفته است و برای استفاده از آنها فقط کافیست نام آن تابع را فراخوانی کنید. در لیست زیر چند مورد از مهم ترین توابع داخلی پایتون را برایتان آورده ایم:

- تابع **print** که برای چاپ مقدار در خروجی استفاده میشود.
- تابع **input** که برای گرفتن ورودی از کاربر استفاده میشود.
- تابع **sorted** که برای مرتب کردن آبجکت ها استفاده میشود.
- تابع **round** که برای گرد کردن اعداد استفاده میشود.
- تابع **type** که برای مشخص کردن نوع یک آبجکت استفاده میشود.
- تابع **max** که برای مشخص کردن بیشترین مقدار استفاده میشود.
- تابع **sum** که برای جمع کردن چند عدد استفاده میشود.

*در این بخش فقط تعداد کمی از توابع داخلی پایتون را دیدید. پایتون دارای 71 تابع داخلی است.

توابع ساخته توسط کاربر در پایتون:

پایتون دارای توابع داخلی مفید بسیاری است. این توابع یک کار از پیش تعریف شده را انجام می دهند و می توانند در هر برنامه ای بنا به نیاز فراخوانی شوند. با این حال، اگر یک تابع داخلی مناسب برای هدف خود پیدا نکردید، می توانید یک تابع را ایجاد کنید. به این نوع توابع که توسط شخص برنامه نویس ایجاد میشوند User defined functions گفته میشود.

```
1. def show():
2.     pass
```

توابع ناشناس یا تک خطی در پایتون:

در پایتون، توابع ناشناس به توابعی گفته میشود که بدون نام باشند. در حالی که توابع معمولی با کلمه کلیدی def ساخته میشوند، برای ساخت توابع ناشناس باید از کلمه کلیدی lambda استفاده کنید. به همین دلیل توابع ناشناس، توابع لامبدا نیز نامیده میشوند. ساختار توابع لامبدا به شکل زیر است:

```
1. double = lambda x: x * 2
2. print(double(5))
```

شرط در پایتون:

زمانی به تصمیم‌گیری نیاز است که کاربر بخواهد یک کد صرفا در صورتی اجرا شود که یک شرط خاص صادق باشد. دستورات شرطی در پایتون برای تصمیم‌گیری مورد استفاده قرار می‌گیرند if…elif…else .از جمله دستورات شرطی پایتون است.؛ آن‌ها به برنامه شما اجازه می‌دهند تا با آزمایش شرایط و اجرای بلوک‌های کد خاص، به موقعیت‌ها یا ورودی‌های مختلف واکنش متفاوتی نشان دهد. در اینجا برخی از کاربردهای اصلی دستورات شرطی در پایتون آورده شده است:

- تصمیم‌گیری: عبارات شرطی برنامه را قادر می‌سازد تا تصمیم بگیرد. بر اساس شرایط خاص، برنامه می‌تواند تصمیم بگیرد که کدام دستورات یا بلوک‌های کد را اجرا کند.
- کنترل جریان برنامه: دستورات شرطی جریان اجرای برنامه را تعیین می‌کنند. آن‌ها می‌توانند به اجرای یک بلوک کد خاص در زمانی که شرایط خاصی برآورده شده است و یک بلوک کد متفاوت در صورت عدم وجود آن کمک کنند.
- پیاده‌سازی منطق پیچیده: عبارات شرطی تودرتو، یعنی گزاره‌های شرطی در سایر دستورات شرطی، امکان ایجاد ساختارهای منطقی پیچیده‌تر و فرآیندهای تصمیم‌گیری پیشرفته‌تری را فراهم می‌کند.
- ارزیابی ورودی کاربر: از عبارات شرطی می‌توان برای بررسی اعتبار ورودی کاربر استفاده کرد. به‌عنوان مثال، اگر سن کاربر را بپرسید، می‌توانید از یک عبارت شرطی استفاده کنید تا بررسی کند که آیا سن وارد شده یک مقدار قابل قبول است یا خیر.
- کارایی برنامه: گاهی اوقات، محاسبات یا عملیات سنگین فقط باید تحت شرایط خاص اجرا شوند. عبارات شرطی می‌توانند اطمینان حاصل کنند که این عملیات فقط در مواقع ضروری انجام می‌شود و کارایی برنامه را بهبود می‌بخشد.

الف) دستور if در پایتون:

```
1.if test expression:
2.      statement(s)
```

در بالا، برنامه test expression را ارزیابی می‌کند و تنها اگر True باشد، آن را اجرا می‌کند. اگر ارزیابی test expression برابر با False باشد، دستور اجرا نمی‌شود. در پایتون، بدنه دستور if به وسیله دندانه‌گذاری (Indentation) نمایش داده می‌شود. بدنه با یک دندانه (تورفتگی) آغاز می‌شود و اولین خط بدون دندانه، پایان دستور را نشان می‌دهد. پایتون مقادیر غیر صفر را به عنوان True در نظر می‌گیرد. در این زبان، Noneو 0 به عنوان False در نظر گرفته می‌شوند.

