آموزش رایگان پایتون (python)- فرمت‌بندی و جستجو در رشته، مفهوم فهرست در پایتون

در شماره گذشته آموزش پایتون با مبحث رشته‌ها و پیاده‌سازی عملیات مختلف روی رشته‌ها آشنا شدیم و به ذکر مثال‌هایی در این زمینه پرداختیم. در این شماره قصد داریم نکات دیگر مرتبط با رشته‌ها را بررسی خواهیم کرد.

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

جست‌وجو، جایگزینی، شمارش و.... درون یک رشته

در بیشتر موارد نیاز دارید درون رشته‌ای که از کاربر یا یک بانک‌اطلاعاتی دریافت می‌کنید، جست‌وجویی، عبارتی را جایگزین عبارت دیگری کرده یا یک رشته را درون رشته دیگری اضافه کنید. پایتون توابع مختلفی برای جست‌وجو در رشته‌ها در اختیارتان قرار می‌دهد. توابع زیر جزء معروف‌ترین توابع پایتون هستند:

count(*str*, *beg*= 0, *end*=len(*string*))

این تابع تعداد تکرار یک رشته را داخل رشته دیگری شمارش می‌کند.

endswith(*suffix*, *beg*=0, *end*=len(*string*)

یک رشته را دریافت کرده و اگر رشته اصلی با رشته مدنظر پایان یافته باشد مقدار True و در غیر این صورت مقدار False را باز می‌گرداند. دو پارامتر این رشته نقطه شروع و پایان این عمل را مشخص می‌کند.

find(*str*, *beg*=0, *end*=len(*string*))

رشته‌ای را درون رشته دیگر جست‌وجو می‌کند. در این حالت ایندکس اولین موردی که پیدا شده است باز گردانده می‌شود.

index(*str*, *beg*=0, *end*=len(*string*)

عملکرد این تابع شبیه به تابع find است، با این تفاوت که اگر پارامتر مشخص شده در رشته اصلی پیدا نشد، پیغام خطای ValueError را نشان خواهد داد.

replace(*old*, *new* [, *max*])

این تابع برای جایگزینی یک رشته با رشته‌ای که درون پرانتز مشخص شده است استفاده می‌شود.

rfind(*str*, *beg*=0, *end*=len(*string*))

عملکرد این تابع همانند تابع find بوده با این تفاوت که آخرین ایندکس را باز خواهد گرداند.

rindex(*str*, *beg*=0, *end*=len(*string*))

عملکرد این تابع نیز همانند تابع rfind بوده، با این تفاوت که اگر پارامتر مشخص شده در رشته اصلی پیدا نشود، پیغام خطای ValueError را نشان خواهد داد.

startswith(*prefix*, *beg*=0, *end*=len(string))

تابع فوق ارزیابی می‌کند که رشته مدنظر با رشته مشخص شده شروع شده است یا خیر. مقدار خروجی این تابع از نوع منطقی است.

پیدا کردن داده‌ها یا ویرایش رشته‌ها یکی از مهم‌ترین وظایف دنیای برنامه‌نویسی است، اما برای آشنایی بهتر با نحوه عملکرد هر یک از توابع بالا به مثال زیر دقت کنید.

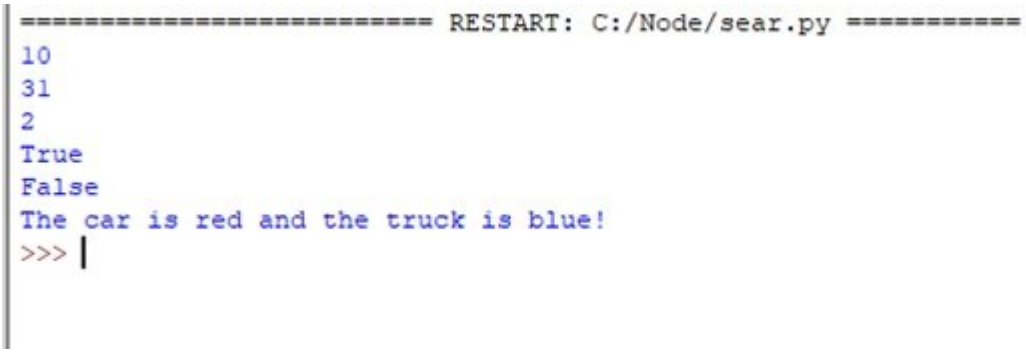
IDLE.1 را باز کرده و از منوی File گزینه New File را انتخاب کنید.

