ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/313011537

Fundamental of computer and programming with Python (in Persian language)

Book · January 2017

CITATIONS
0

READS
2,058

1 author:

Mohsen Rahmanian
Jahrom University

14 PUBLICATIONS **60** CITATIONS

SEE PROFILE

All content following this page was uploaded by Mohsen Rahmanian on 28 January 2017.

The user has requested enhancement of the downloaded file.

فصل سوم

زبان برنامه نویسی پایتون

در فصل‌های گذشته با سختار کامپیوترها و همچنین نحوهٔ حل مسئله به وسیلهٔ الگوریتم آشنا شدیم؛ حال وقت آن فرا رسیده که با زبان برنامه‌نویسی پایتون به عنوان ابزاری که می‌تواند استفاده از این دانش‌ها را در محیط واقعی عملی نماید بیشتر آشنا شویم.

۳-۱ معرفی پایتون

شکل ۳-۱ گایدو ون رسم طراح زبان پایتون

اولین نسخهٔ پایتون در سال ۱۹۸۹ توسط آقای گایدو وَن‌رُسُم[1] (شکل ۱-۳) به عنوان جانشین زبان ABCدرکشور هلند ارائه شد. گایدو نام زبانش را از یک گروه کمدی بریتانیایی به نامMonty Python’s Flying Circus که در دهۀ ۱۹۷۰ مشغول به فعالیت بودند گرفته است. نام محیط توسعهٔ IDLE که همراه مفسر پایتون ارائه می‌شود نیز از ابتدای نام اعضای همین گروه استخراج شده است.

نحو زبان پایتون بسیار ساده است؛ برنامه‌هایی که توسط آن نوشته می‌شود روشن می‌باشند و به راحتی خوانده و فهمیده می‌شوند. این درحالی است که پایتون خصوصیات حرفه‌ای زیادی را در اختیار کاربرانش قرار می‌دهد؛ نشانهٔ آن استفاده زیاد از این زبان برنامه‌نویسی است. Yahoo، Google، YouTube و NASA از جمله سازمانها و شرکتهایی هستند که از پایتون در برنامه هایشان استفاده می‌کنند.

مفسر پایتون برای تمامی سیستم عامل‌ها پیاده سازی شده است و توسط توسعه‌دهندگانش به خوبی پشتیبانی می‌شود. می‌توان آنرا از وبگاه www.python.org بصورت رایگان دریافت کرد

1. Guido Van Rossum

(برای اطلاعات کامل‌تر در مورد نحوهٔ دریافت و نصب بر روی سیستم عامل‌های مختلف پیوست ب را ببینید).

۳-۱-۱ PyCharm محیط برای توسعهٔ برنامه های پایتون

PyCharm یک محیط توسعهٔ یکپارچه[1] یا IDE می‌باشد. هر IDE از مجموعه‌ای از نرم‌افزارهای کاربردی که برای توسعهٔ برنامه‌ها مورد نیاز می‌باشد تشکیل شده است. معمولاً این ابزارها شامل *ویرایشگر* برای طراحی و تغییر برنامه‌ها، *مترجم* برای ترجمهٔ برنامه‌ها و برنامه *اشکال‌زدا*[2] می‌باشند. برنامه‌های اشکال‌زدا ابزارهایی را برای تحت کنترل قرار گرفتن اجرای برنامه‌ها در اختیار برنامه‌نویسان قرار می‌دهد این امر موجب می‌شود خطاهای احتمالی سریع‌تر و راحت‌تر کشف و اصلاح شوند.

برنامه‌های پایتون معمولاً توسط مفسر پایتون ترجمه می‌شوند؛ بنابراین امکان اجرای دستورات در یک محیط تعاملی فراهم می‌باشد. محیطی که این امکان را در اختیار برنامه‌نویس قرار می‌دهد *پوستهٔ تعامل پایتون*[3] نام دارد. تعامل با این پوسته بسیار شبیه به استفاده از ماشین حساب می‌باشد، البته در این محیط تعاملی شما دیگر تنها به عملیاتهای محاسبات (جمع، ضرب و...) محدود نیستید بلکه می‌توانید تمام دستورات پایتون را اجرا نمایید. نمونه‌ای از نحوهٔ استفاده از پوستهٔ تعاملی پایتون در شکل ۳-۲ آمده است. در این مثال عبارت ۹ * ۴ + ۲ در جلوی اعلان پوستهٔ تعاملی (<<<) وارد شده و بلافاصله نتیجهٔ آن یعنی ۳۸ نمایش داده شده است.

شکل ۳-۲ پوستهٔ تعاملی پایتون

1. Integrated Development Environment
2. Debugger
3. Python Interactive Shell

IDE

محیط توسعهٔ یکپارچه ترکیبی از ابزارها برای توسعهٔ نرم‌افزار می‌باشد.

شکل ۳-۳ نحوهٔ استفاده از ماژول

۳-۱-۲ کتابخانهٔ استاندارد پایتون

کتابخانهٔ استاندارد پایتون مجموعه‌ای از ماژولهای از پیش‌ساخته می‌باشد که هر کدام از آنها قابلیتهای خاصی را که فراتر از هستهٔ پایتون است، فراهم می‌نمایند (در مورد نحوهٔ ساخت ماژولها در فصل‌های آتی صحبت خواهیم کرد). برای مثال ماژول math توابع ریاضی تکمیلی را ارائه می‌دهد و یا ماژول random قابلیت تولید اعداد تصادفی را به برنامه‌ها اضافه می‌کند (با دیگر ماژولهای پایتون در طول این کتاب آشنا خواهید شد). برای استفاده از قابلیتهای ماژول مورد نظر در یک برنامهٔ خاص باید از دستور import همانند آنچه در شکل ۳-۳ آمده استفاده کرد.

مثالی که در شکل ۳-۳ آمده نشان می‌دهد که با استفاده از دستور `import math` می‌توان به توابع درون ماژول math از قبیل cos، sin و log10 دسترسی داشت. در بخشهای بعدی با نحوهٔ استفاده از ماژولها بیشتر آشنا خواهید شد.

کتابخانهٔ استاندارد پایتون مجموعه‌ای از ماژولها می‌باشد که هر کدام قابلیتهایی فراتر ازهستهٔ اصل پایتون را فراهم می‌نمایند.

۳-۱-۳ کمی درمورد پایتون

در این بخش درمورد پایتون به اندازه‌ای که بتوان برنامه‌های ساده‌ای نوشت صحبت خواهیم کرد. از آنجا که تمام برنامه‌های کامپیوتری اطلاعاتی را دریافت کرده، بر روی آنها پردازش انجام

داده و در نهایت نتیجه را نمایش می‌دهند؛ ما هم در این بخش با نحوهٔ نمایش متغیرها، نحوهٔ انجام محاسبات سادهٔ ریاضی و در نهایت انجام سادهٔ عملیات ورودی و خروجی صحبت خواهیم کرد.

متغیرها:

همانگونه که در فصل قبل بیان شد متغیرها یکی از مفاهیم بنیادی در برنامه‌های کامپیوتری می‌باشند. همانگونه که در مثال زیر می‌بینیم می‌توان گفت که متغیر نامی است که به یک مقدار اختصاص داده می‌شود.

```
n = ۵
```

برای مثال انتسابهای زیر را درنظر بگیرید:

```
pi = 3.14
radius = 11.2
area = pi × (radius * *2)
radius = 14.3
```

متغیرهای pi و radius به دو مقدار متفاوت اختصاص داده شده‌اند، سپس متغیرarea به مقداری که از روی مقادیر متغیرهای pi و radius ایجاد شده، مقید می‌شود؛ بعد از آن متغیرradius به مقداری دیگر منسوب شده. این انقیادها[1] در شکل ۳-۴ نشان داده شده است.

شکل ۳-۴ انقیاد متغیر به مقدار

اگر متغیری به مقداری منسوب شده باشد مثلاً x=۹، هرجا که متغیر x در محاسبات ظاهر شود مقداری که متغیر به آن منسوب شده بجای آن استفاده می‌شود. برای مثال اگر داشته باشیم۱۲ + x این به معنای ۱۲ + ۹ می‌باشد. یا در انتسابهای بالا area = pi × (radius **۲) به معنای area = ۳٫۱۴ × (۱۱٫۲ **۲) است.