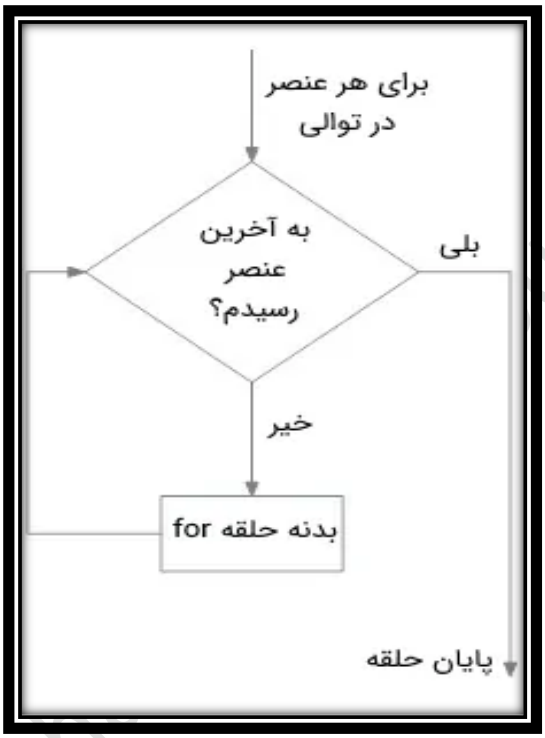
حلقه for در پایتون:

حلقه for در پایتون برای تکرار کردن کاری در یک توالی لیست(List) تاپل(Tuple) ،رشته (String) یا دیگر اشیای قابل تکرار، مورد استفاده قرار می‌گیرد. تکرار کردن کاری در یک توالی، «پیمایش (Traversal) «نامیده می‌شود .(برای مطالعه بیشتر)

```
for val in sequence:
        Body of for
```

در اینجا، val متغیری است که مقدار هر عنصر درون توالی را طی هر تکرار دریافت می‌کند. حلقه تا هنگامی ادامه پیدا می‌کند که به آخرین آیتم در توالی برسد. بدنه حلقهfor ، با استفاده از دندانه‌گذاری(Indentation) ، از کل کد جدا می‌شود.

فلوچارت حلقه for :



در قطعه کد زیر، مثالی از یک حلقه for در پایتون را مشاهده می‌کنید.

```
1  # List of numbers
2  numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]
3
4  # variable to store the sum
5  sum = 0
6
7  # iterate over the list
8  for val in numbers:
9      sum = sum+val
10
11 # Output: The sum is 48
12 print("The sum is", sum)
```

The sum is 48

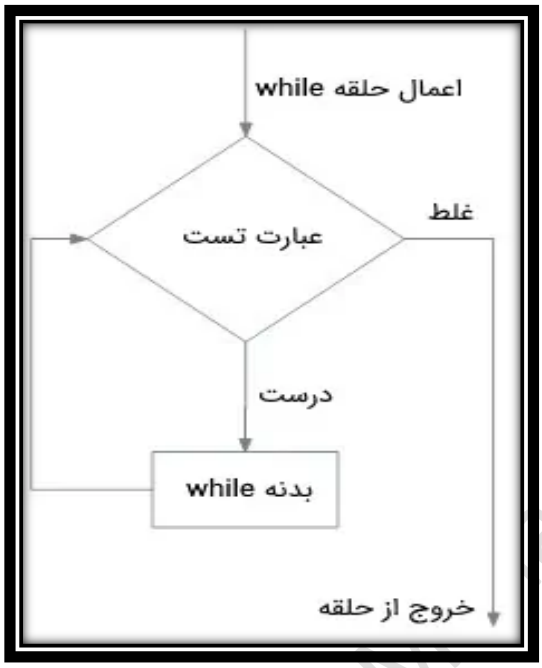
حلقه while در پایتون:

حلقه while در پایتون برای تکرار کردن یک بلوک از کد تا هنگامی که عبارت تست (شرط) صحیح باشد، مورد استفاده قرار می‌گیرد. معمولا از این حلقه‌ها هنگامی استفاده می‌شود که تعداد تکرارها از پیش معلوم نیست.

```
1.while test_expression:
2.    Body of while
```

در حلقهwhile ، عبارت تست (شرط) ابتدا چک می‌شود. بدنه حلقه تنها در صورتی اجرا می‌شود که حاصل ارزیابی test_expression برابر با True باشد. پس از یک تکرار، عبارت تست مجددا بررسی می‌شود. فرایند تا جایی ادامه پیدا می‌کند که حاصل ارزیابی test_expression برابر با False باشد. در پایتون، بدنه حلقه while از طریق دندانه‌گذاری (Indentation)شناسایی می‌شود. بدنه با دندانه آغاز می‌شود و اولین خط بدون تورفتگی (دندانه)، نشان‌گر پایان حلقه است. پایتون هر مقدار غیر صفری را به عنوان True شناسایی می‌کند .و صفر به عنوان False تفسیر می‌شوند.