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

```
SearchMe = "The apple is red and the berry is blue!"

print(SearchMe.find("is"))

print(SearchMe.rfind("is"))

print(SearchMe.count("is"))

print(SearchMe.startswith("The"))

print(SearchMe.endswith("The"))

print(SearchMe.replace("apple", "car")

.replace("berry", "truck"))
```

قطعه کد بالا با تعریف متغیری به نام SearchMe کار خود را آغاز می‌کند که درون این متغیر دو واژه is قرار دارد. تکرار این واژه درون رشته فوق مهم است، زیرا ما قصد داریم فرآیند جست‌وجوی یک کلمه درون یک رشته، شمارش تعداد تکرارها و.... را با استفاده از این واژه به شما نشان دهیم. زمانی که از تابع find استفاده می‌کنید، فرآیند جست‌وجو از ابتدای رشته آغاز می‌شود. در مقابل زمانی که از تابع rfind استفاده می‌کنید جست‌وجو از انتهای رشته آغاز می‌شود. البته شما همیشه نمی‌دانید که یک مجموعه کاراکتری چند مرتبه درون یک رشته تکرار شده‌اند. تابع count به شما اجازه می‌دهد تعداد تکرار یک رشته را بررسی کنید. بسته به نوع داده‌ای که با آن کار می‌کنید، گاهی اوقات داده‌ها به شدت قالب‌بندی شده‌اند و شما مجبور هستید از الگوی خاصی برای جست‌وجوی یک رشته استفاده کنید. به‌طور مثال، شما می‌توانید بررسی کنید که آیا یک رشته با مجموعه‌ای خاص از کارکترها پایان پذیرفته است یا خیر. بخش انتهایی قطعه کد بالا apple را با var و berry را با truck جایگزین می‌کند. دقت کنید در مثال ما، فرآیند جایگزین کردن کد در دو خط قرار گرفته است. در برخی موارد، کدهای شما باید در چند خط ظاهر شوند تا خوانایی کدها حفظ شود.

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



فرمت‌بندی رشته‌ها

پایتون به شما اجازه می‌دهد رشته‌های خود را به روش‌های مختلفی قالب‌بندی کنید. ما رشته‌ها را از آن جهت فرمت‌بندی می‌کنیم تا خوانایی آن‌ها بهبود پیدا کرده و به راحتی قابل درک شوند. فرمت‌بندی رشته‌ها به معنای اضافه کردن فونت‌های خاص یا جلوه‌های ویژه نیست. بلکه فقط به نحوه ارائه داده‌ها اشاره دارد. به‌طور مثال، کاربر ممکن است تمایل داشته باشد اعدادی که مقدار اعشاری دارند را به روال معمول مشاهده کند. در حالی که روش‌های مختلفی برای فرمت‌بندی رشته‌ها وجود دارد اما تابع format اصلی‌ترین مکانیزمی است که برای فرمت‌بندی رشته‌ها از آن استفاده می‌شود. با استفاده از این تابع و براکت‌ها {} شما قادر هستید از کاراکترهای ویژه‌ای برای فرمت‌بندی استفاده کنید. به‌طور مثال {0} به اولین عنصری که درون یک رشته قرار دارد اشاره دارد، {1} به دومین عنصر و به همین ترتیب. خصلت‌هایی که برای فرمت‌بندی از آن‌ها استفاده می‌شود همراه با یک دو نقطه : ظاهر می‌شوند. زمانی که شما تنها به دنبال یک فرمت‌بندی خاص هستید، از براکت‌ها همراه با دو نقطه و کاراکتری استفاده می‌کنید که برای فرمت‌بندی استفاده می‌شود. به‌طور مثال {f:} باعث می‌شود که خروجی اعداد در صفحه‌نمایش دارای نقطه ثابت باشند. اگر به دنبال عدد کامل هستید باید از ترکیب {f:0} استفاده کنید، در این حالت در خروجی یک عدد ثابت قرار می‌گیرد. ترکیب نحوی که برای فرمت‌بندی عناصر استفاده می‌شود به شرح زیر است:

[[*fill*]*align*][*sign*][#][0][*width*][,][.*precision*][*type*]

توضیح هر یک از خصلت‌های بالا به شرح زیر است:

Fill: می‌تواند هر کاراکتر قابل چاپی باشد. از این گزینه برای پر کردن فضای خالی استفاده می‌شود.

Align: طراز داده‌ها درون صفحه‌نمایش را مشخص می‌کند. شما می‌توانید از کاراکترهای زیر استفاده کنید:

<: چپ‌چین

>: راست‌چین

^: وسط‌چین

Sign: برای نمایش علامت اعداد استفاده می‌شود.

+: اعداد مثبت یک علامت مثبت و اعداد منفی یک علامت منفی خواهند داشت.

-: اعداد منفی یک علامت منفی خواهند داشت.

<space>: اعداد مثبت با یک فضای خالی و اعداد منفی با یک علامت منفی نشان داده خواهند شد.

#: تعیین می‌کند که خروجی باید از فرمت نمایش پیشنهادی برای اعداد استفاده کند. به‌طور مثال، اعداد هگزا همراه با پیشوند 0x نشان داده خواهند شد. این تکنیک اجازه می‌دهد که بتوانید قالبی برای نمایش اعداد در پایه‌های مختلف تعیین کنید. تبدیل پایه با استفاده از کاراکترهای b برای مبنای 2، o برای مبنای 8 و x برای مبنای شانزده استفاده می‌شود.

Width: برای تعیین اندازه رشته استفاده می‌شود.

,: کاما برای تفکیک اعداد استفاده شده و اجازه می‌دهد یک عدد را سه رقم سه رقم از سمت راست جدا کرد.

precision : تعداد کاراکترهایی که پس از نقطه اعشار ظاهر می‌شوند را مشخص می‌کند.

type: نوع خروجی را حتا اگر مغایر با نوع ورودی باشد تعیین می‌کند. نوع‌ها به سه گروه زیر تقسیم می‌شوند:

- String: از کاراکتر s یا هیچ کاراکتری برای یک رشته مشخص شده استفاده می‌شود.
- Interger: برای نوع‌های صحیح می‌توان از کاراکترهای b (باینری)، c (کاراکتر)،d (اعشاری)، o (اوکتاو)، x (هگزادسیمال با حروف کوچک)، X (برای هگزادسیمال با حروف بزرگ) و n استفاده کرد.
- Floating point برای نوع‌های نقطه اعشاری استفاده می‌شود. e (عدد علمی یا همان نپرین که از جداکننده e با حرف کوچک استفاده می‌کند)، E (عدد علمی یا همان نپرین که از جداکننده e با حرف بزرگ استفاده می‌کند)، f (نقطه اعشار با حرف کوچک)، F (نقطه اعشار با حرف بزرگ)،g (فرمت‌بندی کلی با حرف کوچک)، G (فرمت‌بندی کلی با حرف بزرگ)، n و % استفاده می‌شود.

دقت کنید خصلت‌هایی که به آن‌ها اشاره شد باید در مکان درست خود درج شوند، زیرا پایتون نمی‌داند منظور شما از به‌کارگیری این کاراکترها چیست. به‌طور مثال اگر از خصلت طرازبندی پیش از خصلت کاراکتر پر کننده استفاده شود، پایتون پیغام خطایی نشان خواهد داد. برای درک بهتر نحوه فرمت‌بندی یک رشته اجازه دهید در یک مثال عملی این قالب‌ها را استفاده کنیم.

