آموزش رایگان پایتون (python)- ایجاد حلقه‌های تودرتو در پایتون

در شماره گذشته آموزش پایتون به سراغ مبحث حلقه‌سازی در پایتون رفتیم و با نحوه ساخت و مدیریت حلقه for آشنا شدیم. در این شماره مبحث ساخت حلقه‌ها با فرمان while و نحوه به‌کارگیری حلقه‌ها به شکل تودرتو را بررسی خواهیم کرد.

برای مطالعه بخش بیستم و یکم آموزش رایگان پایتون اینجا کلیک کنید

فرمان pass

پایتون یکسری قابلیت‌های خاص در اختیارتان قرار می‌دهد که سایر زبان‌های برنامه‌نویسی فاقد چنین قابلیت‌هایی هستند. در شماره گذشته یاد گرفتیم که چگونه از دستور continue برای صرفنظر کردن از یک پردازش خاص استفاده کنیم، به گونه‌ای که از حلقه خارج نشویم و فقط از روی عنصری که نباید پردازش شود عبور کنیم. **پایتون** دستور دیگری به نام pass دارد که ظاهری شبیه به فرمان continue داشته، با این تفاوت که اجازه می‌دهد کدی که درون یک بلوک if قرار دارد اجرا شود. به عبارت دقیق‌تر دستور pass زمانی استفاده می‌شود که ما تنها به ساختار برنامه‌نویسی یک دستور نیاز داریم، به عبارت دقیق‌تر دستور pass یک عملیات تهی بوده و بیشتر در مکان‌هایی از یک برنامه قرار می‌گیرد که کدهای هنوز نوشته نشده‌اند و در آینده درج خواهند شد. برای روشن شدن این موضوع به مثال زیر دقت کنید.

1.IDLE را باز کرده و از منوی File گزینه New File را رانتخاب کنید.

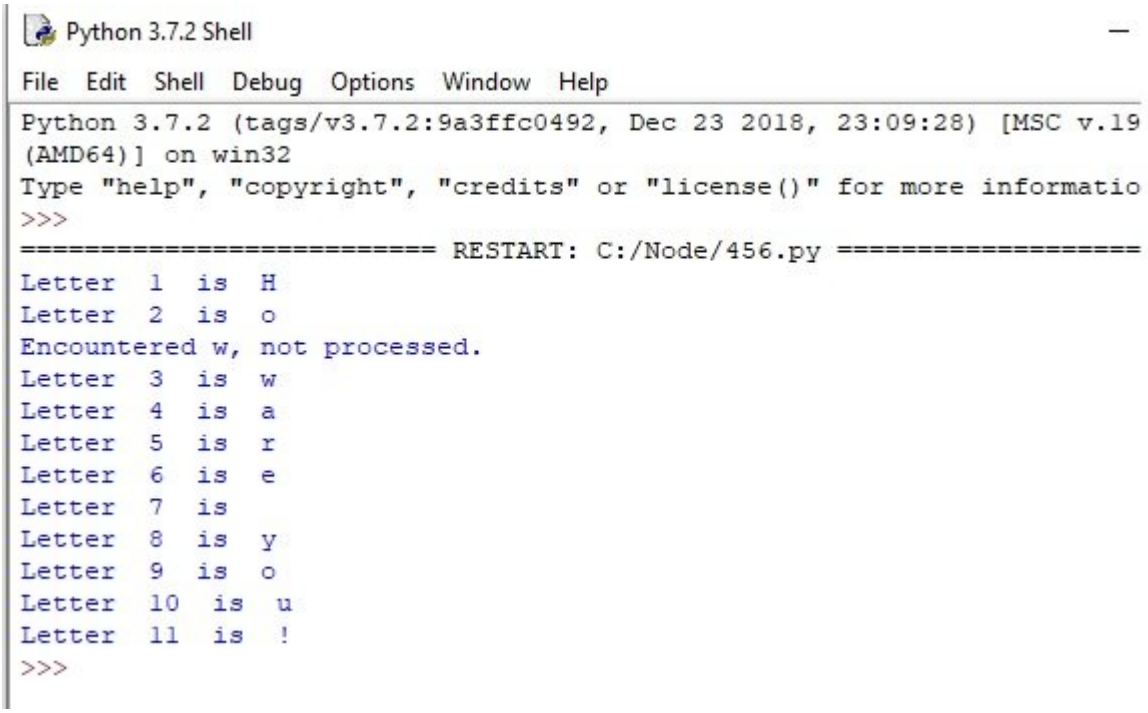
2.قطعه کد زیر را درون پنجره جدید وارد کنید.

```
LetterNumber = 1

for Letter in "Howare you!":

    if Letter=="w":

        pass

        print ("Encountered w, not processed.")
```

```
    print("Letter " , LetterNumber, " is ", Letter)

    LetterNumber +=1
```