1. Binding

متغیردرپایتون

درپایتون، متغیر تنها یک نام است که با یک مقدار در ارتباط می‌باشد.

برخی از عملگرهای ریاضی:

عملگرهای عمومی پایتون عبارتند از: + (جمع)، - (تفریق)، * (ضرب)، / (تقسیم) و** (توان). در پایتون علامت جمع، تفریق و تقسیم همان علامتهای استاندارد در ریاضیات می‌باشند، ۹-۴ ۱۲+۷ و ۲۱/۵. اما برای ضرب و توان از علامت ستاره (*) استفاده می‌کنیم.

$$۲^۳ \equiv ۳ ** ۲ \qquad ۱۰ \times ۵ \equiv ۱۰ * ۵$$

دقت شود که درپایتون از علامت// برای تقسیم صحیح استفاده می‌شود. یعنی:

$$۱۲/۵ = ۲/۵ \qquad ۱۲//۵ = ۲$$

ورودی و خروجی:

اکثر برنامه‌هایی که می‌نویسیم نیازمند دریافت اطلاعات از کاربر می‌باشند. در پایتون از تابع input برای انجام اینکار استفاده می‌شود:

```
>>> name = input('What is your name? ')
What is your name? Ali
>>> name
'Ali'
```

در این مثال تابع input با نمایش پیغامی برای کاربر اطلاعات ورودی را دریافت کرده و به متغیر name منسوب می‌کند. معادل این دستور در زبان روندنما بصورت زیر است:

از تابع print برای نمایش اطلاعات استفاده می‌شود.

```
>>> print('My name is', name)
My name is Ali
```

دقت نمایید که در تابع print نیز مانند جعبهٔ خروجی در روندنما از کاما برای جدا کردن اقلامی که قرار است چاپ شود استفاده می‌کنیم. این تابع بصورت پیش فرض یک فاصله بین

اقلامی که می‌خواهد چاپ شود قرار می‌دهد. پس خروجی دستور بالا `My name is Ali` است نه `My name isAli`.

در مورد ورودی/خروجی، متغیرها و عملگرها در فصل‌های آتی بیشتر صحبت خواهیم کرد.

۳-۱-۴ چگونه از PyCharm استفاده کنیم؟

ابتدا باید گفت که نسخهٔ آموزشی PyCharm برای کمک به برنامه‌نویسان کم تجربه و با هدف یادگیری سریع و کارآمد برنامه‌نویسی با ابزاری مدرن و حرفه‌ای طراحی شده است. نسخهٔ آموزشی PyCharm کاملاً رایگان با کد متن باز می‌باشد. برنامه‌نویسان مبتدی بدون پرداخت هزینه‌ای می‌توانند به آسانی آن را از اینترنت دریافت کنند.

بیشتر ویرایشگرهای برنامه، محیط‌های توسعهٔ یک‌پارچه و دیگر ابزارهای برنامه‌نویسی برای کسانی که تنها می‌خواهند برنامه‌نویسی یاد بگیرند بسیار پیچیده می‌باشند. این ابزارها برنامه‌نویس را مجبور می‌کنند تا وقت زیادی را صرف یادگیری نحوهٔ عملکرد ابزار نماید. اما PyCharm درعین سادگی به اندازه‌ای قدرتمند است که شما را به یک توسعه دهندهٔ حرفه‌ای تبدیل نماید.

برای آشنایی بیشتر شما با نحوهٔ نوشتن برنامه‌های پایتون با استفاده از PyCharm، برنامهٔ ساده‌ای می‌نویسیم که نام کاربر را گرفته و پیغام خوش آمدگویی را برای آن نمایش می‌دهد. هدف اصلی ما از این مثال آشنایی با مفهوم نحوهٔ ساختن و اجرای برنامه‌های پایتون و آشنایی بیشتر با توابع `print` و `input` می‌باشد.

برای نوشتن برنامه در PyCharm ابتدا باید یک پروژه بسازیم. برای اینکار گزینهٔ Create New Project را از صفحهٔ Quick Start انتخاب کنید (شکل ۳-۵). از صفحهٔ جدید گزینهٔ Pure Python را انتخاب کرده و در پنجرهٔ جدید ابتدا نام HelloWorld را برای پروژه انتخاب و سپس نوع مفسر پایتون را تعیین کنید. بعد از فشردن دکمهٔ Create پروژه‌ای خالی به نام HelloWorld ایجاد می‌شود (شکل ۳-۶).

شکل ۳-۵ ساختن فایل برنامهٔ پایتون

شکل ۳-۶ شکل برنامه بعد از ساختن پروژهٔ جدید

شما یک پروژهٔ خالی ایجاد کرده‌اید که هیچ فایلی در آن وجود ندارد. برای اضافه کردن یک فایل جدید با کلیک راست بر روی نام پروژه که در سمت چپ برنامه قرار دارد و انتخاب گزینهٔ New → Python file می‌توانید نام فایل جدید را وارد کنید. برای مثال نام main رابنویسید. حال یک فایل خالی ایجاد شده که می‌توانید برنامه زیر را در آن بنویسید:

سپس گزینهٔ Save را از منوی File انتخاب کرده و تغییرات فایل را ذخیره نمایید. دقت نمایید که پسوند فایلهای پایتون ”.py“ است.

برای اجرای برنامه گزینهٔ Run ’main.py’ را از منوی Run انتخاب کنید و یا دکمهٔ Shift+F10 را فشار دهید. اگر برنامه را درست نوشته باشید خروج را در قسمت پایین نرم‌افزار مانند شکل ۳-۷ مشاهده خواهید کرد.

شکل ۳-۷ خروجی حاصل از اجرای برنامهٔ helloWorld.py

اگر برنامه را اشتباه نوشته باشید مثلاً در نوشتن نام تابع print غلط املایی داشته باشید با پیغام خطا مواجه می‌شوید (شکل ۳-۸).

شکل ۳-۸ خطا در اجرای برنامهٔ helloWorld.py

۳-۱-۵ برنامه: تبدیل گالن بنزین

قسمتهای اصلی اولین برنامه‌ای که می‌نویسیم شامل: نمایش پیغام، دریافت اطلاعات، پردازش اطلاعات ورودی و در نهایت نمایش نتایج بصورت مطلوب می‌باشد. در ابتدا برنامه پیغامی مبنی بر ورود یک عدد حقیقی که میزان بنزین در واحد گالن است را نمایش می‌دهد. سپس برنامه اطلاعات زیر را از آن استخراج کرده و بصورت مطلوب نمایش خواهد داد:

۱. مقدار بنزین در واحد لیتر،
۲. تعداد بشکه‌های نفت خامی که برای تولید این مقدار بنزین لازم است،
۳. میزان تولیدی گاز CO_2 توسط این مقدار بنزین در واحد پوند،
٤. مقدار اتانولی که انرژی آن معادل انرژی همان مقدار بنزین است،
۵. قیمت آن به دلار.

تبدیلات زیر را می‌توانید به راحتی از تارنمای جهانی مخصوصاً دانشنامهٔ آزاد[1] بدست آورید:

۱. هر گالن معادل ۳٫۷۸۵۴ لیتر می‌باشد.
۲. بطور تقریبی هر بشکه نفت خام ۱۹٫۵ گالن بنزین تولید می کند. هر بشکه نفت تقریباً ۲۴ گالن است.
۳. هر گالن بنزین تقریباً ۲۰ پوند CO_2 تولید می‌کند.
٤. هر گالن بنزین تقریباً ۱۱۵۰۰۰ BTU[2] انرژی تولید می‌کند. این درحال است که هر گالن اتانول ۷۵۷۰۰ BTU انرژی دارد.
۵. در حال حاضر قیمت هر بشکه نفت خام ۵۰ دلار است.

برنامهٔ ۳-۱ پیاده‌سازی راه‌حل مسئله است. برای آشنایی با نحوهٔ عملکرد برنامه به نمونه‌هایی از خروجی برنامه که در ادامه آمده است به دقت توجه نمایید.