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

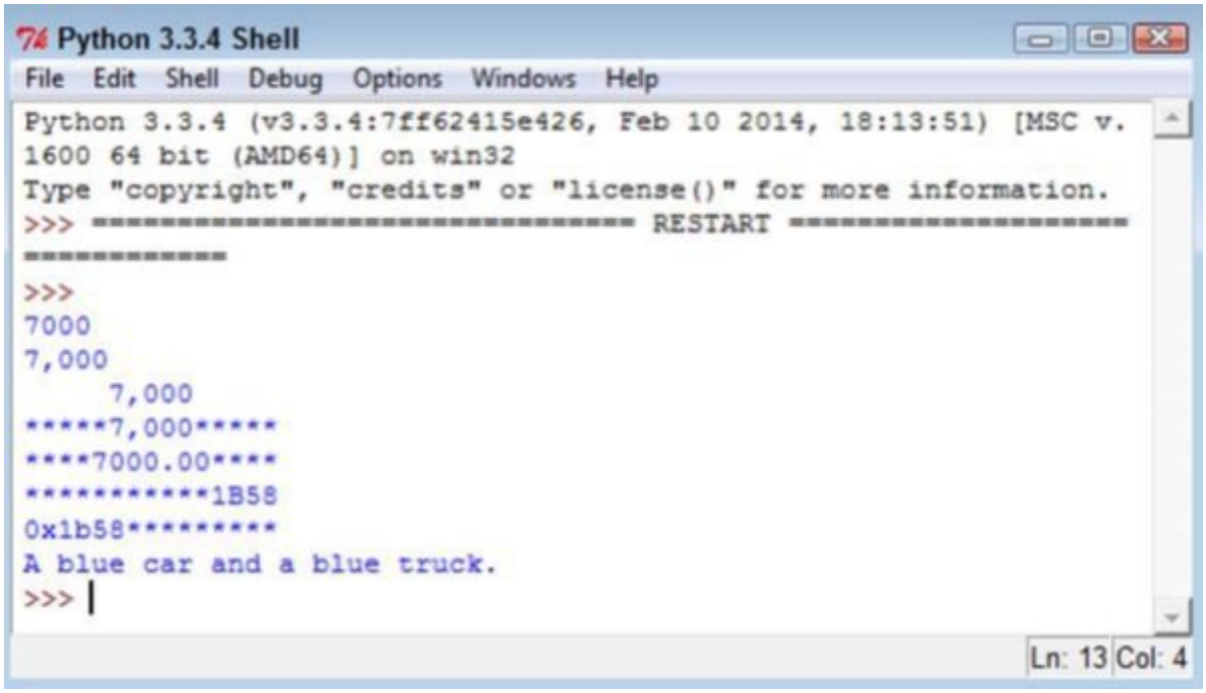
2. در پنجره جدید قطعه کد زیر را وارد کنید.

```
Formatted = "{:d}"

print(Formatted.format(7000))

Formatted = "{:,d}"

print(Formatted.format(7000))

Formatted = "{:^15,d}"

print(Formatted.format(7000))

Formatted = "{:*^15,d}"

print(Formatted.format(7000))

Formatted = "{:*^15.2f}"

print(Formatted.format(7000))

Formatted = "{:*>15X}"
```

```
print(Formatted.format(7000))

Formatted = "{:*<#15x}"

print(Formatted.format(7000))

Formatted = "A {0} {1} and a {0} {2}."

print(Formatted.format("blue", "car", "truck"))
```