قطعه کدی که مشاهده می‌کنید مشابه با قطعه کدی است که شماره قبل آن‌را نوشتیم و در آن قید کردیم که کاراکتر w درون عبارت چاپ نشود. اگر به خاطر داشته باشید در آن قطعه کد به شما گفتیم فرمان چاپی که پس از دستور if قرار دارد هیچ‌گاه اجرا نخواهد شد، زیرا دستور continue باعث پرش به ابتدای حلقه می‌شود. در قطعه کد فوق پس از آن‌که حلقه به کاراکتر w رسید و دستور pass اجرا شد، فرمان print که پس از pass قرار دارد اجرا می‌شود.

3. برنامه را ذخیره کرده و از منوی Run گزینه Run Module را انتخاب کنید.



کنترل اجرای حلقه با فرمان else

پایتون فرمان کنترلی دیگری برای مدیریت حلقه‌ها در اختیارتان قرار می‌دهد که شما در زبان‌های دیگر آن‌را پیدا نمی‌کنید. این فرمان کنترلی else نام دارد. بلوک else اجازه می‌دهد زمانی که شرط یک حلقه برقرار نیست و حلقه اجرا نمی‌شود دستوراتی درون این بلوک وارد کرده و به این مسئله رسیدگی کنید. به‌طور مثال، ممکن است مجبور شوید به کاربر اطلاع دهید که هیچ عنصری برای پردازش کردن وجود ندارد. برای روشن شدن موضوع به مثال زیر دقت کنید:

1. IDLE را باز کرده و از منوی File گزینه New File را رانتخاب کنید.

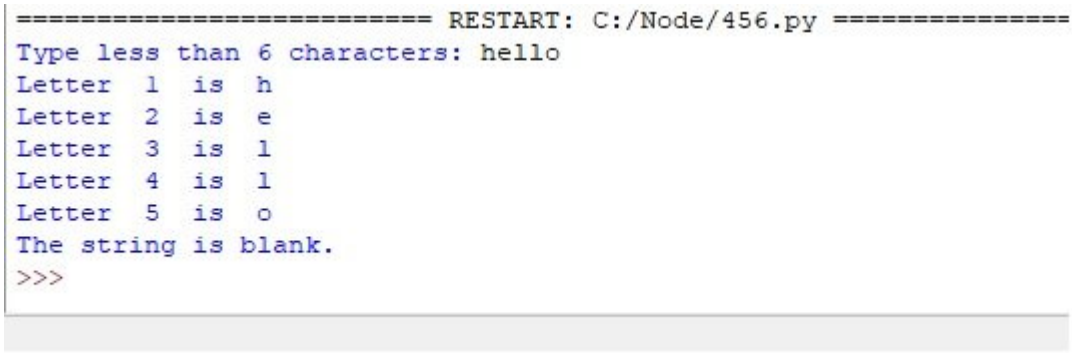
2. قطعه کد زیر را درون پنجره جدید وارد کنید.

```
Value = input ("Type less than 6 characters: ")

LetterNum = 1

for Letter in Value:

    print("Letter ", LetterNum, " is ", Letter)
```

