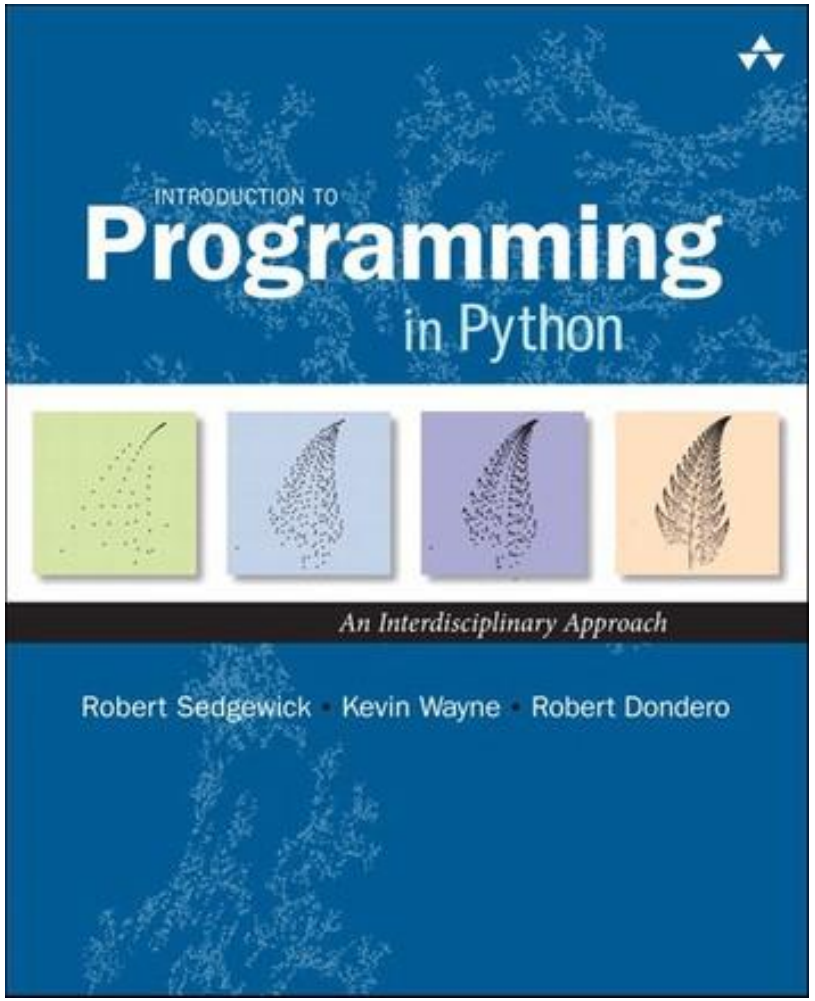
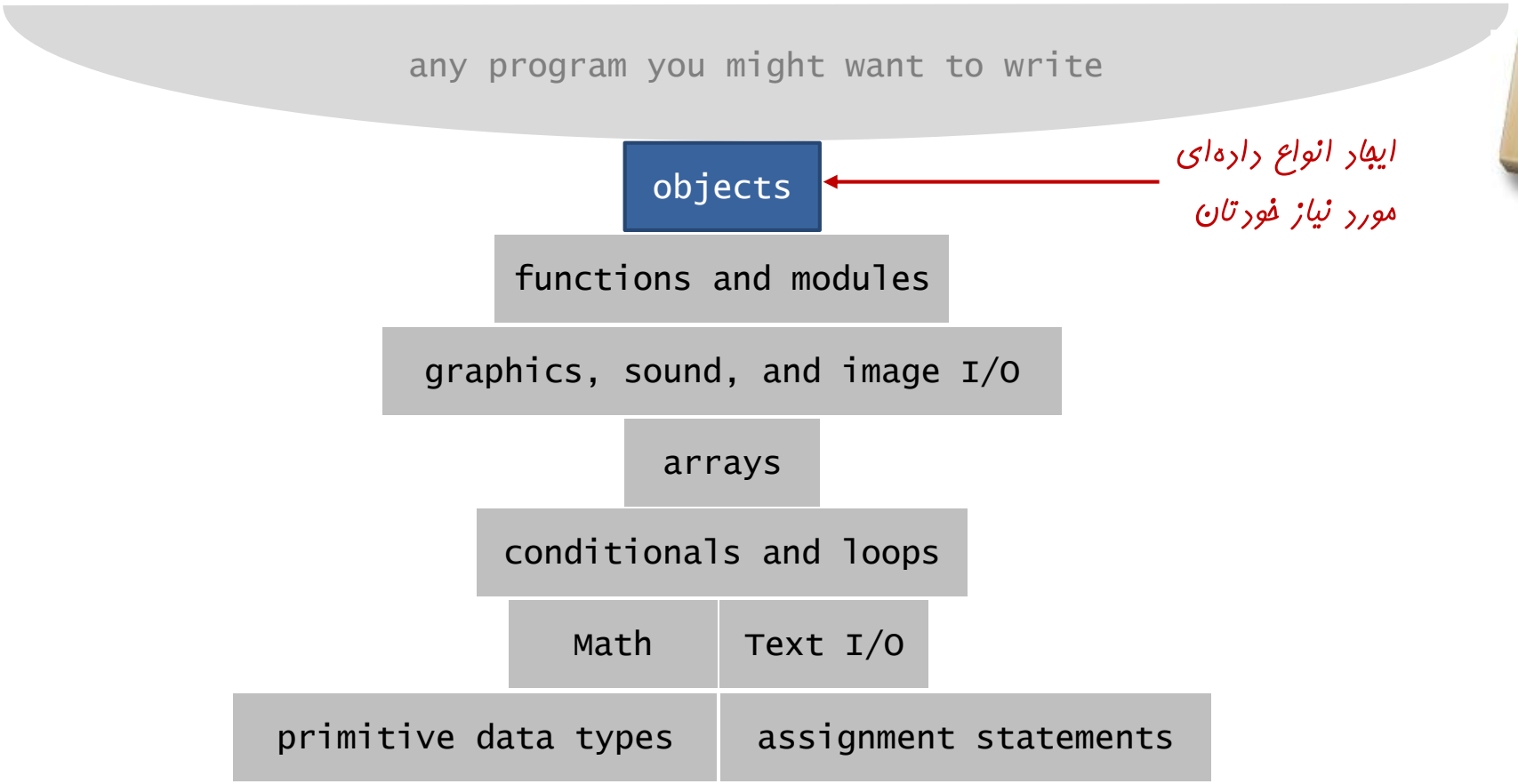
برنامه‌نویسی شی‌گرا: استفاده از انواع داده‌ای

سید ناصر رضوی **www.snrazavi.ir**
۱۳۹۸

۳-۱ استفاده از انواع داده‌ای



اجزای برنامه‌نویسی



انواع داده‌ای

- نوع داده‌ای. یک مجموعه از مقادیر و عملیات قابل انجام بر روی آنها.
- انواع اولیه. عملیات آنها مستقیماً به دستورالعمل‌های ماشین ترجمه می‌شوند.

Data Type	*Values*	*Operations*
`boolean`	`True, False`	`not, and, or, xor`
`int`	-2^{31} to $2^{31} - 1$	`add, subtract, multiply`
`float`	2^{32} possible real values	`add, subtract, multiply`

- برنامه‌نویسی شی‌گرا. می‌خواهیم برنامه‌هایی بنویسیم که بتوانند هر نوع از داده‌ها را پردازش کنند.
  - رنگ‌ها، تصاویر، رشته‌ها، جریان‌های ورودی، ...
  - اعداد مختلط، بردارها، ماتریس‌ها، چندجمله‌ای‌ها، ...
  - نقطه‌ها، چندضلعی‌ها، ذرات باردار، اجرام آسمانی، ...