1. Wikipedia
2. British Thermal Units

کد ۱-۳: برنامهٔ محاسبهٔ مقدار بنزین

```
1  #Converting gasoline measures
2  gallons = float(input('Please enter the number of gallons of '+\
3                        'gasoline: '))
4  print('Original number of gallons is:', gallons)
5  #Calculate the equivalent liters
6  liters = gallons * 3.7854
7  #Calculate the amount of crude oil required
8  barrels_oil = gallons / 19.5
9  #The amount of CO2 produced
10 co2 = gallons * 20
11 #Calculate the equivalent energy
12 gasoline_energy = gallons * 115000
13 ethanol_gallons = gasoline_energy / 75700
14 #Calculate the price of gasoline
15 price = barrels_oil * 50
16 #Print desired outputs
17 print(gallons, 'gallons is the equivalent of', liters,'liters')
18 print(gallons, 'gallons of gasoline requires', barrels_oil, \
19      'barrels of oil')
20 print(gallons,'gallons of gasoline produce',co2,'pounds of CO2')
21 print(gallons, 'gallons of gasoline is energy equivalent to', \
22      ethanol_gallons, 'gallons of ethanol')
23 print(gallons,'gallons of gasoline requires',price,'US dollars')
24 print('Thank you for attentions.')
```

```
>>>
Please enter the number of gallons of gasoline: 100
Original number of gallons is: 100.0
100.0 gallons is the equivalent of 378.54 liters
100.0 gallons of gasoline requires 5.128205128205129 barrels of oil
100.0 gallons of gasoline produce 2000.0 pounds of CO2
100.0 gallons of gasoline is energy equivalent to 151.91545574636723
                                    gallons of ethanol
100.0 gallons of gasoline requires 256.4102564102564 US dollars
Thank you for attentions.
>>>
Please enter the number of gallons of gasoline: 390000000
Original number of gallons is: 390000000.0
390000000.0 gallons is the equivalent of 1476306000.0 liters
390000000.0 gallons of gasoline requires 20000000.0 barrels of oil
390000000.0 gallons of gasoline produce 7800000000.0 pounds of CO2
390000000.0 gallons of gasoline is energy equivalent to
                            592470277.4108323 gallons of ethanol
390000000.0 gallons of gasoline requires 1000000000.0 US dollars
Thank you for attentions.
>>>
```

چند نکتهٔ مهم در این برنامه وجود دارد:

۱. درخطوط ۲، ۱۸ و ۲۱ برای شکستن خط از علامت «\» استفاده شده است. در پایتون می‌توان از این سبک نوشتاری برای شکستن یک خط و نوشتن آن در خطوط متوالی استفاده کرد.

۲. تابع input مقدار ورودی کاربر را بصورت یک مقدار رشته‌ای وارد برنامه می‌کند، لذا باید قبل از استفاده آنرا به نوع مورد نظر تبدیل نماییم. تابع float در خط ۲ مقدار ورودی را به نوع اعشاری تبدیل می‌نماید. اگر بخواهیم مقدار ورودی به نوع صحیح تبدیل شود باید از تابع int استفاده کرد.

همانطور که در برنامه مشاهده می‌نمایید خطوط ۱ الی ۴ مقدار بنزین در واحد گالن را از کاربرگرفته و در خطوط ۶ الی ۱۵ محاسبات مورد نظر انجام می‌شود. خطوط ۱۷ الی ۲۴ نتایج را بطور مطلوبی نمایش می‌دهند.

همانطور که قبلاً اشاره شد خطوط ۱، ۵، ۷، ۹، ۱۱، ۱۴ و ۱۶ توضیحات می‌باشند که تاثیری بر روی اجرای برنامه ندارند؛ فقط خوانایی برنامه را بالا می‌برند.

خلاصهٔ بخش

- زبان برنامه نویسی پایتون / گایدو وَن‌رُسُوم (طراح زبان پایتون)
- توضیحات در پایتون
- مقدمه‌ای بر متغیرها، عملگرهای ریاضی و ورودی/خروجی در پایتون
- آشنایی با ثابتهای رشته‌ای در تابع input
- آشنایی با کتابخانهٔ استاندارد پایتون و دستور import
- آشنایی با پوستهٔ تعاملی پایتون و محیط توسعهٔ یکپارچهٔ PyCharm

۲-۳ داده ها و عبارتها

تولید، جمع آوری و تجزیه و تحلیل داده‌ها نیروی محرکهٔ دنیای امروز می‌باشد. داده‌های زیادی بطور متناوب در حال تولید شدن می‌باشند. فروشگاههای زنجیره‌ای سالیانه یک ترابایت اطلاعات خرید مشتریان را برای تعیین الگوی خرید آنها تولید و ارزیابی می‌کنند. کاربران فیس‌بوک تاکنون نزدیک به ۴۰ میلیارد عکس که برای ذخیرهٔ آنها یک *پتابایت* حافظه نیاز می‌باشد، ایجاد کرده‌اند. پیش بینی می‌شود رادیوتلسکوپ‌های خاص هر چهار ساعت یک *هگزابایت* اطلاعات تولید کنند. طبق گفته‌ها، مقدار دادهٔ تولید شده در هر سال حدود دو *زِتابایت*

تخمین زده می‌شود که این مقدار هر از دو سال بیش از دو برابر می‌شود. برای آشنایی بیشتر با واحدهای انداز ه‌گیری داده‌ها جدول ۳-۱ را ببینید.

جدول ۳-۱ معیارهای سنجش اندازهٔ داده‌ها

عنوان		اندازه (بایت)
Kilobyte(KB)	$۱۰^{۳}$	۱, ۰۰۰
Kibibyte[۱]	$۲^{۱۰}$	۱, ۰۲۴
Megabyte(MB)	$۱۰^{۶}$	۱, ۰۰۰, ۰۰۰
Megibyte	$۲^{۲۰}$	۱, ۰۴۸, ۵۷۶
Gigabyte(GB)	$۱۰^{۹}$	۱, ۰۰۰, ۰۰۰, ۰۰۰
Gibibyte	$۲^{۳۰}$	۱, ۰۷۳, ۷۴۱, ۸۲۴
Terabyte(TB)	$۱۰^{۱۲}$	۱, ۰۰۰, ۰۰۰, ۰۰۰, ۰۰۰
Tibibyte	$۲^{۴۰}$	۱, ۰۹۹, ۵۱۱, ۶۲۷, ۷۷۶
Petabyte(PB)	$۱۰^{۱۵}$	۱, ۰۰۰, ۰۰۰, ۰۰۰, ۰۰۰, ۰۰۰
Pebibyte	$۲^{۵۰}$	۱, ۱۲۵, ۸۹۹, ۹۰۶, ۸۴۲, ۶۲۴
Exabyte(EB)	$۱۰^{۱۸}$	۱, ۰۰۰, ۰۰۰, ۰۰۰, ۰۰۰, ۰۰۰, ۰۰۰
Exbibyte	$۲^{۶۰}$	۱, ۱۵۲, ۹۲۱, ۵۰۴, ۶۰۶, ۸۴۶, ۹۷۶
Zettabyte(ZB)	$۱۰^{۲۱}$	۱, ۰۰۰, ۰۰۰, ۰۰۰, ۰۰۰, ۰۰۰, ۰۰۰, ۰۰۰
Zebibyte	$۲^{۷۰}$	۱, ۱۸۰, ۵۹۱, ۶۲۰, ۷۱۷, ۴۱۱, ۳۰۳, ۴۲۴

[۱]سازمان استاندارد بین‌المللی برای جلوگیری از اشتباه توصیه کرده کیلوبایت، مگابایت و ... را تنها برای $۱۰^{۶}$, $۱۰^{۳}$و... بکار برده شود.

در این بخش نحوهٔ نمایش و انجام عملیات بر روی داده‌ها در پایتون بررسی خواهد شد.