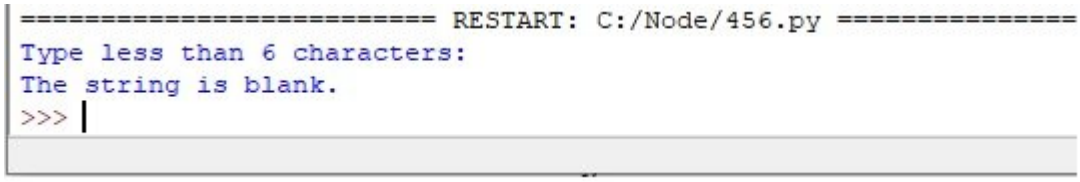
```
    LetterNum+=1
else:
    print("The string is blank.")
```

3. برنامه را ذخیره کرده و از منوی Run گزینه Run Module را انتخاب کنید.

4. اکنون واژه Hello را تایپ کرده و کلید اینتر را فشار دهید. خروجی این برنامه همانند تصویر زیر است.



5. برنامه را دومرتبه اجرا کرده و این مرتبه فقط کلید اینتر را فشار دهید. اکنون بخش else حلقه for اجرا خواهد شد.



در قطعه کد بالا زمانی که کاربر کلید اینتر را فشار می‌دهد به معنای آن است که هیچ رشته‌ای وجود ندارد که حلقه for کاراکترهای درون آن را شمارش کند، در این حالت بلوک else اجرا می‌شود. اما توجه داشته باشید که یک دنباله خالی در اصل به معنای یک خطای زمان اجرا یا وضعیتی است که باید به شیوه دستی و متفاوت از مثالی که به آن اشاره شد بررسی شود. در قطعه کد بالا هدف تنها نشان دادن نحوه به‌کارگیری دستور else با حلقه for بود.

پردازش داده‌ها با حلقه while

پایتون به شما اجازه می‌دهد به شیوه دیگری نیز یک حلقه بسازید. زمانی که به درستی اطلاع ندارید که برای پرداش داده‌ها به چند مرتبه تکرار دستورات نیاز دارید، در این حالت **پایتون** پیشنهاد می‌کند از حلقه while استفاده کنید. در این حالت به جای آن‌که برای **پایتون** به شرح صریح و مشخص تعداد دفعات اجرای دستورات را مشخص کنید، از حلقه‌ای که در ظاهر وضعیت بینهایت داشته، اما در حقیقت از طریق یک شرط منطقی کنترل می‌شود استفاده می‌کنید. این مدل حلقه‌ها به ویژه زمانی که در حال دانلود فایل‌هایی هستید که اندازه مشخصی ندارند یا زمانی که در حال استریم کردن داده‌های صوتی یا ویدیویی از منبعی هستید مفید هستند. در مجموع در هر وضعیتی که به درستی نمی‌دانید مجموعه‌ای از دستورات چند مرتبه باید تکرار شوند بهتر است از حلقه while استفاده کنید. یک حلقه while به جای آن‌که با یک دنباله کار کند بر مبنای یک شرط کار می‌کند. در این حالت مادامی که شرط حلقه در وضعیت درست (true) قرار دارد حلقه به کار خود ادامه می‌دهد. (البته عکس این قضیه نیز صادق است که بیشتر در ارتباط با شرایطی که بر مبنای مقادیر منطقی true و false کار می‌کنند، صدق می‌کند.) ترکیب نحوی حلقه while به شرح زیر است:

```
While Sum<5:
```

دستور بالا با کلمه کلیدی while آغاز می‌شود. در ادامه شرطی مشخص می‌شود. در خط بالا مقدار متغیری به نام Sum بررسی شده و مادامی که مقدار Sum کوچک‌تر از 5 باشد، حلقه به کار خود ادامه خواهد داد. دقت کنید در مثال بالا برنامه تنها یک مسئله را می‌داند، مادامی که مقدار Sum کمتر از 5 است حلقه باید تکرار شود، در این حالت این وظیفه شما است که با هر بار اجرای حلقه یک مقدار به Sum اضافه کنید تا برنامه در یک حلقه بینهایت گرفتار نشود. زمانی که قصد دارید از یک حلقه while استفاده کنید، باید سه کار زیر را حتما انجام دهید:

1. ابتدا متغیری را تعریف کنید. (به‌طور مثال پیش از اجرای یک حلقه متغیری از نوع صحیح تعریف کرده و مقداری را به عنوان شروع برای آن تعریف کنید. (Sum=0)

2. وضعیت متغیر یا عبارت مشخص شده درون حلقه while بررسی کنید. (به‌طور مثال Sum < 5)

3. وضعیت شرط را به‌روزرسانی کنید تا اطمینان حاصل کنید که حلقه در وضعیت بینهایت قرار نخواهد گرفت. Sum+=1

همانند حلقه for در ارتباط با حلقه while نیز شما می‌توانید رفتار این حلقه را با استفاده از اصلاح‌کننده‌های زیر کنترل کنید

Break: به کار حلقه جاری پایان دهید

Continue: بدون تاخیر به ابتدای حلقه پرش کرده و از پردازش عنصر جاری صرفنظر کنید.

Else: به بررسی وضعیتی بپردازید که اگر حلقه به دلیل عدم وجود شرط لازم اجرا نشد، تهمیدات لازم در نظر گرفته شود.

برای آن‌که با نحوه کار حلقه while بهتر آشنا شوید به مثال زیر دقت کنید:

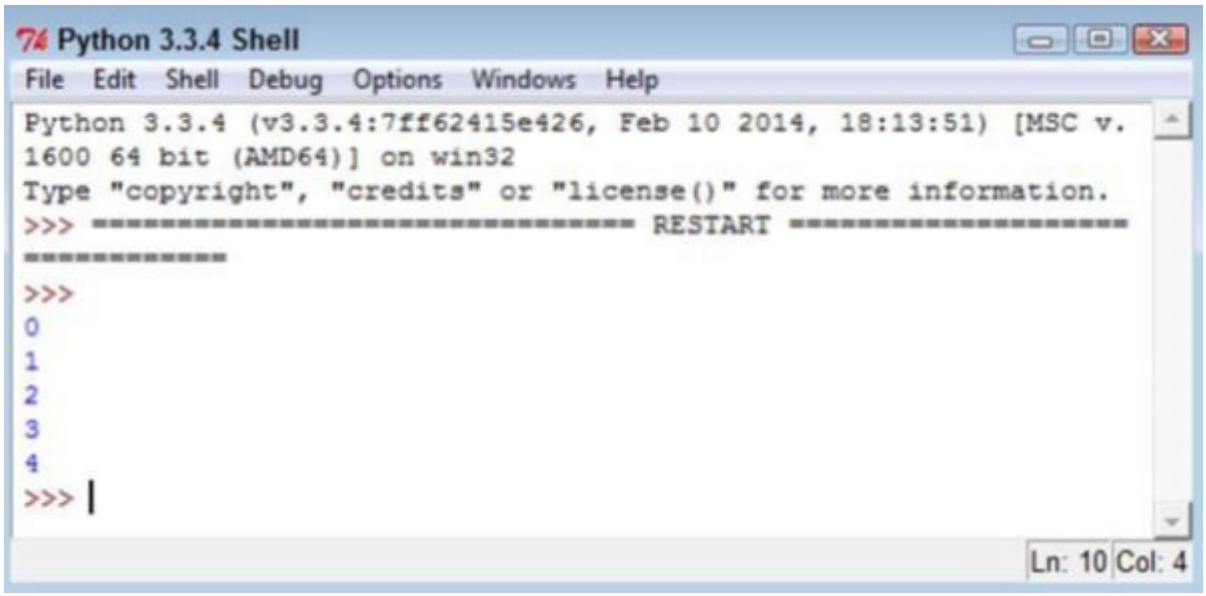
1.محیط IDLE را باز کرده و از منوی File گزینه New File را انتخاب کنید.