قطعه کد بالا با تعریف یک متغیر فرمت‌بندی کننده مقدار اعشار کار خود را آغاز می‌کند. در ادامه یک جداکننده مقدار هزارتایی استفاده شده است. در مرحله بعد یک فیلد مشخص کننده عرض تعریف شده تا داده‌ها در مرکز نشان داده شوند. در نهایت، به این فیلد یک کاراکتر ستاره اضافه شده که در خروجی نشان داده می‌شود. البته نوع‌های داده‌ای دیگر نیز در این مثال وجود دارند. در مرحله بعد همان داده‌ها در فرمت نقطه اعشار استفاده شده‌اند. در این مثال مقدار هگزا هم با حروف بزرگ و هم با حروف کوچک نشان داده شده است. خروجی حروف بزرگ با راست‌چین و خروجی حروف کوچک با چپ‌چین نشان داده شده است. در نهایت، این مثال به شما نحوه استفاده از فیلدهای عددی را نشان داده است. دو خط آخر این قطعه کد نشان می‌دهد که چگونه از الگوی {0} برای دسترسی به مقادیر درون یک رشته دسترسی پیدا کنید. در آخرین خط مشاهده می‌کنید که ما از تکنیک قالب‌بندی برای درج عبارت and a در میان نام سه ماشین چگونه استفاده کرده‌ایم.

3. برنامه را ذخیره کرده و از منوی Run گزینه New Module را انتخاب کنید. خروجی قطعه کد بالا همانند شکل زیر است.



ما سعی کردیم، نکات مهمی که در مورد رشته‌ها وجود دارد را در این مقاله به شما نشان دهیم. البته موارد دیگری نیز در ارتباط با رشته‌ها وجود دارند که پیشنهاد می‌کنیم با صرف کمی وقت آن نکات را نیز بیاموزید.

مقدمه‌ای بر فهرست‌ها

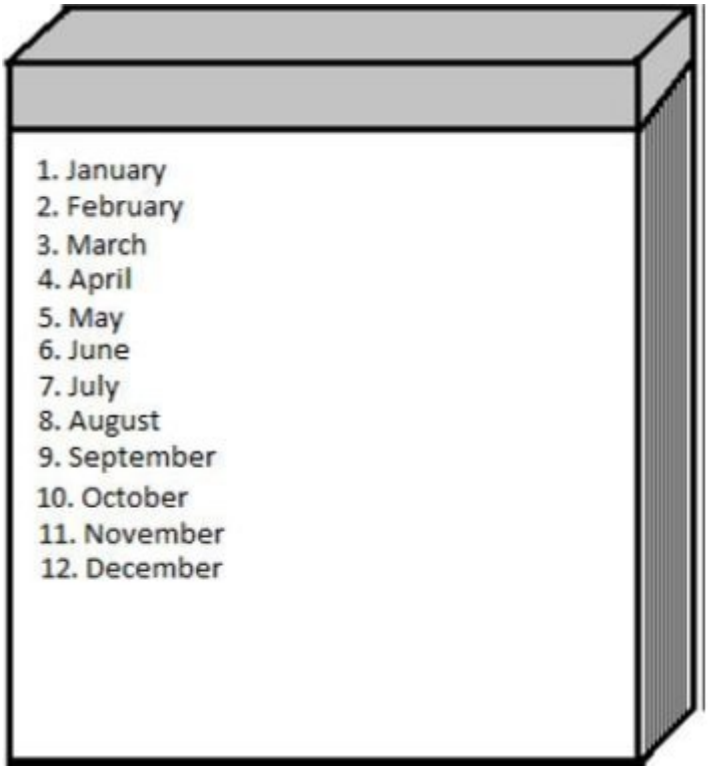
بیشتر مردم از این واقعیت غافل هستند که بسیاری از تکنیک‌های برنامه‌نویسی بر مبنای الگوهای دنیای واقعی ساخته شده‌اند. به‌طور مثال، بیشتر مردم برای ذخیره کردن وسایل خود از یک جعبه استفاده می‌کنند، برنامه‌نویسان نیز با الهام گرفتن از این مفهوم متغیرها را ابداع کردند. یکی دیگر از مفاهیم پر کاربرد دنیای واقعی فهرست‌ها هستند. هر شخصی بر مبنای نیاز کاری و سلیقه خود به روش‌های مختلفی اطلاعات را درون فهرست‌ها سازمان‌دهی می‌کند. در