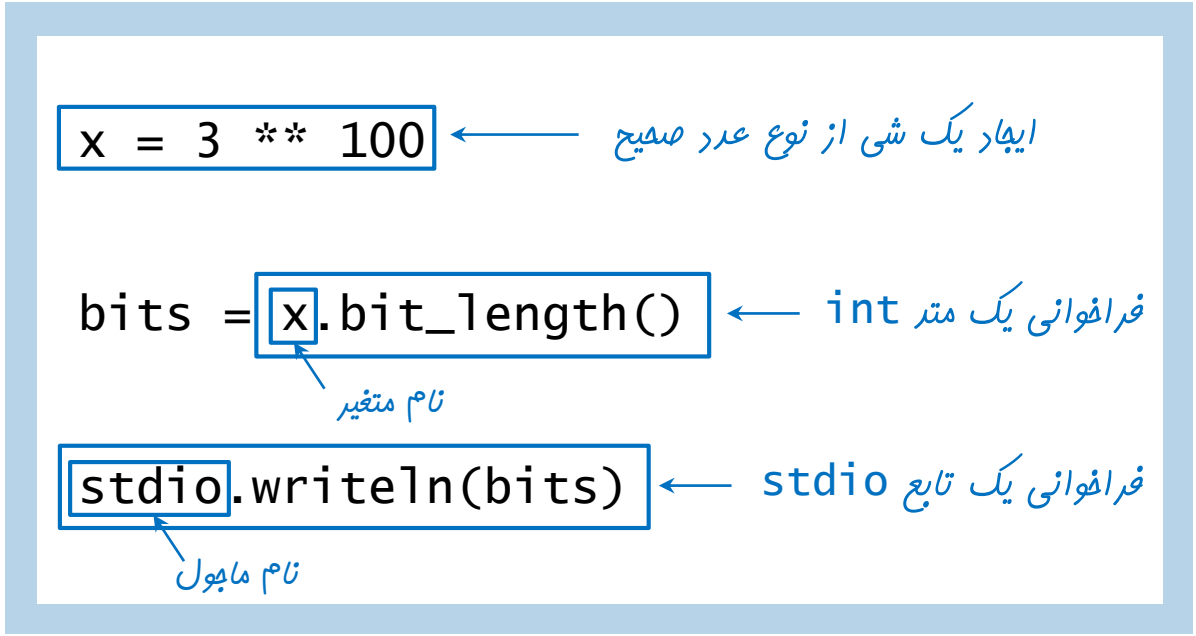
اشیا

- شی. دربردارنده مقدار یک نوع داده‌ای؛ نام متغیر به شی ارجاع می‌دهد.

- کاربرد.
  - به وسیله اشیا می‌توانیم هر نوع داده‌ای را که نیاز داریم، خودمان ایجاد کنیم.
  - سپس عملیات لازم را بر روی آنها تعریف و از آنها در برنامه‌هایمان استفاده می‌کنیم.

Data Type	*Values*	*Operations*
Color	24 bits (R, G, B)	get red component, brighten
Picture	2D array of colors	get or set color of pixel (i, j)
String	sequence of characters	length, substring, compare

کار با اشیا: ایجاد و استفاده



مقایسه متدها با توابع

- متد. تابع مرتبط با یک شی خاص.

	متد	تابع
نمونه فراخوانی	`x.bit_length()`	`stdio.writeln(bits)`
فراخوانی با	نام متغیر	نام ماجول
پارامترها	یک ارجاع به شی و آرگومان‌ها	آرگومان‌ها
هدف اصلی	دستکاری مقدار شی	محاسبه مقدار برگشتی

پردازش رشته

عمل	توصیف
`len(s)`	تعداد کاراکترهای رشته `s`
`s + t`	یک رشته جدید حاصل از اتصال رشته‌های `s` و `t`
`s += t`	یک رشته جدید حاصل از اتصال رشته‌های `s` و `t` و انتساب آن به `s`
`s[i]`	کاراکتر واقع در اندیس `i` از رشته `s`
`s[i:j]`	کاراکتر واقع در اندیس `i` تا کاراکتر واقع در اندیس `j - 1` از رشته `s`
`s[i:j:k]`	یک برش از `i` تا `j` با اندازه گام `k`
`s < t`	آیا `s` کوچک‌تر از `t` است؟
`s <= t`	آیا `s` کوچک‌تر یا مساوی `t` است؟
`s == t`	آیا `s` مساوی با `t` است؟
`s != t`	آیا `s` مساوی با `t` نیست؟
`s >= t`	آیا `s` بزرگ‌تر یا مساوی `t` است؟
`s > t`	آیا `s` بزرگ‌تر از `t` است؟

پردازش رشته

عمل	توصیف
s in t	آیا s به عنوان زیررشته در t ظاهر شده است؟
s not in t	آیا s به عنوان زیررشته در t ظاهر نشده است؟
s.count(t)	تعداد رخدادهای زیررشته t در رشته s
s.find(t, start)	اولین اندیس وقوع t در رشته s با شروع از start
s.upper()	یک کپی از s که در آن حروف کوچک با حروف بزرگ جایگزین شده‌اند
s.lower()	یک کپی از s که در آن حروف بزرگ با حروف کوچک جایگزین شده‌اند
s.startswith(t)	آیا رشته s با زیررشته t آغاز شده است؟
s.endswith(t)	آیا رشته s با زیررشته t پایان یافته است؟
s.strip()	یک کپی از s که در آن فضاهای خالی ابتدایی و انتهایی حذف شده‌اند
s.replace(old, new)	یک کپی از s که در آن تمام رخدادهای old با new جایگزین شده‌اند
s.split(delimiter)	یک آرایه از زیررشته‌های s که با delimiter جدا شده‌اند
delimiter.join(a)	اتصال رشته‌های موجود در آرایه a و قرار دادن delimiter بین آنها

پردازش رشته

- مثال. تبدیل DNA به mRNA با تغییر حروف T به U.

```
def translate(dna):
    dna = dna.upper()
    rna = dna.replace('T', 'U')
    return rna
```

ATGCGGTACTTA

AUGCGGUACUUA

پردازش رشته

- مثال. بررسی این که آیا یک رشته با معکوس خود برابر است یا خیر.

```
def isPalindrome(s):
    n = len(s)
    for i in range(n//2):
        if s[i] != s[n-1-i]:
            return False
    return True
```

0	1	2	3	4	5	6	7	8	9	10
a	i	b	o	h	p	h	o	b	i	a

پردازش رشته

- مثال. استخراج نام فایل و پسوند فایل از یک آرگومان خط فرمان.

```
s = sys.argv[1]
dot = s.find('.')
base = s[:dot]
extension = s[dot+1:]
```

mandill.jpg

base extension

پردازش رشته

- مثال. بررسی این که آیا رشته‌های موجود در یک آرایه به ترتیب صعودی قرار دارند.

```
def isSorted(a):
    for i in range(1, len(a)):
        if a[i] < a[i-1]:
            return False
    return True
```

یافتن ژن

- ژنوم. نمایش ژنوم به عنوان یک رشته بر روی الفبای {A, C, T, G}.

- ژن. یک زیررشته از ژنوم که بیانگر یک واحد عملیاتی است.
  - با ATG شروع می‌شود. [کودون شروع]
  - مضربی از ۳ نوکلئوتید. [کودون‌های غیر از شروع و پایان]
  - با TAG، TAA یا TGA پایان می‌یابد. [کودون‌های پایان]

- هدف. یافتن تمام ژن‌ها.

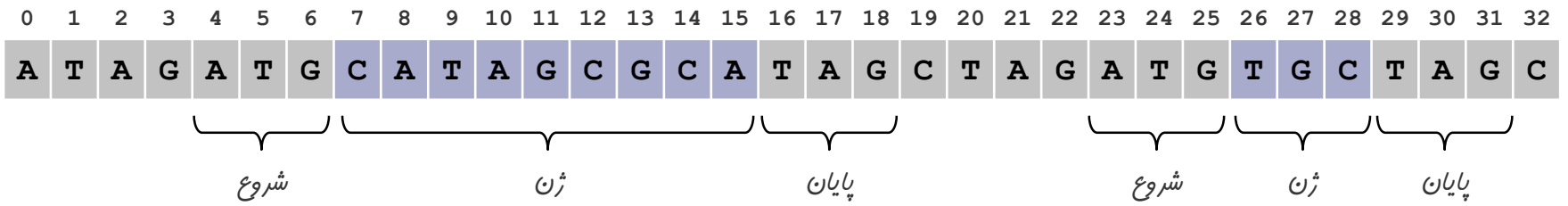
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
A	T	A	G	A	T	G	C	A	T	A	G	C	G	C	A	T	A	G	C	T	A	G	A	T	G	T	G	C	T	A	G	C

یافتن ژن

- ژنوم. نمایش ژنوم به عنوان یک رشته بر روی الفبای {A, C, T, G}.

- ژن. یک زیررشته از ژنوم که بیانگر یک واحد عملیاتی است.
  - با ATG شروع می‌شود. [کودون شروع]
  - مضربی از ۳ نوکلئوتید. [کودون‌های غیر از شروع و پایان]
  - با TAG، TAA یا TGA پایان می‌یابد. [کودون‌های پایان]

- هدف. یافتن تمام ژن‌ها.