2. قطعه کد زیر را درون پنجره جدید وارد کنید.

```
Sum = 0

while Sum < 5:

    print(Sum)

    Sum+=1
```

قطعه کد بالا سه کاری که باید برای ساخت یک حلقه به گونه‌ای که به شکل درستی اجرا شده و پایان پذیرد را نشان می‌دهد. در قطعه کد بالا مقدار 0 به متغیر Sum تخصیص داده می‌شود که اولین گام ساخت یک شرط است. این وضعیت خودش در قالب بخشی از یک دستور while شناخته می‌شود. در هر بار اتمام حلقه یک واحد به sum افزوده می‌شود تا زمانی که مقدار Sum بزرگ‌تر از مقداری شود که در حلقه while به آن اشاره شده است. زمانی که مقدار sum برابر با 5 شد حلقه دیگر اجرا نشده و کار آن پایان می‌پذیرد.

3. برنامه را ذخیره کرده و از منوی Run گزینه New Module را انتخاب کنید. خروجی این حلقه همانند تصویر زیر است:



حلقه‌های تودرتو

اما حلقه‌ها همیشه به این شکل ساده استفاده نمی‌شوند. شما می‌توانید ترکیبی از دو حلقه را داشته باشید. به عبارت دقیق‌تر حلقه‌های for و while را به شکل تودرتو استفاده کنید. عملکرد حلقه‌ها به شکل متداخل یکسان بوده، اما رفتار آن‌ها کمی متفاوت است. برای روشن شدن بهتر مطلب اجازه دهید یک برنامه جدول ضرب بنویسیم که از هر دو حلقه استفاده می‌کند. (در قطعه کد زیر دستوراتی را مشاهده می‌کنید که برای فرمت‌بندی رشته‌ها استفاده می‌شوند. در شماره‌های آتی با نحوه کار این دستورات آشنا خواهیم شد، در حال حاضر هدف آشنایی با حلقه‌های متداخل است.)

1. IDLE را باز کرده و از منوی File گزینه New File را انتخاب کنید.

2. دستورات زیر را درون پنجره باز شده وارد کنید.

```
X = 1

Y =1

print('{:>4}'.format(' '), end= ' ')

for X in range(1,11):

    print('{:>4}'.format(X),end=' ')


print()


for X in range(1,11):

    print('{:>4}'.format(X), end=' ')

    while Y<=10:

        print('{:>4}'.format(X*Y),end=' ')
```

```
        Y+=1
    print()
    Y=1
```