۳-۲-۱ ثابتها

به دنباله‌ای از یک یا بیشتر نویسه[1] که ترکیب آنها خود معنایی را القا می‌کند ثابت[2] گفته می‌شود. برای مثال ۱۲، ۱۳٫۸۷و "علی" جزء ثابتها می‌باشند. بطور کلی ثابتها به دو دستهٔ ثابتهای عددی و ثابتهای رشته‌ای تقسیم شده‌اند.

۳-۲-۱-۱ ثابتهای عددی

ثابتهای عددی، ثابتهایی هستند که تنها از نویسه‌های عددی (۹ - ۰)، علامت عدد که بصورت اختیاری استفاده می‌شود (+ یا -) و درصورت لزوم علامت اعشار تشکیل شده‌اند. البته در نشانه‌گذاری علمی از علامت «e» نیز استفاده می‌شود. اگر در ثابت عددی از علامت اعشار استفاده شود به آن عدد ممیزشناور و یا بطور خلاصه عدد اعشاری گفته می‌شود (مانند ۱۲٫۵۶)؛ در غیر اینصورت آن را عدد صحیح می‌نامند. دقت نمایید که هیچگاه در ثابتهای عددی از کاما استفاده نمی‌شود. در جدول ۳-۲ نمونه‌هایی از ثابتهای عددی که در پایتون استفاده می‌شود آمده است.

جدول ۳-۲ مثالهایی برای ثابتهای عددی

ثابت‌های عددی					
مقادیر صحیح	مقادیر اعشاری			اعداد غلط	
۵	۵۰۰۰٫۱۲۵	۵٫۱۲۵	۰٫۵ یا ٫۵	۴۰٫۵۶٫۷۷	۵,۰۰۰٫۱۲۵
۲۵۰۰	۲۵۰۰٫۱۲۵	۲۵۰۰٫	۲۵۰۰٫۰	۲,۵۰۰٫۱۲۵	۲,۵۰۰
+۲۵۰۰	+۲۵۰۰٫۱۲۵	+۲۵۰۰٫	+۲۵۰۰٫۰	+۲,۵۰۰٫۱۲۵	+۲,۵۰۰
-۲۵۰۰	-۲۵۰۰٫۱۲۵	-۲۵۰۰٫	-۲۵۰۰٫۰	-۲,۵۰۰٫۱۲۵	-۲,۵۰۰

اعدادی که در آنها علامت استفاده نشده بصورت پیش فرض مثبت در نظر می‌گیریم.

محدودهٔ اعداد ممیز شناور:

در زبان برنامه‌نویسی پایتون محدودیتی برای اندازهٔ اعداد صحیح وجود ندارد[3] اما همانگونه که قبلاً بیان شد اندازهٔ نمایش اعداد اعشاری محدود می‌باشد. اعداد اعشاری در پایتون هم از نظر بازهٔ عددی محدود هستند هم از لحاظ دقت. در این زبان از استاندارد اعداد اعشاری بزرگ (IEEE

1. Character
2. Literal
۳. البته این بدان معنی نیست که می‌توان عدد صحیح به هر اندازه داشت بلکه پایتون مکانیزمی در نظر گرفته که می‌توان اعداد بزرگ تعریف کرد.

آزمایش کنید!

دستورات زیر را در محیط تعاملی پایتون اجرا کرده و نتایج آنرا به دقت بررسی نمایید:

```
>>> 1024          >>> -1024          >>> .1024
???               ???                ???
>>> 1,024         >>> 0.1024         >>> 1,024.46
???               ???                ???
```

754) با محدودهٔ اعداد از $۱۰^{-۳۰۸}$ تا $۱۰^{۳۰۸}$ و ۱۶ یا ۱۷ رقم دقت استفاده می‌شود. برای نمایش محدودهٔ مقادیر، در پایتون نیز می‌توان از نمایش علمی استفاده کرد:

9.0045602e+5 معادل است با 9.0045602×10^{5} با هشت رقم دقت و عدد بزرگی مانند 1.006249505236801e8 معادل است با $1.006249505236801\times10^{8}$ با ۱۶ رقم دقت و 4.239e-16 معادل است با 4.239×10^{-16} با ۴ رقم دقت.

توجه به محدودیتهای اعداد اعشاری بسیار مهم است. برای مثال ممکن است ضرب دو عدد اعشاری در کامپیوتر باعث بروز سرریزریاضی[1] شود. سرریزریاضی زمان رخ می‌دهد که نتیجه محاسبه شده از لحاظ اندازه آنقدر بزرگ باشد که نتوان آنرا نمایش داد. مثلاً:

```
>>> 1.5e200 * 2.0e210
inf
>>>
```

نتیجهٔ این عبارت مقدار `inf` به معنای بینهایت است که با نتیجه درست یعنی 3.0e410 متفاوت است؛ علت کاملاً مشخص است، نتیجهٔ واقعی آنقدر بزرگ است که قابل نمایش نیست یعنی سرریزریاضی رخ داده است. بطور مشابه تقسیم دو عدد اعشاری ممکن است باعث پاریزریاضی[2] شود. این اتفاق زمانی رخ می‌دهد که نتیجه محاسبه آنقدر کوچک باشد که نتوان آنرا نشان داد. مثلاً:

```
>>> 1.0e-300 / 1.0e100
0.0
>>>
```

نتیجه 0.0 می‌شود که با 1.0e-400 متفاوت است.

1 Arithmetic Overflow
2 Arithmetic Underflow

آزمایش کنید!

دستورات زیر را در محیط تعاملی پایتون اجرا کرده و نتایج آنرا به دقت بررسی نمایید:

```
>>> 1.2e200*2.4e100      >>> 1.2e20/2.4e100
???                      ???
>>> 1.2e200*2.4e200      >>> 1.2e-200/2.4e200
???                      ???
```

سرریز و پاریز ریاضی را به راحتی می‌توان تشخیص داد. نکتهٔ مهم و قابل توجه از دست دادن اطلاعات مربوط به رقمهای اعشاری (دقت عدد اعشاری) است. برای مثال در ریاضیات نتیجهٔ تقسیم ۷ بر ۳ برابر است با ... ۲٫۳۳۳۳۳ یعنی بینهایت تکرار ۳. اما می‌دانیم که برای نمایش اعداد اعشاری، عدد باید تعداد محدودی رقم اعشاری داشته باشد. پس تنها راه این است که تقریب مناسبی از عدد را بجای خود عدد نگهداری کنیم. مثلاً در پایتون نتیجه تقسیم اخیر بصورت ۲٫۳۳۳۳۳۳۳۳۳۳۳۳۳۳۳۵ ذخیره می‌شود.

```
>>> 7/3
2.3333333333333335
>>>
```

همانگونه که می‌بینید پایتون بعد از ۱۶ تکرار نتیجه را متوقف می‌نماید. انتظار داریم حاصلضرب ۳ در $\frac{۷}{۳}$ برابر با ۶٫۹۹۹۹۹۹۹۹۹۹۹۹۹۹۹۹ شود اما پوستهٔ تعاملی پایتون مقدار ۷٫۰ را نمایش می‌دهد، چرا؟

```
>>> (7/3)*3
7.0
>>>
```

علت آن است که پایتون برای اینکه بتواند بر روی تعداد ارقامی که می‌خواهد نمایش دهد کنترل داشته باشد نتیجه را بصورت گرد شده نمایش می‌دهد. حاصل تقسیم ۱/۳ برابر با ۰٫۳۳۳۳۳۳۳۳۳۳۳۳۳۳۳۳ می‌باشد با توجه به آن به مثال زیر توجه کنید:

```
>>> 1/3+1/3+1/3+1/3+1/3+1/3
1.9999999999999998
>>> 6*(1/3)
2.0
>>>
```

نتیجهٔ محاسبهٔ اول وابسته به نمایش مقدار گرد شدهٔ ۰٫۳۳۳۳۳۳۳۳۳۳۳۳۳۳۳۳ می‌باشد، لذا درانتهای نتیجه بجای ۹ مقدار ۸ قرار گرفته است. اما در محاسبهٔ ضرب نتیجه به مقدار ۲٫۰ گرد

شده است. به هر حال مهم نیست که پایتون از چه روشی برای نمایش نتایج استفاده می‌کند، مهم آن است که در نظر داشته باشیم که مقادیر هم از نظر محدودهٔ قابل نمایش و هم از لحاظ دقت محدود می‌باشند. در برنامه‌های کاربردی که روزانه از آن استفاده می‌کنیم، از دست دادن کمی دقت مشکل‌ساز نیست؛ اما برای محاسبات دقیق مهندسی باید این تغییرات را مدنظر داشته باشیم.