بررسی ژن: پیاده‌سازی


```
def isPotentialGene(dna):
    if (len(dna) % 3) != 0: return False
    if not dna.startswith('ATG'): return False
    for i in range(len(dna) - 3):
        if dna[i:i+3] == 'TAA': return False
        if dna[i:i+3] == 'TAG': return False
        if dna[i:i+3] == 'TGA': return False
    if dna.endswith('TAA'): return True
    if dna.endswith('TAG'): return True
    if dna.endswith('TGA'): return True
    return False
```

```
% python gene.py ATGCATAGCGCATAG
true

% python gene.py ATGCGCTGCGTCTGTACTAG
false

% python gene.py ATGCCGTGACGTCTGTACTAG
false
```

یافتن ژن: الگوریتم

- الگوریتم. ژنوم را از چپ به راست پویش کن.
  - اگر کودون شروع، آنگاه مقدار **beg** را برابر با اندیس **i** قرار بده.
  - اگر کودون پایان و زیررشته مضربی از سه است:
    - ژن را به خروجی بفرست.
    - مقدار **beg** را برابر با

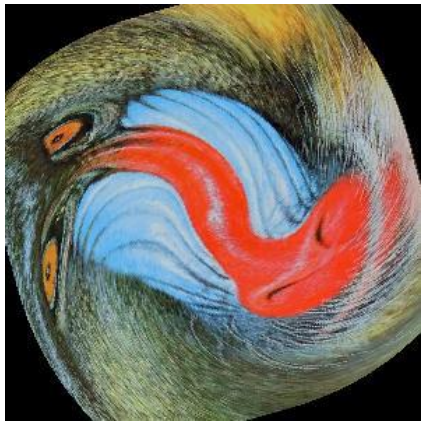
i	codon		beg	gene	input string
	start	*stop*			
0			-1		ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
1		TAG	-1		ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
4	ATG		4		ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
9		TAG	4	مضرب ۳	ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
16		TAG	4	CATAGCGCA	ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
20		TAG	-1		ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
23			23		ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
29		TAG	23	TGC	ATAGATGCATAGCGCATAGCTAGATGTGCTAGC

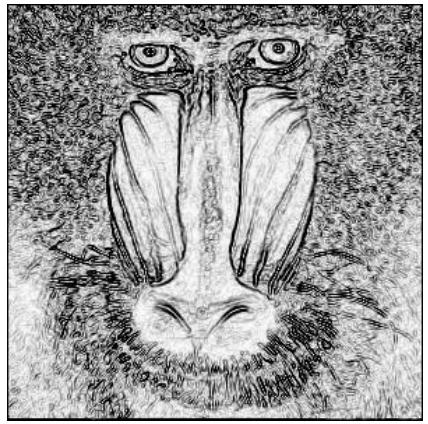
یافتن ژن: پیاده‌سازی


```
def main():
    start = sys.argv[1]
    stop  = sys.argv[2]
    genome = stdio.readAll()
    beg = -1
    for i in range(len(genome) - 2):
        codon = genome[i:i+3]
        if codon == start: beg = i
        if codon == stop and beg != -1:
            gene = genome[beg+3:i]
            if len(gene) % 3 == 0:
                stdio.writeln(gene)
                beg = -1
```

```
% more genomeTiny.txt
ATAGATGCATAGCGCATAGCTAGATGTGCTAGC

% python genefind.py ATG TAG < genomeTiny.txt
CATAGCGCA
TGC
```

پردازش تصویر



نوع داده‌ای رنگ

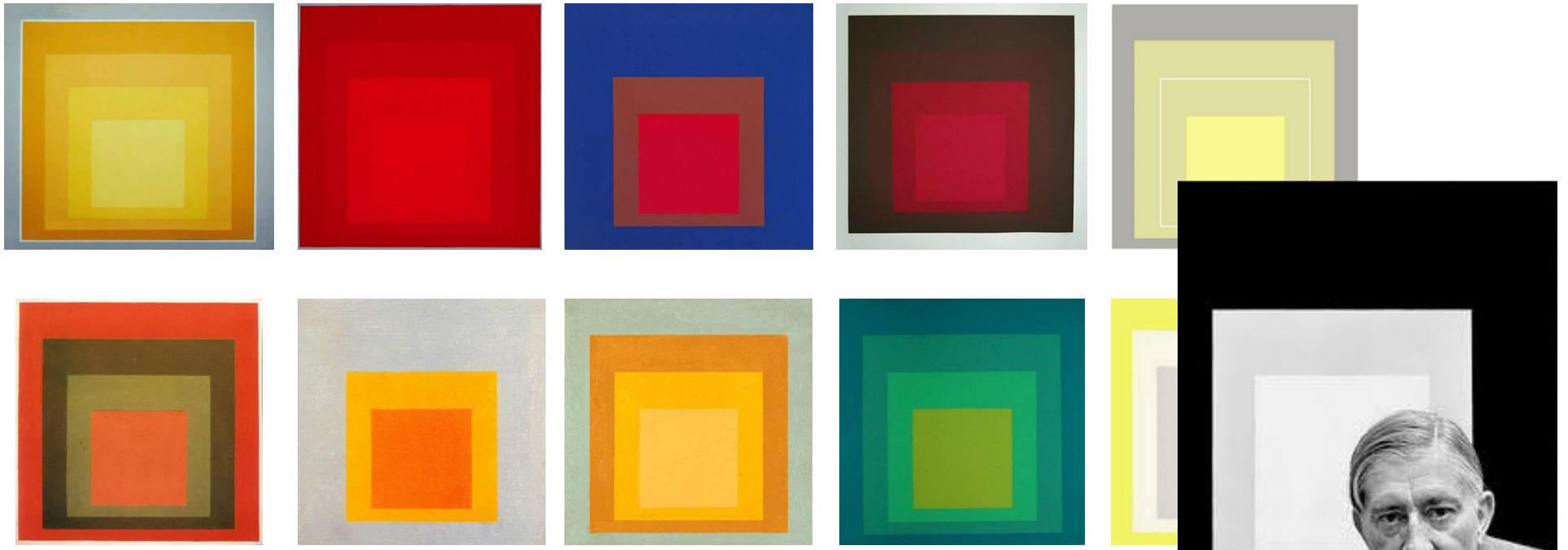
- رنگ. درک چشم از تشعشعات الکترومغناطیس.
- مجموعه مقادیر. [نمایش RGB] $۲۵۶^۳$ مقدار ممکن، که بیانگر مقدار قرمز، سبز و آبی هر یک در بازه صفر و ۲۵۵ هستند.

عمل	توصیف
`Color(r, g, b)`	*ایجاد یک رنگ جدید با مولفه‌های قرمز، سبز و آبی مشخص شده*
`c.getRed()`	*مولفه قرمز از شی رنگ* `c`
`c.getGreen()`	*مولفه سبز از شی رنگ* `c`
`c.getBlue()`	*مولفه آبی از شی رنگ* `c`
`str(c)`	*یک بازنمایی رشته‌ای از* `c` *به صورت* `'(R, G, B)'`

R	G	B	Color
255	0	0	
0	255	0	
0	0	255	
0	0	0	
255	255	255	
255	0	255	
105	105	105	

مربع‌های آلبرز

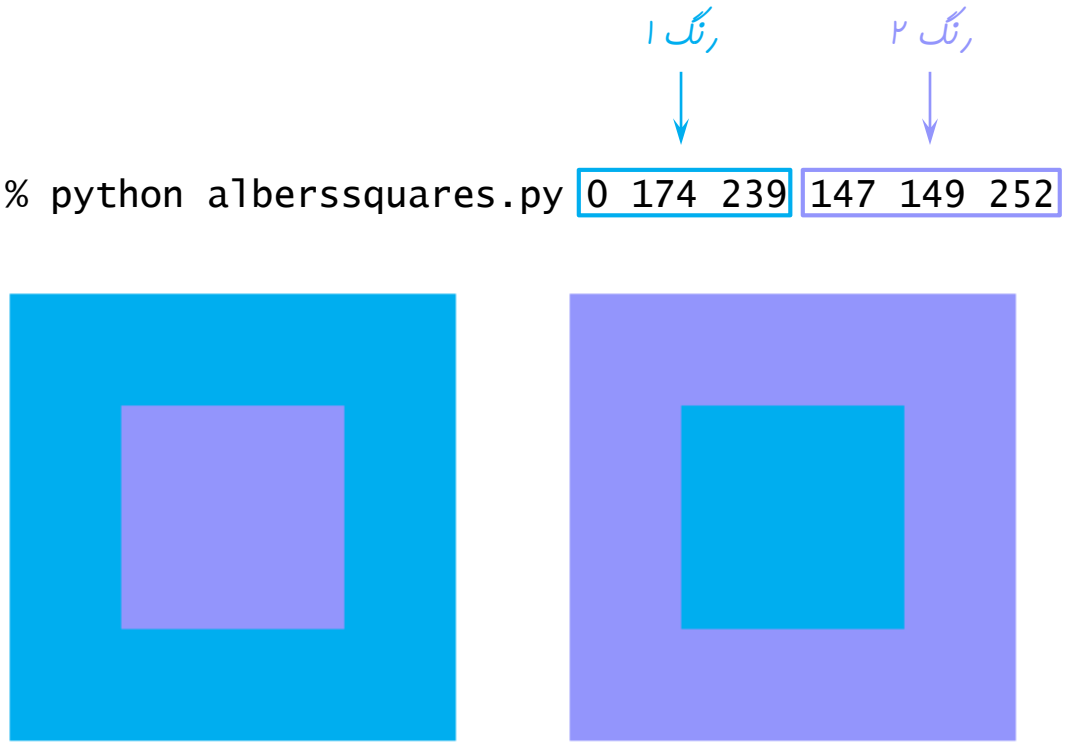
- ژوزف آلبرز. شیوه نگرش مردم به رنگ‌ها را دگرگون نمود.