قطعه کد بالا با تعریف دو متغیر X و Y کار خود را آغاز کرده که این دو متغیر برای نگه داشتن مقادیر مربوط به سطرها و ستون‌هایی که قرار است یک جدول ضرب را به وجود آورند استفاده می‌شوند. X متغیر سطر بوده و Y بیان‌گر متغیر ستون است. برای آن‌که جدول فوق قابل خواندن باشد، باید تیتری که بیان‌گر اعداد 1 تا 9 است در بالای جدول قرار گیرد. زمانی که کاربر مقدار 1 را در بالا و در سمت چپ مشاهده می‌کند، با ضرب این دو مقدار در یکدیگر نتیجه را در ستون پایین مشاهده می‌کند. اولین فرمان print برای اضافه کردن یک فاصله استفاده می‌شود تا اعداد بهتر مشاهده شوند. در همه دستوراتی که برای قالب‌بندی استفاده شده ما از 4 کاراکتر فضای خالی استفاده کردیم. بخش {:>4} قطعه کدی است که اندازه یک ستون را تعیین می‌کند. تابع format(‘ ’) برای مشخص کردن فضایی که قرار است نشان داده شود استفاده می‌شود. اولین حلقه برای نمایش مقادیر 1 تا 9 در بالای جدول استفاده می‌شود. تابع range یک مجموعه ترتیبی از اعداد را ایجاد می‌کند. زمانی که از تابع Range استفاده می‌کنید، شما باید یک مقدار شروع که در این مثال عدد 1 است و یک مقدار انتهایی که در این مثال مقدار 11 است را مشخص کنید. در این مقطع، مکان‌نما در پایان ردیفی که تیتر را مشخص می‌کند قرار می‌گیرد. برای انتقال مکان‌نما به خط بعد تابع print را بدون آنکه محتوایی را چاپ کند فراخوانی می‌کنیم. با توجه به این‌که قصد داریم یک جدول ضرب 1*1 تا 10 *10 را چاپ کنیم باید ده سطر و ده ستون در اختیار داشته باشیم تا اطلاعات را نشان دهند. حلقه for به پایتون می‌گوید که ما به ده سطر نیاز داریم. اگر به شکل زیر دقت کنید سر تیتر را مشاهده می‌کنید. فراخوانی تابع print مقدار سطر تیتر را نشان می‌دهد. البته شما باید این اطلاعات را قالب‌بندی کنید. این کد از یک فضای خالی چهار کاراکتر برای نمایش بهتر اطلاعات استفاده می‌کند. در ادامه حلقه while اجرا می‌شود. این حلقه ستون‌ها را در هر ردیف جداگانه چاپ می‌کند. مقادیر ستون‌ها حاصل‌ضرب دو مقدار X*Y هستند. ما حاصل ضرب را با چهار کاراکتر فرمت‌بندی کرده و روی صفحه نشان می‌دهیم. در انتهای حلقه while یک واحد به مقدار Y افزوده می‌شود تا زمانی‌که مقدار Y بزرگ‌تر از 10 شود. زمانی که این‌گونه شد برنامه از حلقهwhile خارج شده و دومرتبه به حلقه for باز می‌گردد. فرمان print برای پایان دادن به سطر جاری اجرا شده و مقدار Y برابر با 1 می‌شود، اکنون برنامه دومرتبه به حلقه while وارد شده و حاصلضرب سطر بعدی را چاپ می‌کند. این روند ادامه پیدا می‌کند تا زمانی که دو مقدار 10 در 10 در یکدیگر ضرب شوند.

3. برنامه را ذخیره کرده و از منوی Run گزینه Run Module را انتخاب کنید. خروجی این برنامه همانند تصویر زیر است:

```
========================== RESTART: C:/Node/456.py ======================
       1    2    3    4    5    6    7    8    9   10
   1   1    2    3    4    5    6    7    8    9   10
   2   2    4    6    8   10   12   14   16   18   20
   3   3    6    9   12   15   18   21   24   27   30
   4   4    8   12   16   20   24   28   32   36   40
   5   5   10   15   20   25   30   35   40   45   50
   6   6   12   18   24   30   36   42   48   54   60
   7   7   14   21   28   35   42   49   56   63   70
   8   8   16   24   32   40   48   56   64   72   80
   9   9   18   27   36   45   54   63   72   81   90
  10  10   20   30   40   50   60   70   80   90  100
>>>
```

در شماره آینده آموزش **پایتون** به سراغ مباحث دیگر این زبان برنامه‌نویسی خواهیم رفت.

تاریخ انتشار:

16 اسفند 1397

نشانی منبع:
https://www.shabakeh-mag.com/workshop/programming/14767/%D8%A2%D9%85%D9%88%D8%B2%D8%B4-%D8%B1%D8%A7%DB%8C%DA%AF%D8%A7%D9%86-%D9%BE%D8%A7%DB%8C%D8%AA%D9%88%D9%86-python-%D8%A7%DB%8C%D8%AC%D8%A7%D8%AF-%D8%AD%D9%84%D9%82%D9%87%E2%80%8C%D9%87%D8%A7%DB%8C-%D8%AA%D9%88%D8%AF%D8%B1%D8%AA%D9%88-%D8%AF%D8%B1-%D9%BE%D8%A7%DB%8C%D8%AA%D9%88%D9%86