فلوچارت حلقهwhile :



مثالی از حلقه while در پایتون:

```
n = 10
sum = 0
i = 1
while i <= n:
    sum = sum + i
    i = i+1
print("The sum is", sum)
```

در برنامه بالا(فلوچارت)، عبارت تست تا هنگامی درست (True) خواهد بود که متغیر شمارنده(Counter Variable) ، یعنی ، کوچکتر یا مساوی n باشد (۱۰ در برنامه بالا). در هر تکرار از حلقه، نیاز به افزایش متغیر شمارنده در بدنه حلقه است. این مسأله بسیار مهمی است که متاسفانه برخی از برنامه‌نویسان تازه‌کار آن را فراموش می‌کنند. انجام چنین کاری منجر به ایجاد حلقه‌های بی‌پایان یا حلقه‌های نامتناهی می‌شود. در نهایت، در حلقه‌ای که به درستی عمل کند و بی‌پایان نیز نباشد، نتایج در خروجی نمایش داده می‌شوند.

** Hand Tracking with MediaPipe and OpenCV**

Introduction:

-This Python script utilizes the MediaPipe library along with OpenCV to perform real-time hand tracking.

-The code identifies hands in a live video stream from the webcam and displays information about each detected hand.

Explanation:

** .1Importing Libraries**:

- The script begins by importing the necessary libraries:

`- mediapipe`: A powerful library for building real-time, cross-platform applications that process video streams.

`- cv2`: OpenCV, an open-source computer vision and machine learning software library.

** .2Defining Your Name**:

- The variable `my_name` is assigned the value "Fazel-3/8/2024". This string will be displayed in the video window to identify the creator or programmer.

** .3Displaying Your Name**:

- The function `display_name()` is defined, which adds your name to the video frame using `cv2.putText()`. This function accepts the current frame as an argument and overlays the name at the bottom of the frame.

** .4Initializing MediaPipe Hands Module**:

- The `mediapipe.solutions.hands` module is imported and aliased as `handsModule`. This module provides functionality for hand detection and tracking.

** .5Capturing Video Stream**:

- The code initializes the webcam using `cv2.VideoCapture(0)` to capture the video stream. The argument `0` indicates the default webcam device.

** .6Configuring MediaPipe Hand Tracking**:

- The `with` statement configures the MediaPipe hand tracking module with specific parameters:

`- static_image_mode=False`: Enables real-time hand tracking.

` - min_detection_confidence=0.7`: Sets the minimum confidence score required for hand detection. Lower confidence thresholds may lead to more false positives.

` - min_tracking_confidence=0.7`: Sets the minimum confidence score required for hand tracking. Lower values may result in less accurate hand tracking.

` - max_num_hands=2`: Specifies the maximum number of hands to track simultaneously. This parameter controls the resource allocation and processing load.

** .7Main Loop**:

- Inside the main loop, the script continuously reads frames from the video stream captured by the webcam.

** .8Processing Hand Landmarks**:

- The `hands.process()` method processes each frame to detect and track hands. It returns results containing information about the detected hand landmarks and their positions.

- If hands are detected (`results.multi_hand_landmarks`), the script iterates over each detected hand.

** .9Drawing Landmarks and Labels**:

- For each detected hand, the landmarks are drawn on the frame using `drawingModule.draw_landmarks()`. This function visualizes the landmarks, such as finger joints and palm keypoints, on the video frame.

- Hand labels (e.g., "Left-Hand 1", "Right-Hand 2") are added to the frame using `cv2.putText()`. This provides additional information about the detected hands, including their side (left or right) and index.

** .10Displaying Your Name**:

- Your name is displayed at the bottom of the video frame using the `display_name()` function. This personalized touch helps identify the creator of the software.

** .11Displaying Video Frame**:

- The modified frame, including hand landmarks and labels, is displayed in a window named "Frame" using `cv2.imshow()`. Users can observe the real-time hand tracking process and the associated information overlaid on the video stream.

** .12Exiting the Program**:

- The program continues to run until the user presses the "q" key. This key press event triggers the program to break out of the main loop and release the video stream resources.

Conclusion:

-This script demonstrates how to leverage MediaPipe and OpenCV libraries to develop real-time hand tracking applications in Python.

-By analyzing and processing video frames captured from the webcam, the script identifies hands, visualizes their landmarks, and provides contextual information about each detected hand.

-Through detailed explanations and code comments, new Python learners can gain insights into the underlying mechanisms of hand tracking algorithms and their implementation using popular computer vision libraries.