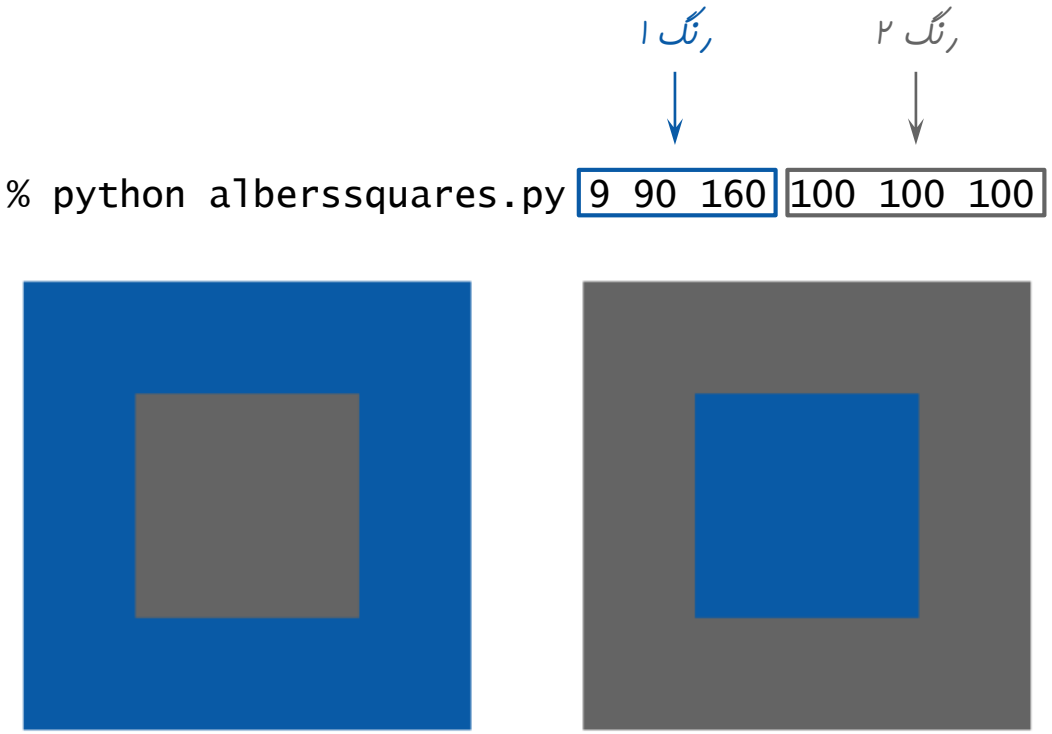


مربع‌های ژوزف البرز (۱۹۴۹ – ۱۹۷۵)

مربع‌های آلبرز

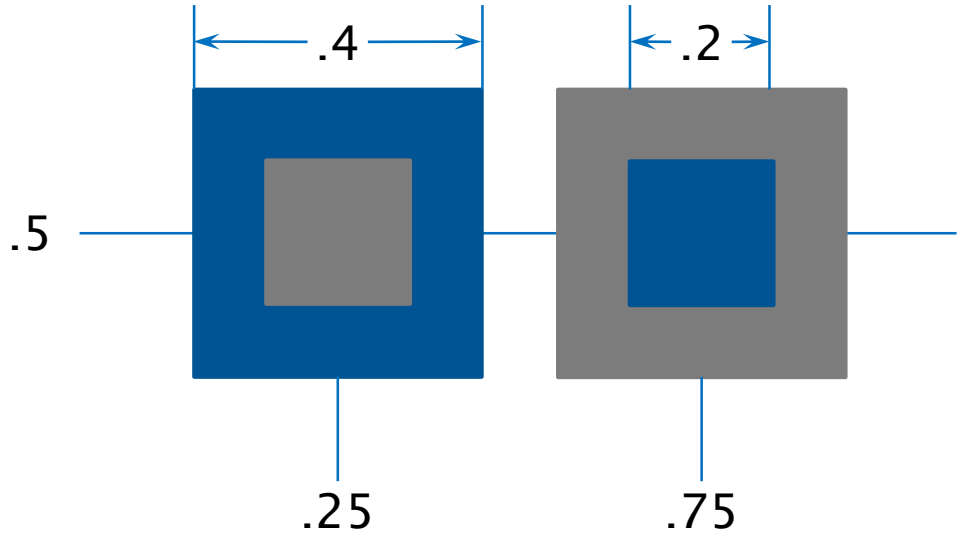
- ژوزف آلبرز. شیوه نگرش مردم به رنگ‌ها را دگرگون نمود.





استفاده از رنگ‌ها در پایتون

```
import sys, stddraw
from color import Color

r1 = int(sys.argv[1])
g1 = int(sys.argv[2])
b1 = int(sys.argv[3])
c1 = Color(r1, g1, b1)

r2 = int(sys.argv[4])
g2 = int(sys.argv[5])
b2 = int(sys.argv[6])
c2 = Color(r2, g2, b2)

stddraw.setCanvasSize(512, 256)
stddraw.setYscale(.25, .75)

stddraw.setPenColor(c1)
stddraw.filledSquare(.25, .5, .2)

stddraw.setPenColor(c2)
stddraw.filledSquare(.25, .5, .1)

stddraw.setPenColor(c2)
stddraw.filledSquare(.75, .5, .2)

stddraw.setPenColor(c1)
stddraw.filledSquare(.75, .5, .1)

stddraw.show()
```




```
% python alberssquares.py 9 90 160 100 100 100
```

روشنایی تک‌رنگ

- روشنایی تک‌رنگ. درخشندگی مؤثر یک رنگ.
- فرمول NTSC.

$$Y = 0.299r + 0.587g + 0.114b$$

```
def luminance(c):
    red   = c.getRed()
    green = c.getGreen()
    blue  = c.getBlue()
    return (.299 * red) + (.587 * green) + (.114 * blue)
```

سازگاری رنگ

- پرسش. بر روی صفحه نمایش کامپیوتر و تلفن‌های همراه، کدام رنگ قلم بر روی کدام رنگ پس‌زمینه خواناتر است؟

- پاسخ. [یک قاعده سر انگشتی.] اختلاف روشنایی باید بزرگ‌تر یا مساوی ۱۲۸ باشد.


```
def areCompatible(c1, c2):
    return abs(luminance(c1) - luminance(c2)) >= 128.0
```

سطح خاکستری

- سطح خاکستری. زمانی که مقدار مؤلفه‌های R، G و B با هم برابر باشند، رنگ حاصل یک رنگ خاکستری در بازه صفر (سیاه) تا ۲۵۵ (سفید) خواهد بود.
- تبدیل به خاکستری. استفاده از روشنایی به عنوان مقدار مؤلفه‌ها.

```
def toGray(c):
    y = int(round(luminance(c)))
    return Color(y, y, y)
```

R	G	B	
9	90	166	رنگ

0.299 * 9 + 0.587 * 90 + 0.114 * 166 = 74.445

سطح خاکستری

- سطح خاکستری. زمانی که مقدار مؤلفه‌های R، G و B با هم برابر باشند، رنگ حاصل یک رنگ خاکستری در بازه صفر (سیاه) تا ۲۵۵ (سفید) خواهد بود.
- تبدیل به خاکستری. استفاده از روشنایی به عنوان مقدار مؤلفه‌ها.

```
def toGray(c):
    y = int(round(luminance(c)))
    return Color(y, y, y)
```

مولفه قرمز	255	0	0	0	255	0	119	105
مولفه سبز	0	255	0	0	255	64	33	105
مولفه آبی	0	0	255	0	255	128	27	105
رنگ								
درخشندگی	76	150	29	0	255	52	58	105
رنگ خاکستری								

ارجاع به اشیا

- رنه ماگریت. «این یک پیپ نیست.»