حقیقت، شما سعی می‌کنید از فهرست‌ها برای سازمان‌دهی هرچه بهتر اطلاعات استفاده کنید. بهترین مثالی که در این زمینه می‌توان به آن اشاره کرد، فهرست کتاب‌ها است که اجازه می‌دهند به ساده‌ترین شکل به مفاهیمی که درون یک کتاب قرار دارند دسترسی پیدا کنید. پایتون نیز راهکار جالبی برای مرتب‌سازی اطلاعات در قالب فهرست‌ها در اختیارتان قرار می‌دهد. یکی از مهم‌ترین مسائلی که باعث می‌شود افراد در کار کردن با فهرست‌ها در **پایتون** با مشکل روبرو شوند، به این دلیل است که آن‌ها در مورد فهرست‌هایی که ایجاد می‌کنند به درستی فکر نمی‌کنند. زمانی که فهرستی ایجاد می‌کنید، ممکن است بدون انجام یک محاسبه ساده شروع به درج مقادیر درون فهرست‌ها کنید. در مجموع باید بگوییم هر کاری که با فهرست‌ها در دنیای واقعی انجام می‌دهید در **پایتون** نیز می‌توانید همان کارها را با فهرست‌ها انجام دهید. تنها تفاوتی که وجود دارد این است که باید برای **پایتون** تشریح کنید که فهرست‌ها قرار است چه کاری انجام دهند. فهرست‌ها در **پایتون** فوق‌العاده مهم هستند. ما در مقاله شماره آینده به شما نشان می‌دهیم که چگونه فهرست‌ها را ایجاد کرده، آن‌ها را مدیریت کرده، جست‌وجویی در آن‌ها انجام داده، فهرست‌ها را چاپ کرده و کارهای دیگر مرتبط با فهرست‌ها را انجام دهید. پس از آشنایی با فهرست‌ها متوجه خواهید شد که این عناصر تا چه اندازه برنامه‌های شما را انعطاف‌پذیرتر و قدرتمندتر می‌کنند.

سازمان‌دهی اطلاعات درون یک برنامه کاربردی

پایتون به فهرست‌ها به عنوان یک نوع دنباله نگاه می‌کند. دنباله‌ها به سادگی این امکان را در اختیارتان قرار می‌دهند تا عناصر داده‌ای را در یک مکان واحد اما به شکل موجودیت‌های جدا از هم ذخیره کنید. **پایتون** از انواع مختلفی از دنباله‌ها پشتیبانی می‌کند که Queues، Stacks، Dictionaries، Tuples و Deques از جمله این موارد هستند.

در میان دنباله‌هایی که به آن‌ها اشاره کردیم؛ فهرست‌ها ساده‌تر از ننوع‌های دیگر درک شده و یک مثال عینی از فهرست‌هایی هستند که در دنیای واقعی از آن‌ها استفاده می‌کنید. کار کردن با فهرست‌ها به شما کمک می‌کند شناخت دقیق‌تری در مورد سایر دنباله‌ها به دست آورید. فهرست‌ها دارای یک شروع، وسط و یک پایان هستند. تصور زیر آیتم‌های شماره‌گذاری شده در یک فهرست را نشان می‌دهند.



در شماره آینده آموزش **پایتون** مبحث فهرست‌ها را بررسی خواهیم کرد.

تاریخ انتشار:

21 اسفند 1397

نشانی منبع:
https://www.shabakeh-mag.com/workshop/programming/14787/%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-%D9%81%D8%B1%D9%85%D8%AA%E2%80%8C%D8%A8%D9%86%D8%AF%DB%8C-%D9%88-%D8%AC%D8%B3%D8%AA%D8%AC%D9%88-%D8%AF%D8%B1-%D8%B1%D8%B4%D8%AA%D9%87%D8%8C-%D9%85%D9%81%D9%87%D9%88%D9%85-%D9%81%D9%87%D8%B1%D8%B3%D8%AA-%D8%AF%D8%B1