آزمایش کنید!

دستورات زیر را در محیط تعاملی پایتون اجرا کرده و نتایج آنرا به دقت بررسی نمایید:

```
>>> 1/10                    >>> 6*(1/10)
???                         ???
>>> 1/10+1/10+1/10          >>> 6*1/10
???                         ???
>>> 10*(1/10)               >>> 10*1/10
???                         ???
```

برای کنترل تعداد ارقام اعشاری قابل نمایش، پایتون تابع `format` را ارائه کرده است (توابع را در فصل پنجم خواهید خواند). این تابع دو مقدار را به عنوان پارامتر می‌گیرد مقدار اول عدد اعشاری مورد نظر است و مقدار دوم فرمت نهایی عدد اعشاری است. دقت نمایید که تابع `format` نسخهٔ رشته‌ای نتیجه را بر می‌گرداند. به مثالهای زیر توجه نمایید:

```
>>> 18/5
3.6
>>> format(18/5, '.3f')
'3.600'
>>> 15/21
0.7142857142857143
>>> format(15/21, '.4f')
'0.7143'
>>>
```

در مثالهای بالا ’.3f’ و ’.4f’ قالب خروجی می‌باشند که با استفاده از آن تابع، نتیجه را با ۳ یا ۴ رقم اعشار دقت گرد کرده بصورت رشته باز می‌گرداند. برای اعداد بسیار بزرگ یا بسیار کوچک می‌توان از نماد علمی استفاده کرد:

```
>>> 2**100
1267650600228229401496703205376
>>> format(2**100, '.6e')
```

```
'1.267651e+30'
>>>
```

در این مثال عدد با نماد علمی و شش رقم دقت قالب بندی شده است.

در قالب بندی اعداد، برای تقسیم بندی سه تایی اعداد بزرگ می‌توان از کاما (,) نیز استفاده کرد.

```
>>> format(123456789.4567, ',.2f')
'123,456,789.46'
>>>
```

آزمایش کنید!

دستورات زیر را در محیط تعاملی پایتون اجرا کرده و نتایج آنرا به دقت بررسی نمایید:

```
>>> format(11/12, '.2f')      >>> format(11/12, '.2e')
???                           ???
>>> format(11/12, '.3f')      >>> format(11.12, '.3e')
???                           ???
```

۳-۲-۱-۲ ثابتهای رشته‌ای

ثابتهای رشته‌ای مجموعه‌ای از نویسه‌های الفبایی می‌باشند، مثلاً 'Hello, world!'، "Mohsen Rahmanian"، "I'm 32 years old.". در پایتون ثابتهای رشته‌ای یا مختصراً رشته‌ها بین دو علامت نقل قول تک (') یا نقل قول دوتایی (") محصور می‌شوند. رشته‌ها باید در یک خط باشند (به استثنای زمانی که از ممیز وارون استفاده می‌کنیم). قبلاً از رشته‌ها برای نمایش خروجی استفاده کرده‌ایم:

```
>>> print('Welcome to python!')
Welcome to Python!
>>>
```

هر رشته شامل صفر یا بیشتر نویسهٔ الفبا، اعداد، نویسه‌های خاص و فضای خالی می‌باشد. رشته‌هایی با طول صفر (یعنی یک جفت نقل قول تنها) به رشتهٔ *تهی* معروف است. این رشته، با رشته‌ای که تنها شامل یک فضای خالی است متفاوت است. رشته‌ها می‌توانند شامل نقل قول تکی باشند؛ دراینصورت باید رشته را با نقل قول دوتایی محصور کرد. اینکار برای درج علامت نقل قول دوتایی در رشته نیز بصورت عکس انجام می‌شود.

```
"Mohsen's son", 'Ali says:"Hello ..." '
```

در پایتون نسخهٔ ۳ برنامه‌ها از کدگذاری یون کد۸ (UTF-8) استفاده می‌کنند لذا شما می‌توانید به راحتی نویسه‌های یونی‌کد را به برنامه اضافه نمایید. برای مثال در UTF-8 نویسهٔ 'A' به

(۶۵)۰۱۰۰۰۰۰۱ و 'B' به (۶۶)۰۱۰۰۰۰۱۰ کد می‌شود. این توالی برای نویسه‌های عددی نیز صادق است؛ '۰' به (۴۸)۰۰۱۱۰۰۰۰ و '۱' به (۴۹)۰۰۱۱۰۰۰۱ کد می‌شوند.

بسیار مهم است که بین مقادیر عددی و نمایش رشته‌ای آنها تفاوت قائل شویم. نمایش عددی ۱۳۷ با رشتهٔ '۱۳۷' متفاوت است (جدول ۳ . ۳ را ببینید).

جدول ۳-۳ تفاوت نمایش عددی و رشته ای مقادیر عددی

مقدار عددی	مقدار رشته ای
$\underbrace{10001011}_{137}$	$\underbrace{'00110001'}_{'1'}\underbrace{'00110011'}_{'3'}\underbrace{'00110111'}_{'7'}$

در پایتون برای تعیین کد یک نویسه می‌توان از تابع `ord` استفاده کرد. این تابع نویسه‌ای را به عنوان آرگومان گرفته کد آنرا بصورت یک عدد ده‌دهی باز می‌گرداند. مثلاً `ord('1')` برابر با ۴۹ و `ord('A')` برابر با ۶۵ می‌باشد. اگر بخواهیم نویسهٔ مربوط به یک کد را بدانیم از تابع `chr` استفاده می‌نماییم. مثلاً `chr(67)` برابر با `'C'` است.

۳-۲-۱-۳ نویسه‌های کنترل

نویسه‌های کنترلی، نویسه‌های خاصی می‌باشند که در خروجی نمایش داده نمی‌شوند بلکه وظیفهٔ آنها کنترل نمایش اطلاعات در خروجی است. این نویسه‌ها بر روی صفحه کلید وجود ندارند بلکه از ترکیب چند نویسه تشکیل می‌شوند. به ترکیب این نویسه‌ها *دنبالهٔ کنترلی* گفته می‌شود. هر دنبالهٔ کنترلي با نویسهٔ *گریز*[1] شروع می‌شود؛ بدان معنی که معنای عمومی نویسهٔ بعدی تغییر کرده است. در پایتون نویسهٔ ممیز وارون (\) این نقش را بازی می‌کند. برای مثال دنبالهٔ کنترلی `'\n'` نشان دهندهٔ نویسهٔ خط جدید است و برای درج ادامهٔ خروجی در خط جدید استفاده می‌شود. به مثال زیر توجه کنید:

```
>>> print("Hello Mohsen.\nI'm Ali.")
Hello Mohsen.
I'm Ali.
>>>
```

1. Escape Character

جدول ۳-۴ نویسه‌های کنترلی

عملکرد	نماد نویسه	کد مبنای ۱۶
صدای زنگ	\a	0x07
نویشه برگشت به عقب (backspace)	\b	0x08
انصراف یا Ctrl+x	\cx , \C-x	0x18
گریز یا Escape	\e	0x1b
صفحه جدید در پرینتر	\f	0x0c
خط جدید	\n	0x0a
نمایش مبنای ۸، درحالی که nها باید بین ۷-۰ باشند	\nnn	
انتقال به ابتدای خط	\r	0x0d
فضای خالی	\s	0x20
پرش افقی	\t	0x09
پرش عمودی در پرینتر	\v	0x0b
نمایش مبنای ۱۶، درحالی که nها باید بین ۹-۰ و یا a-f باشند	\xnn	

درجدول۳-۴ تمام نویسه‌های کنترلی که توسط پایتون پشتیبانی می‌شوند، مشخص شده‌اند.