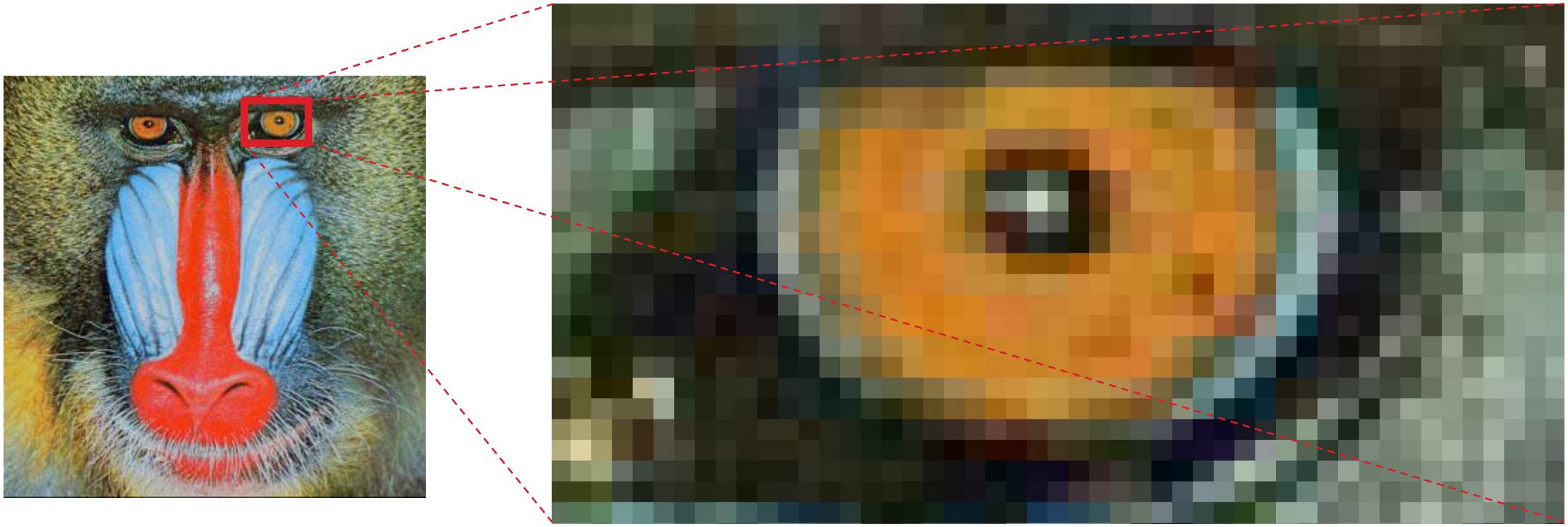
- پایتون. این یک رنگ نیست.

```
sienna = Color(160, 82, 45)
c = sienna.darker()
```

- برنامه‌نویسی شی‌گرا. یک روش طبیعی به منظور مطالعه مدل‌های انتزاعی از دنیای واقعی.

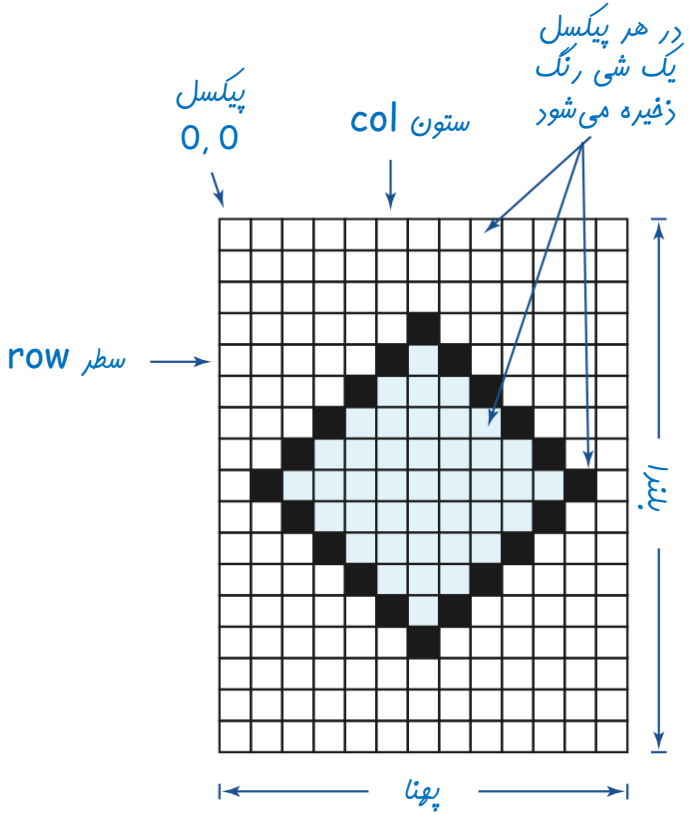
نوع داده‌ای تصویر

- نوع داده‌ای تصویر. پایه پردازش تصویر.
- مجموعه مقادیر. آرایه دو بعدی از اشیای رنگ (پیکسل‌ها).



نوع داده‌ای تصویر

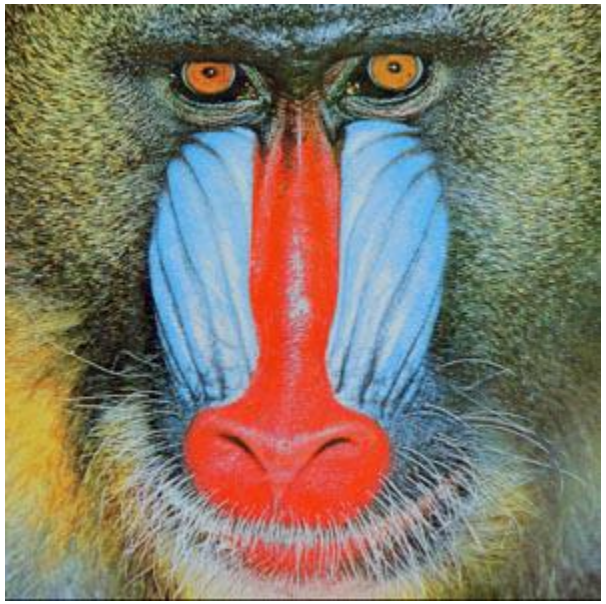
- نوع داده‌ای تصویر. پایه پردازش تصویر.
- مجموعه مقادیر. آرایه دو بعدی از اشیای رنگ (پیکسل‌ها).



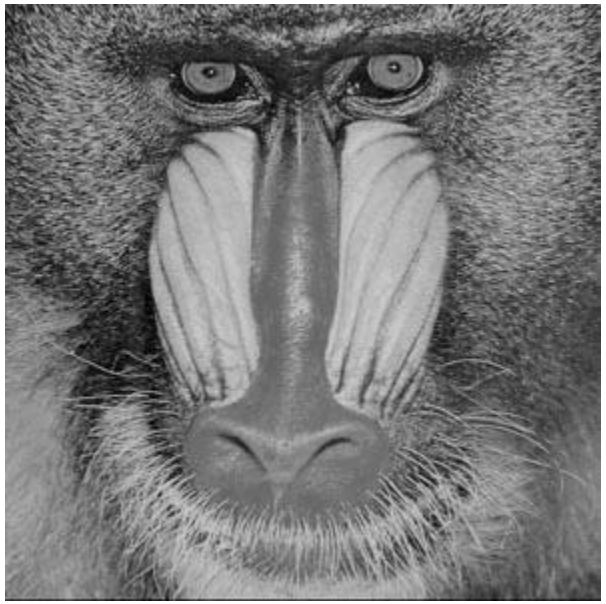
توصیف	عمل
ایجاد تصویر از یک فایل	`Picture(filename)`
ایجاد یک تصویر خالی با ابعاد w در h	`Picture(w, h)`
پهنای تصویر pic	`pic.width()`
بلندای تصویر pic	`pic.height()`
رنگ پیکسل واقع در ستون col و سطر row	`pic.get(col, row)`
تنظیم رنگ پیکسل واقع در ستون col و سطر row با رنگ c	`pic.set(col, row, c)`
ذخیره تصویر pic در یک فایل	`pic.save(filename)`

پردازش تصویر: فیلتر خاکستری

- هدف. تبدیل یک تصویر رنگی به یک تصویر خاکستری.



mandrill.jpg



```
% python grayscale.py mandrill.jpg
```

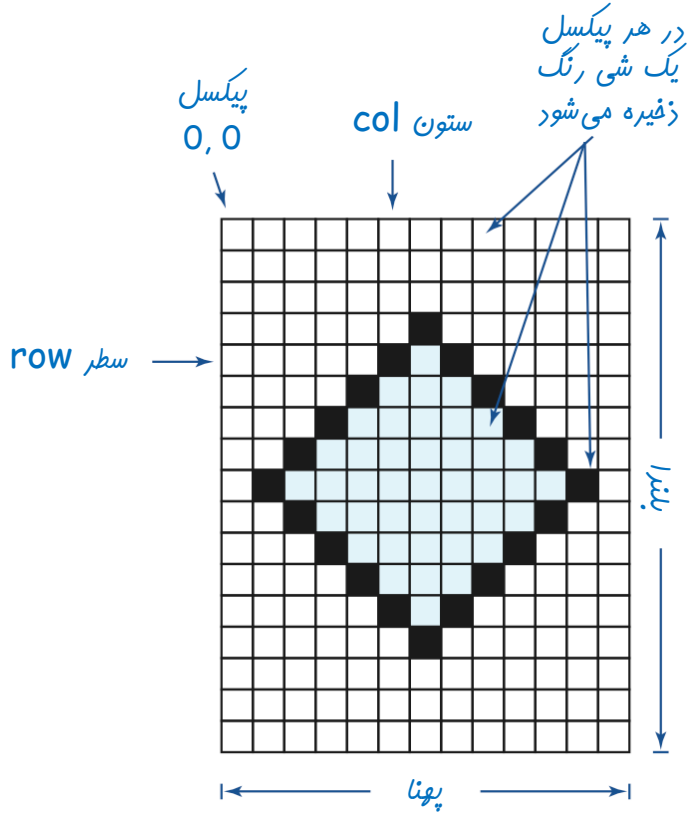
پردازش تصویر: فیلتر خاکستری

- هدف. تبدیل یک تصویر رنگی به یک تصویر خاکستری.

```
import sys, stddraw, luminance
from picture import Picture

pic = Picture(sys.argv[1])

for col in range(pic.width()):
    for row in range(pic.height()):
        pixel = pic.get(col, row)
        gray = luminance.toGray(pixel)
        pic.set(col, row, gray)

stddraw.setCanvasSize(pic.width(), pic.height())
stddraw.picture(pic)
stddraw.show()
```

ایجاد یک تصویر جدید

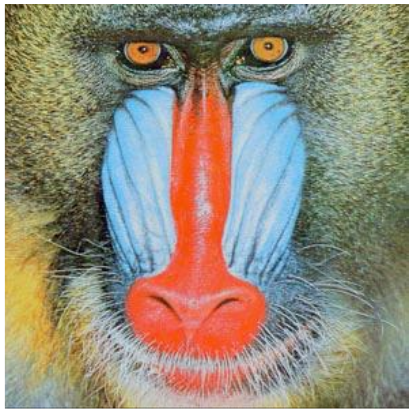
مقداردهی هر پیکسل

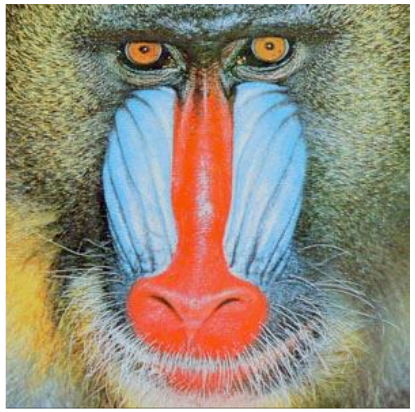


پردازش تصویر: پرسش کلاسی ۱

- نتیجه اجرای تکه‌کد زیر چیست؟

```
pic = Picture(sys.argv[1])
for col in range(pic.width()):
    for row in range(pic.height()):
        pic.set(col, row, pic.get(col, row))
```



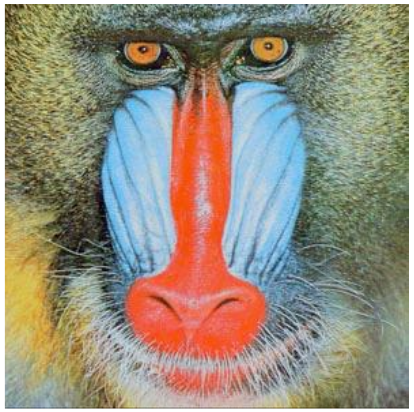


پاسخ. فقط تصویر ورودی را نمایش می‌دهد.

پردازش تصویر: پرسش کلاسی ۲

- نتیجه اجرای تکه‌کد زیر چیست؟

```
pic = Picture(sys.argv[1])
for col in range(pic.width()):
    for row in range(pic.height()):
        pic.set(col, pic.height() - row - 1, pic.get(col, row))
```

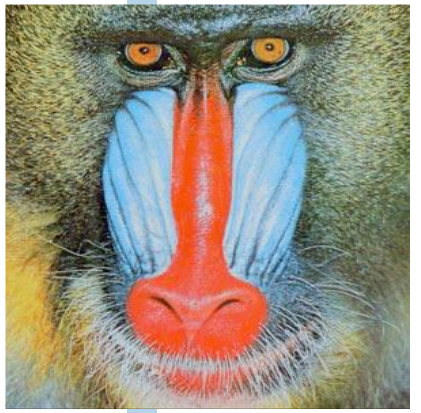


پاسخ. این تکه‌کد تلاش می‌کند تصویر ورودی را سر و ته کند؛
اما دارای یک خطای آموزنده است.

پردازش تصویر: پرسش کلاسی ۳

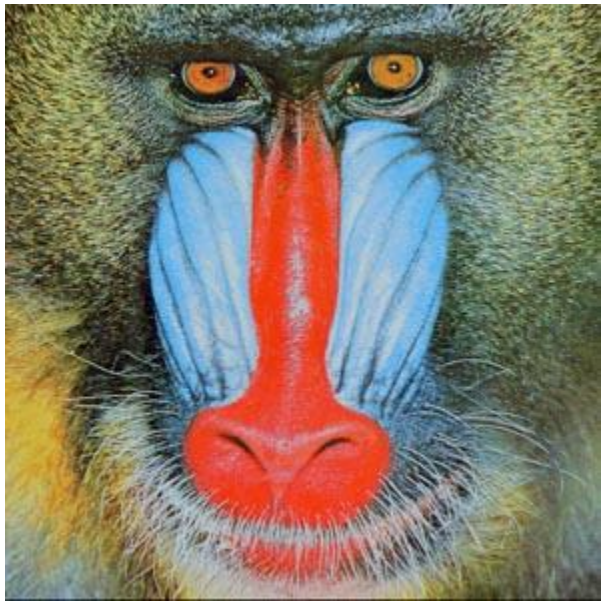
- نتیجه اجرای تکه‌کد زیر چیست؟

```
source = Picture(sys.argv[1])
w = source.width(),
h = source.height()
target = Picture(w, h)
for col in range(w):
    for row in range(h):
        target.set(col, h - row - 1, source.get(col, row))
```



پردازش تصویر: فیلتر تغییر اندازه

- هدف. کوچک کردن یا بزرگ کردن اندازه یک تصویر.