۳-۲-۱-۴ قالب‌بندی رشته‌ها

در بخش قبل از تابع format برای کنترل نحوهٔ نمایش اعداد در خروجی استفاده کردیم. می‌توان از این تابع برای قالب‌بندی رشته‌ها نیز استفاده کرد. شکل کلی تابع format بصورت زیر است:

```
format(value, format_specifier)
```

که در آن value مقداری است که باید نمایش داده شود و format_specifier شامل ترکیبی از گزینه‌های قالب‌بندی می‌باشد. برای مثال اگر بخواهیم رشتهٔ 'Alireza' در یک حوزهٔ ۲۰ نویسه‌ای از چپ مرتب شده باشد، تابع format را بصورت زیر فراخوانی می‌کنیم:

```
format('Alireza', '<20') ⇒ 'Alireza             '
```

برای راست‌چین کردن از علامت «>» و وسط‌چین کردن از علامت «^» استفاده می‌شود.

```
format('Alireza', '^20') ⇒ '       Alireza       '
format('Alireza', '>20') ⇒ '             Alireza'
```

بصورت پیش فرض فضای خالی برای پرکردن کمبود نویسه استفاده می‌شود. می‌توان نویسهٔ دلخواه را جایگزین آن قرار داد:

```
>>> format(' His name ', '*^30')
'********** His name **********'
>>>
```

آزمایش کنید!

دستورات زیر را در محیط تعاملی پایتون اجرا کرده و نتایج آنرا به دقت بررسی نمایید:

```
>>> print('Hello, world!')
???
>>> print("Hello, world!')
???
>>> ord('f')
???
>>> chr(98)
???
>>> print('abc\bdef')
???
>>> print('abc\rdef')
???
>>> str1 = format('-','-<20')
>>> str2 = format('-','->20'))
>>> print(str1, 'Hello World', str2)
???
```

نکات مهم:

- ثابت حرفی دنباله‌ای از یک یا بیشتر نویسه می‌باشد.

- ثابت عددی ثابتی است که تنها از نویسه‌های ۹-۰، علامت عدد (+ یا -) و احتمالاً علامت اعشار (/) تشکیل شده است.

- سرریز زمانی رخ می‌دهد که نتیجه محاسبات برای نمایش بسیار بزرگ باشد.

- پاریز زمانی رخ می‌دهد که نتیجه محاسبات برای نمایش بسیار کوچک باشد.

- از تابع `format` برای قالب‌بندی اعداد و ثابت‌های رشته‌ای استفاده می‌شود.

۳-۲-۲ متغیرها و شناسه‌ها

همانگونه که قبلاً تبیین شد، متغیر نامی است (یک شناسه) که با مقداری در ارتباط است. در طول اجرای یک برنامه، متغیر می‌تواند مقادیر متفاوتی را بپذیرد؛ به همین علت است که آنرا «متغیر» می‌نامند. می‌توان به متغیرها توسط عملگر انتساب «=» مقداری منسوب کرد:

```
num = 10          num = num + 1
```

اغلب برنامه‌نویسان تازه‌کار برداشتی اشتباه از دستور انتساب دارند. در ریاضیات عبارت `num = num + 1` معنایی ندارد. امّا در دنیای برنامه‌نویسی این عبارت به معنای اضافه کردن یک به مقدار متغیر `num` می‌باشد. برای جلوگیری از اشتباه بهتر است که علامت «=» را مانند یک علامت «←» در نظر بگیریم. یعنی `a = a + 1` را بصورت `a ← a + 1` تفسیر کنیم. در شکل جدید تفسیر انتساب کاملاً مشخص است که در اجرای دستور سمت راست، عبارت، ابتدا ارزیابی می‌شود سپس نتیجهٔ آن به متغیر سمت چپ منسوب می‌گردد. اما چون متاسفانه بر روی صفحه کلیدهای استاندارد کلیدی معادل علامت «←» وجود ندارد نمی‌توان از آن استفاده کرد. ولی معنای آن برای درک بهتر عمل انتساب کاملاً مفید است.

باز تاکید می‌شود در پایتون متغیرها به مقادیر منسوب می‌شوند. برای روشن شدن مطلب فرض کنید دو متغیر`num` و `count` با عدد ۲۵ مقداردهی شده‌اند. نتیجهٔ انتساب‌های `num = 25` و `count = 25` مانند شکل ۳-۹ می‌باشد. یعنی هر دو متغیر به یک خانه از حافظه اشاره دارند.

شکل ۳-۹ نتیجه دو انتساب به مقداری برابر

تابع `id` در پایتون شناسهٔ آرگومان ورودی را بر می‌گرداند یعنی آدرس حافظهٔ محلی که آرگومان در آن ذخیره شده است. با دقت به نتیجهٔ اعمال این تابع بر روی دو متغیر `num` و`count` توجه کنید:

```
>>> id(num)
10455808
>>> id(count)
10455808
>>>
```

همانگونه که می‌بینید با وجود اینکه دو متغیر مجزا داریم اما نتیجه یکسان است. مثال زیر را ببینید:

```
>>> num = 13
>>> id(num)
10455424
>>> num = num + 1
>>> id(num)
10455456
>>>
```

انتظار ما این است که متغیرها از هم مجزا باشند، اما اشتراک مقادیر ایجاد مشکل می‌کند. اینجا سوالی مطرح است، آیا با تغییر مقدار num مقدار count نیز تغییر خواهد کرد؟ در این مورد جواب منفی است زیرا هر دو متغیر به مقدار عددی ۲۵ منسوب شده‌اند و در پایتون مقادیر عددی تغییر ناپذیر[1] می‌باشند. به مثال زیر توجه کنید:

در این مثال چون مقدار num عوض می‌شود پس شناسهٔ آن نیز تغییر می‌کند. اما شناسهٔ متغیرcount عوض نخواهد شد. تنها زمان که یک متغیر دوباره منسوب شود شناسهٔ آن تغییر خواهدکرد.

اگر متغیر دیگری به حافظهٔ مقدار اصلی منسوب نشده باشد، حافظه برای استفاده مجدد آزاد می‌گردد. در پایتون می‌توان در طول اجرای برنامه یک متغیر را به مقادیری از نوع‌های متفاوت منسوب کرد.

```
var = 97        → integer
var = 3.1415    → float
var = 'Alireza' → string
```

تنها راه انتساب مقدار به متغیر دستورات بالا نیست. همانگونه که قبلاً گفته شد می‌توان با استفاده از تابع input از مقادیری که توسط کاربر آماده می‌شود نیز استفاده کرد.

```
>>> name = input('What is your name?')
What is your name? Mohsen
>>> name
'Mohsen'
>>>
```

1. immutable

در این مثال مقدار ثابت رشته ای 'Mohsen' به متغیر name منسوب می‌شود. اگر کاربر دکمه Enter را بدون وارد کردن حرفی فشار دهد رشته '' (تهی) به متغیر name نسبت داده می‌شود.

آزمایش کنید!

دستورات زیر را در محیط تعاملی پایتون اجرا کرده و نتایج آنرا به دقت بررسی نمایید:

```
>>> input('Enter a number:')
Enter a number: 12
???
>>> int(input('Enter a number:'))
Enter a number: 12
???
>>> input('Enter a name:')
Enter a name: Alireza
???
>>> int(input('Enter a name:'))
Enter a name: Alireza
???
```

دقت نمایید که تمام مقادیری که تابع input بر می‌گرداند از نوع رشته‌ای است. درصورتی که نیاز به مقادیر عددی باشد باید آنرا به نوع مورد نظر تبدیل کرد. همانگونه که قبلاً گفته شد برای اینکار در پایتون دو تابع تبدیل نوع int() و float() تهیه شده است.

```
>>> std_number = int(input('What is your std. number?'))
What is your std. number? 1234
>>> grd = input('What is your grade?')
What is your grade? 15.6
>>> grade = float(grd)
>>>
```

در این مثالها مقدار رشته ای '1234' از کاربر دریافت شده سپس توسط تابع int به مقدار عددی 1234 تبدیل شده و به متغیر std_number منسوب می‌گردد.