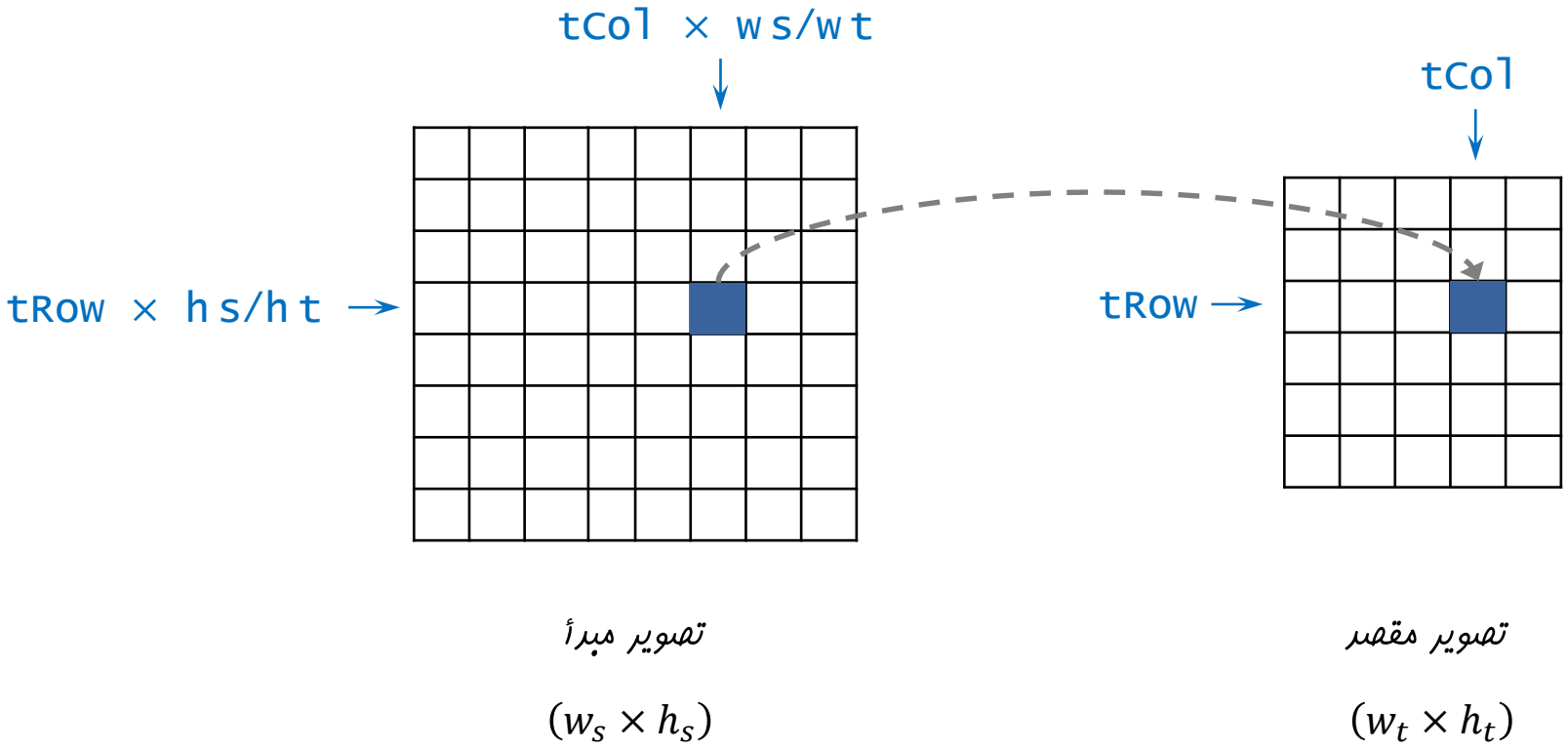


mandrill.jpg (298-by-298)

```
% python scale.py mandrill.jpg 400 200
```

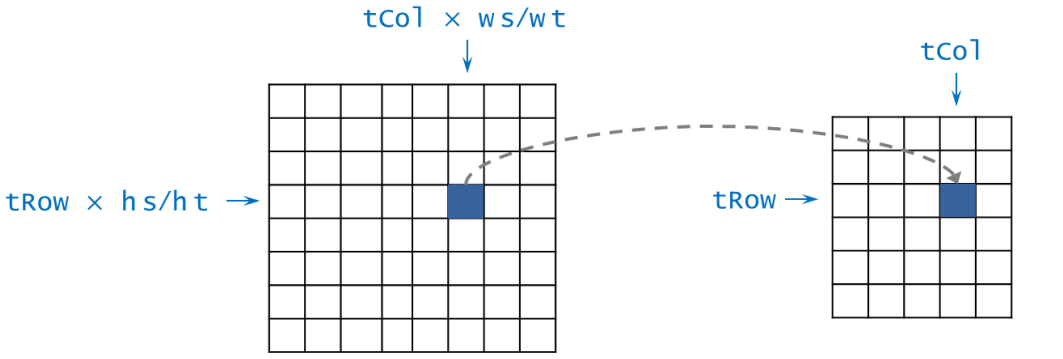
پردازش تصویر: فیلتر تغییر اندازه

- هدف. کوچک کردن یا بزرگ کردن اندازه یک تصویر.



پردازش تصویر: فیلتر تغییر اندازه

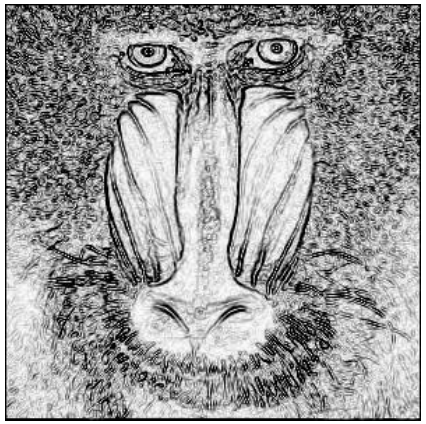
```
import sys, stddraw
from picture import Picture

fileName = sys.argv[1]
w = int(sys.argv[2])
h = int(sys.argv[3])

source = Picture(fileName)
target = Picture(w, h)

for tCol in range(w):
    for tRow in range(h):
        sCol = tCol * source.width() // w
        sRow = tRow * source.height() // h
        target.set(tCol, tRow, source.get(sCol, sRow))

stddraw.setCanvasSize(w, h)
stddraw.picture(target)
stddraw.show()
```



```
% python scale.py mandrill.jpg 400 200
```

چند فیلتر پردازش تصویر دیگر

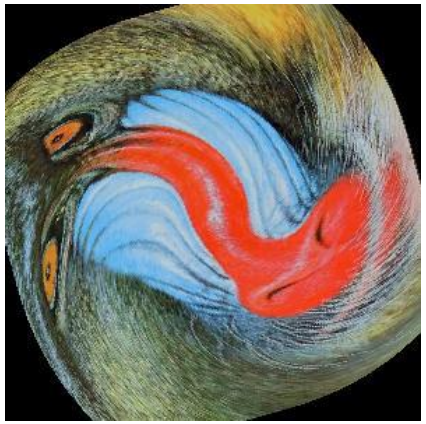
جداسازی مؤلفه‌های RGB



فیلتر لبه سوبل

فیلتر شیشه

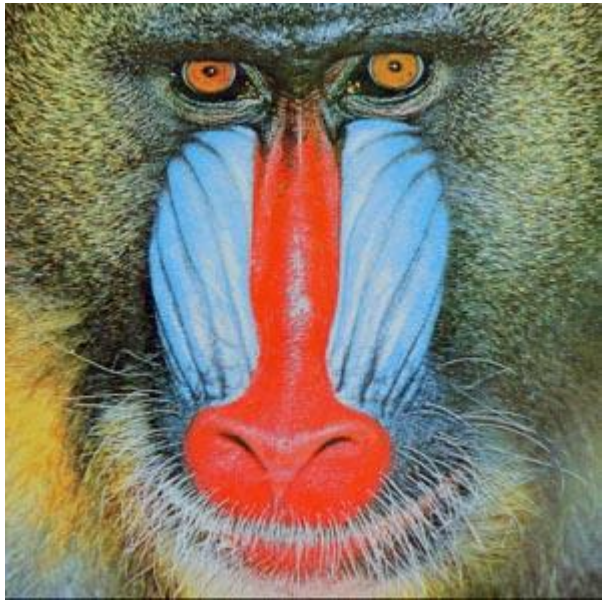
فیلتر موج



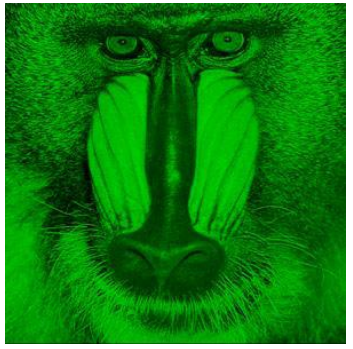
فیلتر گرداب

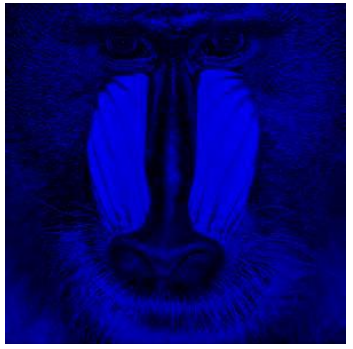
فیلتر جداسازی رنگ‌ها


```
import sys, stddraw
from picture import Picture
from color import Color

def main():

    source = Picture(sys.argv[1])
    w = source.width()
    h = source.height()

    R = Picture(w, h)
    G = Picture(w, h)
    B = Picture(w, h)

    for col in range(w):
        for row in range(h):

            pixel = source.get(col, row)
            r = pixel.getRed()
            g = pixel.getGreen()
            b = pixel.getBlue()

            R.set(col, row, Color(r, 0, 0))
            G.set(col, row, Color(0, g, 0))
            B.set(col, row, Color(0, 0, b))

stddraw.setCanvasSize(w, h)
stddraw.picture(R)
stddraw.show(1000)
stddraw.picture(G)
stddraw.show(1000)
stddraw.picture(B)
stddraw.show()
```

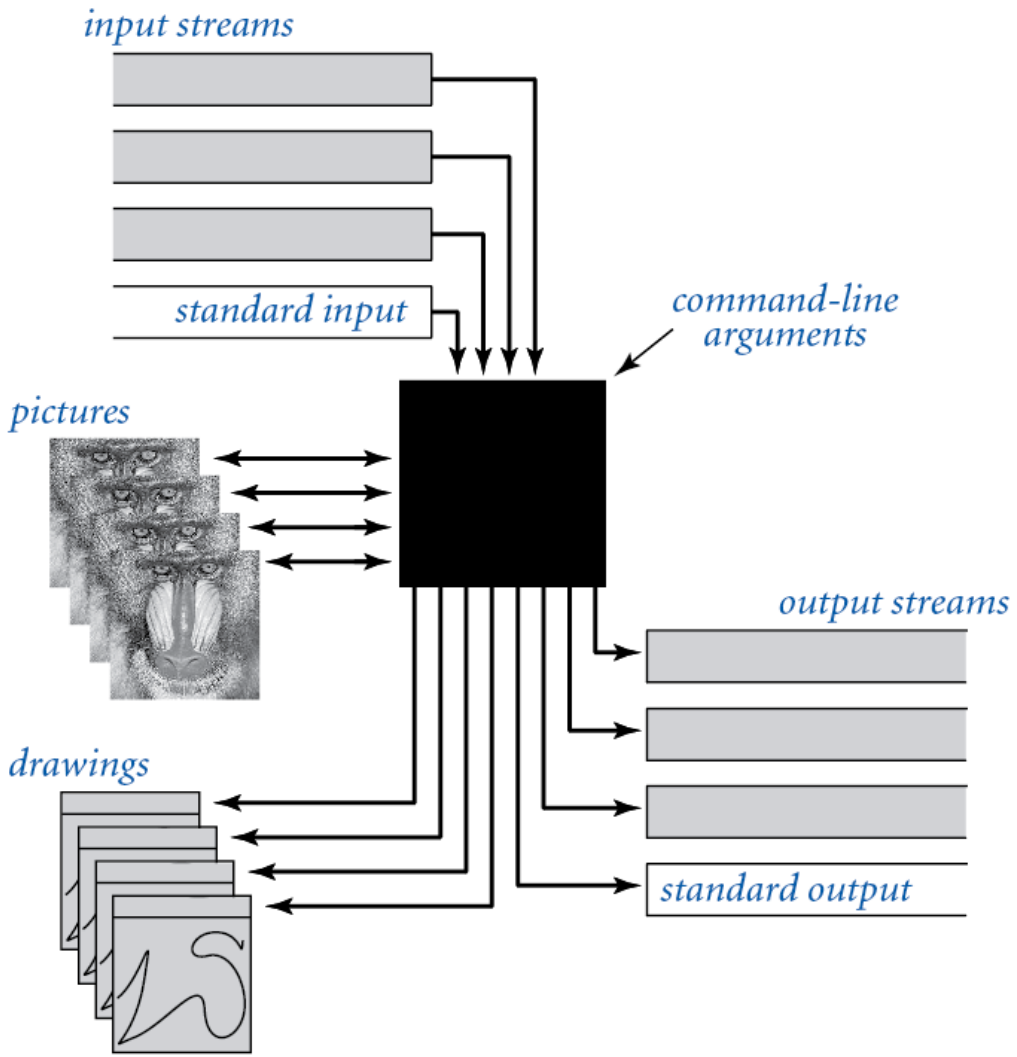


```
% python separatecolors.py mandrill.jpg
```