۳-۲-۲-۱ کلمات کلیدی و شناسه های معروف پایتون

کلمهٔ کلیدی[1] به شناسه‌ای گفته می‌شود که در زبان برنامه‌نویسی معنای خاصی می‌دهد. بنابراین نمی‌توان از کلمات کلیدی به عنوان شناسه استفاده کرد. اگر از کلمات کلیدی به عنوان نام متغیر استفاده کنیم با پیغام خطای املایی (Syntax error) مواجه خواهیم شد.

```
>>> break = 4
 File "<stdin>", line 1
   break = 4
      ^
 SyntaxError: invalid syntax
>>>
```

در جدول ۳-۵ کلمات کلیدی پایتون مشخص شده‌اند.

جدول ۳-۵ کلمات کلیدی پایتون

False	break	else	if	not	while
None	class	expect	import	or	with
True	continue	finally	in	pass	yeild
and	def	for	is	raise	
as	del	from	lambda	return	
assert	elif	global	nonlocal	try	

برای دیدن کلمات کلیدی پایتون در پوستهٔ تعامل تابع help() را فراخوان کرده سپس کلمهٔ keywords را وارد کنید (برای خروج می‌توانید دکمهٔ 'q' را بزنید).

شناسه‌هایی از قبیل int، input، print و ... را می‌توان به عنوان نام یک متغیر استفاده کرد اما اینکار مشکلاتی را به وجود می‌آورد؛ بنابر این بهتر است اینکار را انجام ندهیم.

```
>>> input = 12
>>> input('Enter a name:')
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    input('Enter a name:')
TypeError: 'int' object is not callable
>>>
```

1. keyword

با استفاده از دستور `dir(__builtins__)` می‌توان لیست شناسه‌های معروف پایتون را دید. بهتر است از این شناسه‌ها به عنوان نام استفاده نکنیم.

```
>>> dir(__builtins__)
['ArithmeticError', 'AssertionError', …, 'str', 'sum',
'super', 'tuple', 'type', 'vars', 'zip']
>>>
```

نکات مهم:

ـ متغیر نامی است که در ارتباط با یک مقدار می‌باشد. از عملگر انتساب (=) برای انتساب مقداری به یک متغیر استفاده می‌شود.

ـ تمام مقادیری که توسط تابع `input` برگردانده می‌شود از نوع رشته‌ای است. توابع داخلی `int()` و `float()` برای تبدیل رشته به عدد صحیح یا اعشاری استفاده می‌شوند.

ـ شناسه‌ها دنباله‌ای از یک یا بیشتر نویسه می‌باشند که برای نامگذاری عضوی از برنامه استفاده می‌شوند.

ـ کلمات کلیدی شناسه‌هایی هستند که در زبانهای برنامه‌نویسی معنای خاصی دارند لذا از آنها نمی‌توان برای شناسه استفاده کرد. در صورت استفاده با پیغام خطای املایی مواجه می‌شویم.

۳-۲-۳ انواع داده‌ها و عبارتها

قبل از معرفی انواع داده‌ها و ساختار عبارتها در پایتون بهتر است با انواع عملگرها آشنا شویم. عملگر[1] علامتی است که نشان دهندهٔ عملیاتی است که قراراست بر روی یک یا دو *عملوند*[2] انجام شود. برای مثال علامت + به معنای جمع دو عملوند مانند ۲ و ۳ است (یعنی ۳+۲). برخی عملگرها تنها یک عملوند دارند، به آنها *عملگر یکتایی*[3] می‌گوییم. عملگر منفی کردن جزء این دسته می‌باشد (۲- و یا ۴/۷-). برخی دیگر از عملگرها دو عملوند دارند، آنها را *عملگر دودویی*[4] می‌نامیم. مانند +، ـ (تفریق)، * و ....

1. Operator
2. Operand
3. Unary Operator
4. Binary Operator

جدول ۳-۶ عملگرهای ریاضی در پایتون

عملگر ریاضی		مثال	نتیجه
-x	منفی سازی	-20	-20
x+y	جمع	12+20	32
x - y	تفریق	3-20	-17
x * y	ضرب	3*10	30
x/y	تقسیم	12/5	2.4
x//y	تقسیم صحیح	12//5	2
		12//5.0	2.0
x%y	باقی‌مانده	25%10	5
X**y	توان	10**2	100

درجدول۳-۶ عملگرهای معروف ریاضی در پایتون آمده است. همانگونه که مشاهده می‌کنید در پایتون دو نوع تقسیم وجود دارد: *تقسیم واقعی* که با علامت «/» مشخص می‌شود (حاصل تقسیم 25/10 برابر است با 2.5) و *تقسیم صحیح* که با علامت «//» نمایش داده می‌شود (حاصل25//10 برابر با 2 است). در تقسیم صحیح اگر عملوندها هر دو صحیح بودند نتیجه نیز صحیح است؛ اما اگر یکی یا هر دو عملوند اعشاری باشد نتیجه نیز اعشاری می‌شود (دقت نمایید که نتیجه اعشاری است اما همیشه قسمت اعشار صفر است). در جدول ۳-۷ این موضوع به وضوح مشخص شده است.

جدول ۳-۷ عملگر تقسیم در پایتون

	نوع عملوندها	نوع نتیجه	مثال	نتیجه
تقسیم (/)	int, int	float	7/5	1.4
	int, float	float	7/5.0	1.4
	float, float	float	7.0/5.0	1.4
تقسیم صحیح (//)	int, int	int	7//5	1
	int, float	float	7//5.0	1.0
	float, float	float	7.0//5.0	1.0

آزمایش کنید!

دستورات زیر را در محیط تعاملی پایتون اجرا کرده و نتایج آنرا به دقت بررسی نمایید:

```
>>> 10+35          >>> 4**2          >>> 45//10.0
???                ???               ???
>>> -10+35         >>> 45/10         >>> 2025%10
???                ???               ???
>>> 4*2            >>> 45//10        >>> 2025//10
???                ???               ???
```

مقادیر و انواع داده‌ها:

همانگونه که در بخش قبل مشاهده کردید انجام یک عملگر وابسته به نوع عملوندهای آن می‌باشد. عملوندها (مقادیر) پایه‌ای‌ترین چیزی هستند که یک برنامه می‌تواند آنرا تغییر دهد. مقادیر به دسته‌های مختلف تقسیم بندی می‌شوند: ۴ یک مقدار عددی و "Hello, World" یک مقدار رشته‌ای است. اگر مطمئن نیستید که مقدار مورد نظر از چه نوع است، تابع type() در پایتون برای این منظور ارائه شده است.

```
>>> type("Hello")
<class 'str'>
>>> type(12)
<class 'int'>
>>> type(-4.67)
<class 'float'>
>>> type(False)
<class 'bool'>
```

همانگونه که انتظار می‌رفت تابع type() به درست نوع مقادیر ورودی را تعیین می‌کند. خروجی تابع برای مقادیری شبیه به "21.7" و "-5" چیست؟ بله درست است 'str'. با وجود اینکه مقادیر نشان دهندهٔ اعداد هستند اما چون بین دوعلامت «"» محصور شده‌اند به عنوان مقادیر رشته‌ای تعبیر می‌شوند.

```
>>> type("17")
<class 'str'>
>>> type("3.2")
<class 'str'>
```

عبارتها

عبارتها، ترکیبی از عملگرها و عملوندها می‌باشند که پس از اجرا مقداری را به عنوان نتیجه تولید می‌کنند. عملوندها می‌توانند نام متغیر، مقدار ثابت و یا فراخوانی تابع [۱] باشند. اگر عبارتی را در پوستهٔ تعاملی پایتون اجرا کنید بلافاصله نتیجهٔ آنرا خواهید دید.

```
>>> 12 + (7 * 8.3)
70.10000000000001
```

عبارتهایی که نتیجهٔ آنها مقادیر عددی است، عبارتهای ریاضی نامیده می‌شوند. در مورد ارزیابی عبارتهای ریاضی در فصل قبل بطور مفصل صحبت کردیم. در اینجا تنها به بیان چند مثال اکتفا می‌کنیم. توجه نمایید که اولویت عملگرها طبق جدول۳-۸ می‌باشد:

```
1)  1+2**5-2          →   1+32-2        →   31
2)  4+2**5//10        →   4+32//10      →   4+3     → 7
3)  12%(10/2)         →   12%5.0        →   2
4)  2**2**3           →   2**8          →   256
5)  4**(20//3%4)*3    →   4**(6%4)*3    →   4**2*3 → 48
```

۳-۲-۴ برنامه: تبدیل سن به ثانیه

می‌خواهیم برنامه‌ای بنویسیم که سن شخصی را به ثانیه محاسبه و چاپ نماید. محاسبهٔ دقیق سن افراد به ثانیه امکان پذیر نیست زیرا برای محاسبه باید ثانیه‌ای که فرد در آن متولد شده، اختلاف ساعت ناحیه‌ای که درآنجا متولد شده، سالهای کبیسه‌ای که گذشته و غیره را در نظر گرفت. بنابراین در این مثال تقریبی مناسب از سن فرد به ثانیه محاسبه خواهد شد. پس صورت سوال بصورت زیر است:

می‌خواهیم برنامه‌ای بنویسیم که سن تقریبی فردی را به ثانیه محاسبه و چاپ نماید. برنامه باید تاریخ‌هایی که از اول فروردین ۱۳۰۰ تا تاریخ جاری است را بپذیرد.

در مورد نحوهٔ ورود اطلاعات سخت‌گیری انجام نمی‌دهیم. هیچ قالب خاصی برای ورود تاریخ تولد در نظر گرفته نمی‌شود. فعلاً با توجه به اینکه هنوز دستورات لازم برای چک کردن خطای ورودی را یاد نگرفته‌ایم فرض می‌کنیم کاربر اطلاعات را در محدودهٔ مورد نظر وارد می‌نماید.

---

۱. در فصل ۵ در مورد تابع توضیح داده م شود.

جدول ۳-۸ اولویت عملگرها در پایتون

اولویت	عملگرها	توضیح	شرکت پذیری
بالاترین اولویت	[] , ()	برای دسته بندی، برش، فراخوانی تابع و دسترسی به عناصر آرایه	→
↓	**	عملگر توان	←
	+، -، ~	عملگرهای مثبت ومنفی، همچنین نقیض بیتی	
	×، //،/، ٪	ضرب، تقسیم و باقیماندهٔ تقسیم	→
	+، -	جمع و تفریق	
	<, <=, >, >=	عملگرهای شرطی۱	
	!=, ==	عملگرهای شرطی۲	
	and	عملگر منطقی۱	
پایین ترین اولویت	Or	عملگر منطقی۲	

در این برنامه باید تاریخ تولد کاربر و تاریخ جاری را طوری نگهداری کنیم که اجزای آنها (سال، ماه و روز) را بتوان جداگانه بررسی کرد؛ لذا تاریخ‌ها را با سه مقدار عددی نمایش می‌دهیم. برای مثال تاریخ «۳۰ شهریور ۱۳۶۰» را بصورت زیر نمایش می‌دهیم:

```
year = 1360      month = 6      day = 30
```

برای کارکردن با تاریخ‌های شمسی از ماژول[1] jdatetime که توسط آقای میلاد راستیان تهیه شده استفاده می‌شود. این ماژول بر پایهٔ ماژول datetime از کتابخانهٔ استاندارد پایتون نوشته شده است.

برای شروع بهتر است مسئلهٔ سال کبیسه را درنظر بگیریم. با توجه به اینکه سال کبیسه هر چهار سال یکبار اتفاق می‌افتد (البته با چند استثنا)، متوسط تعداد ثانیه‌های یک سال را در دوره‌های چهارساله بعلاوهٔ یک سال کبیسه محاسبه خواهیم کرد. با توجه به اینکه سالهای غیرکبیسه ۳۶۵ روز و سالهای کبیسه ۳۶۶ روز دارند متوسط تعداد ثانیه‌های یک سال بصورت زیرمحاسبه می‌شوند:

```
numsecs_day = (hours per day) * (mins per hour) * (secs per minute)
```

۱. این کتابخانه براساس پایتون نسخهٔ ۲/۷ تهیه شده که مولف کتاب آنرا برای نسخهٔ ۳ پایتون آماده کرده است.

```
numsecs_year = (days per year) * numsecs_day
avg_numsecs_year = (4 * numsecs_year) + numsecs_day) // 4
avg_numsecs_month = avg_numsecs_year // 12
```

برای محاسبهٔ سن کاربر به ثانیه، تاریخ اول فروردین ۱۳۰۰ به عنوان تاریخ پایه در نظر می‌گیریم. بنابر این باید دو مقدار را محاسبه کنیم: اول تعداد ثانیه‌ها از تاریخ پایه تا تاریخ تولد کاربر و دیگری تعداد ثانیه‌ها از تاریخ پایه تا تاریخ جاری. کم کردن مقدار اول از مقدار دوم مقدار مورد نظر ماست.

اگر بخواهیم تعداد ثانیه‌ها را مستقیم از تاریخ تولد و تاریخ جاری محاسبه کنیم باید تعداد ماه‌ها و سالهای کامل بین این دو تاریخ را محاسبه نماییم. اما اگر از تاریخ پایه استفاده کنیم دیگر به این محاسبات نیازی نداریم. ادامهٔ محاسبات بصورت زیر است:

```
numsecs_1300_to_dob = (year_birth - 1300) * avg_numsecs_year +
 (month_birth - 1) * avg_numsecs_month + (day_birth * numsecs_day)
numsecs_1300_to_today = (current_year - 1300) * avg_numsecs_year +
 (current_month - 1) * avg_numsecs_month + (current_day * numsecs_day)
age_in_secs = numsecs_1300_to_today - numsecs_1300_to_dob
```

برنامهٔ ۲-۳ راه حل مسئله را بیان کرده است.

نکات مهم:

- عملگر علامتی است که نشان دهندهٔ نوع عملی است که بر روی یک یا چند عملوند انجام می‌شود.
- در پایتون علامت «/» نشان دهندهٔ عمل تقسیم واقعی و علامت «//» نشان دهندهٔ عمل تقسیم صحیح است. برای بدست آوردن باقیماندهٔ تقسیم از علامت % استفاده می‌شود.
- عبارت ترکیبی از عملگرها و عملوندها می‌باشد که پس از ارزیابی با مقداری مشخص جایگزین می‌شوند.
- به مجموعه‌ای از مقادیر و عملگرهایی که بر روی آن مقادیر اعمال می‌شوند نوع داده گفته می‌شود.

کد ۳-۲ برنامهٔ محاسبهٔ سن به ثانیه

```
# Age in seconds program
# This program will calculate a person's approximate age in
seconds

import jdatetime

# Get year, month and day of birth
year_birth = int(input("Enter year born (4 digit):"))
month_birth = int(input("Enter month born (1-12):"))
day_birth = int(input("Enter day born (1-31):"))

# Get current year, month and day
year_current = jdatetime.date.today().year
month_current = jdatetime.date.today().month
day_current = jdatetime.date.today().day

# Determine number of seconds in a day, avg. month and avg. year
numsecs_day = 24 * 60 * 60
numsecs_year = 365 * numsecs_day

avg_numsecs_year = ((4 * numsecs_year) + numsecs_day) // 4
avg_numsecs_month = avg_numsecs_year // 12

# Calculate approximate age in seconds
numsecs_1300_to_dob = (year_birth - 1300) * avg_numsecs_year +\
    (month_birth - 1) * avg_numsecs_month + (day_birth *
numsecs_day)
numsecs_1300_to_today=(year_current - 1300)*avg_numsecs_year +\
    (month_current - 1) * avg_numsecs_month + day_current *
numsecs_day)
age_in_secs = numsecs_1300_to_today - numsecs_1300_to_dob
print(numsecs_1300_to_dob, numsecs_1300_to_today)

# Output results
print("You are approximately", age_in_secs, 'seconds old.')
```

View publication stats