ورودی و خروجی



ورودی غیر استاندارد

- ورودی استاندارد. خواندن از پنجره ترمینال.
- هدف. خواندن از چندین جریان ورودی مختلف.
- نوع داده‌ای `InStream`. خواندن ورودی‌های متنی از یک فایل، یک وب‌سایت یا شبکه.
- مثال. آیا دو فایل متنی با هم برابر هستند؟

```
import sys
from instream import InStream

in0 = InStream(sys.argv[1])
in1 = InStream(sys.argv[2])
s = in0.readAll()
t = in1.readAll()
print(s == t)
```

خواندن از یک فایل

خواندن از یک فایل دیگر

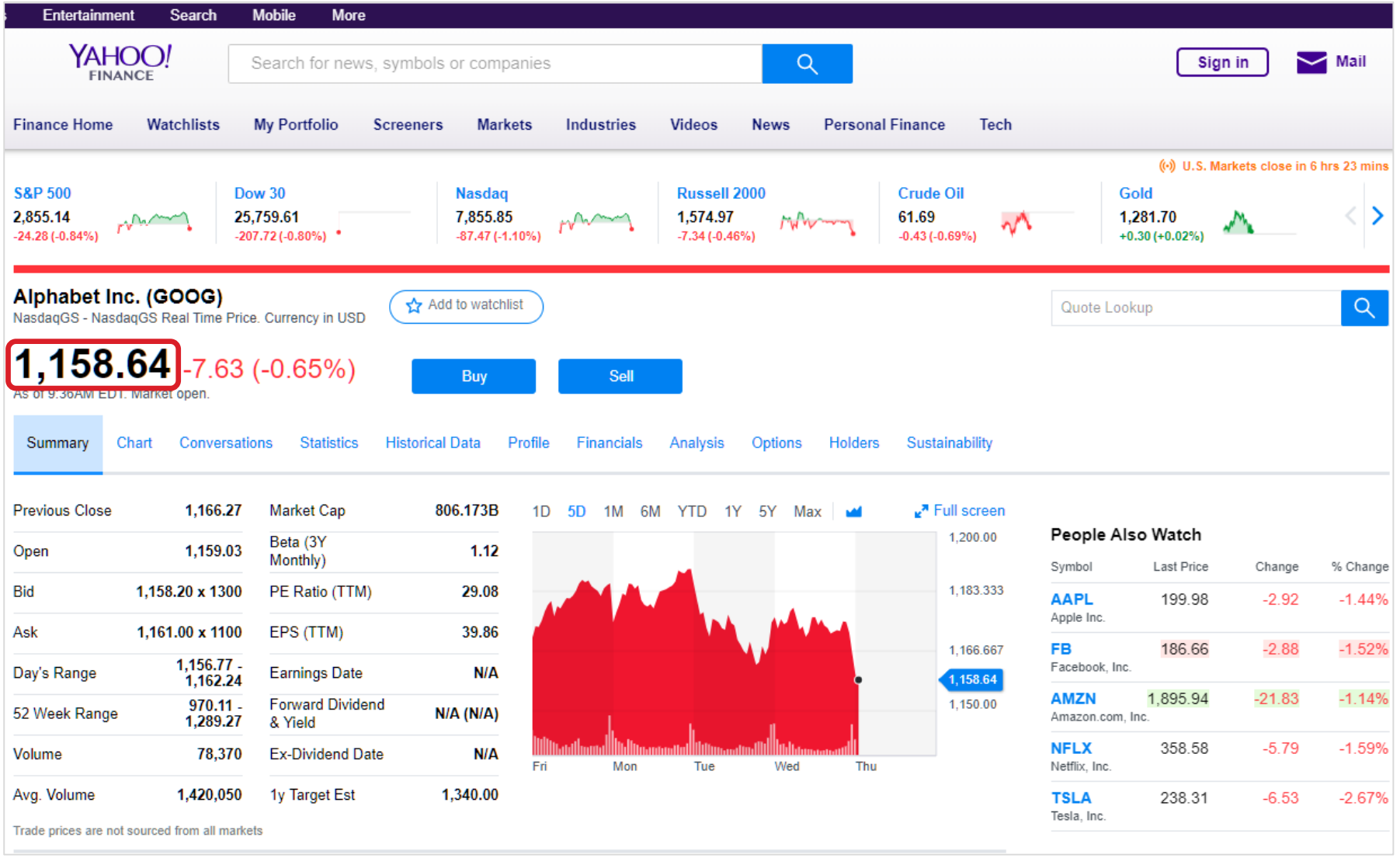
خروجی غیر استاندارد

- ورودی استاندارد. خواندن از پنجره ترمینال.
- هدف. خواندن از چندین جریان ورودی مختلف.
- نوع داده‌ای **OutStream**. نوشتن ورودی‌های متنی در یک فایل، یک وب‌سایت یا شبکه.
- مثال. الحاق چند فایل متنی در یک فایل.

```
import sys
from instream import InStream
from outstream import OutStream

inFilenames = sys.argv[1:-1]
outFilename = sys.argv[-1]
out = OutSream(outFilename)
for filename in inFilenames:
    in = InStream(filename)
    s = in.readAll()
    out.write(s)
```


```
% python cat.py in1.txt in2.txt out.txt
```

یافتن قیمت فعلی سهام گوگل



یافتن قیمت فعلی سهام گوگل

- هدف. یافتن قیمت فعلی سهام گوگل.
  - خواندن فایل HTML از آدرس https://finance.yahoo.com/quote/
  - یافتن قیمت پس از رشته Previous Close

```
def _readHTML(stockSymbol):
    WEBSITE = 'https://finance.yahoo.com/quote/'
    page = InStream(WEBSITE + stockSymbol + '/')
    html = page.readAll()
    return html

def priceOf(stockSymbol):
    html = _readHTML(stockSymbol)
    trade = html.find('Previous Close', 0)
    beg   = html.find('15">', trade)
    end   = html.find('</span>', beg)
    price = html[beg+1:end]
    price = price.replace(',', '')
    return float(price)
```

```
% python stockquote.py goog
1174.10

% python stockquote.py adbe
277.07
```

خرید و فروش سهام

- اندکی رنگ و لعاب.
  - نمودار قیمت را به صورت بلادرنگ ترسیم کنید.
  - اگر قیمت از یک حد مشخص پایین‌تر رفت، به کاربر خبر دهید.
  - کد خود را برای تعیین زمان مناسب خرید و فروش تعمیم دهید.
  - به صورت خودکار به شرکت مورد نظر سفارش‌های خرید و فروش ارسال کنید.

- هشدار! لطفاً، لطفاً، با مسؤولیت خود و پذیرفتن ریسک مالی اقدام به انجام این کار نمایید!

برنامه‌نویسی شی‌گرا: خلاصه

- شی. دربرگیرنده مقدار یک نوع داده‌ای؛ نام متغیر به شی ارجاع می‌دهد.

- جمع‌بندی. ما توانستیم برنامه‌هایی به منظور کار بر روی رنگ‌ها، تصاویر و رشته‌ها بنویسیم.

اولین گام در تسلط بر نامه‌نویسی شی‌گرا: کار با اشیای موجود

- جلسه آینده. نوشتن برنامه‌هایی به منظور کار بر روی اشیای تعریف شده به وسیله خود ما.

گام بعدی: تعریف اشیای جدید!