In [8]:
```
L = list()

def fx(a):
    return a+100

for var in [10,20,30,40,50]:
    r = fx(var)
    L.append(r)
L
```

Out[8]:
```
[110, 120, 130, 140, 150]
```

In [9]:
```
list(map(lambda a:a+100,[10,20,30,40,50]))
```

Out[9]:
```
[110, 120, 130, 140, 150]
```

In [10]:
```
d=dict()
d['K1']=list(map(lambda a:a+100,[10,20,30,40,50]))
d
```

Out[10]:
```
{'K1': [110, 120, 130, 140, 150]}
```

In [ ]:
```
Functional Style code - single line Code
lambda
list comprehension
generator

map,filter,enumerate .. - High order code
```

In [1]:
```
# function call with arguments and returns value
def fx(a,b):
    return a+b
```

In [2]:
```
fx(10,20)
```

Out[2]:
```
30
```

In [3]:
```
# lambda <list of args>:<operation>
lambda a,b:a+b
```

Out[3]:
```
<function __main__.<lambda>(a, b)>
```

In [4]:
```
fy = lambda a,b:a+b
fy(10,20)
```

Out[4]:
```
30
```

In [5]:
```
f1 = lambda a : a>100
f1(150)
```

Out[5]: True

In [6]:
```
f2 = lambda a: 'sales' in a
f2('raj,sales,hyd')
```

Out[6]: True

In [7]:
```
f3 = lambda a: a.upper()
f3('abc')
```

Out[7]: 'ABC'

In [11]:
```
def fx(a):
    if(a > 15):
        a = a+100
        return a
    else:
        a = a+200
        return a
```

In [12]:
```
f3 = lambda a:fx(a)
f3(50)
```

Out[12]: 150

In [13]:
```
# List comprehension  - list append operation

L = list()
for var in [10,20,30,40]:
    r = var+100
    L.append(r)
L
```

Out[13]: [110, 120, 130, 140]

In [14]:
```
# [value for <iterable>]
#          ---(1)---
#  --(2)---
[var+100 for var in [10,20,30,40]]
```

Out[14]: [110, 120, 130, 140]

In [15]:
```
L = list()
for var in [10,20,30,40]:
    if(var >30):
        r = var+100
        L.append(r)
    else:
        r = var + 500
        L.append(r)
L
```

Out[15]:
```
[510, 520, 530, 140]
```

In [16]:
```
[var+100 if var >30 else var+500 for var in [10,20,30,40]]
```

Out[16]:
```
[510, 520, 530, 140]
```

In [ ]:
```
# generator ->function ->returns an iterator(object) //generator
#                                   --------------- yield
```

In [18]:
```
def f1():
    return 10
def f2():
    yield 10

print(type(f1),type(f2))

print(type(f1()),type(f2()))
```

```
<class 'function'> <class 'function'>
<class 'int'> <class 'generator'>
```

In [19]:
```
def f2():
    n=15
    yield n
    yield n+100
    yield n+200
    print("OK-1")
    yield "D1","D2"
    yield "D1",["F1","F2","F3"],["F4","F5","F6"]
```

In [20]:
```
f2()
```

Out[20]:
```
<generator object f2 at 0x000002012F8A3ED0>
```

In [21]:
```
obj = f2()
```

In [25]:
```
obj = f2()
print(next(obj))
print(next(obj))
print(next(obj))
print(next(obj))
print(next(obj))
print(next(obj))
```

```
15
115
215
OK-1
('D1', 'D2')
('D1', ['F1', 'F2', 'F3'], ['F4', 'F5', 'F6'])

---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Cell In[25], line 7
      5 print(next(obj))
      6 print(next(obj))
----> 7 print(next(obj))

StopIteration:
```

In [26]:
```
obj = f2()
for var in obj:
    print(var)
```

```
15
115
215
OK-1
('D1', 'D2')
('D1', ['F1', 'F2', 'F3'], ['F4', 'F5', 'F6'])
```

In [27]:
```
obj = f2()
list(obj)
```

```
OK-1
```

Out[27]:
```
[15, 115, 215, ('D1', 'D2'), ('D1', ['F1', 'F2', 'F3'], ['F4', 'F5', 'F6'])]
```

In [28]:
```
fobj = open("E:\\emp.csv")
# fobj.read() / fobj.readlines()
for var in fobj:
    print(var.strip())
```

```
eid,ename,edept,eplace,ecost
101,raj,sales,pune,1000
102,leo,prod,bglore,2000
103,paul,HR,chennai,3000
104,anu,hr,hyderabad,4000
456,kumar,sales,bglore,3000
105,zion,Hr,mumbai,5000
106,bibu,sales,bglore,1450
107,theeb,sales,noida,4590
108,bibu,sales,bglore,5000
113,kumar,prod,hyderabad,5423DATA
```

In [30]:
```
d={}
d['CSV']=list(open("E:\\emp.csv"))
d
```

Out[30]:
```
{'CSV': ['eid,ename,edept,eplace,ecost\n',
  '101,raj,sales,pune,1000\n',
  '102,leo,prod,bglore,2000\n',
  '103,paul,HR,chennai,3000\n',
  '104,anu,hr,hyderabad,4000\n',
  '456,kumar,sales,bglore,3000\n',
  '105,zion,Hr,mumbai,5000\n',
  '106,bibu,sales,bglore,1450\n',
  '107,theeb,sales,noida,4590\n',
  '108,bibu,sales,bglore,5000\n',
  '113,kumar,prod,hyderabad,5423DATA']}
```

In [ ]:
```
map    ----> map(function,collection) -><generator>
             Vs
filter ---> filter(function,collection) --><generator>
```

In [31]:
```
map(lambda a:a+100,[10,20,30,40,50])
```

Out[31]:
```
<map at 0x2012f8da290>
```

In [32]:
```
obj = map(lambda a:a+100,[10,20,30,40,50])
for var in obj:
    print(var)
```

```
110
120
130
140
150
```

In [33]:
```
obj = map(lambda a:a+100,[10,20,30,40,50])
list(obj)
```

Out[33]:
```
[110, 120, 130, 140, 150]
```

In [35]:
```
obj = map(lambda a:a>25,[10,20,30,40,50])
list(obj)
```

Out[35]:
```
[False, False, True, True, True]
```

In [34]:
```
obj = filter(lambda a:a>25,[10,20,30,40,50])
list(obj)
```

Out[34]:
```
[30, 40, 50]
```

In [36]:
```
list(map(lambda a:a,open("E:\\emp.csv")))
```

Out[36]:
```
['eid,ename,edept,eplace,ecost\n',
 '101,raj,sales,pune,1000\n',
 '102,leo,prod,bglore,2000\n',
 '103,paul,HR,chennai,3000\n',
 '104,anu,hr,hyderabad,4000\n',
 '456,kumar,sales,bglore,3000\n',
 '105,zion,Hr,mumbai,5000\n',
 '106,bibu,sales,bglore,1450\n',
 '107,theeb,sales,noida,4590\n',
 '108,bibu,sales,bglore,5000\n',
 '113,kumar,prod,hyderabad,5423DATA']
```

In [37]:
```
list(filter(lambda a:'sales' in a,open('E:\\emp.csv')))
```

Out[37]:
```
['101,raj,sales,pune,1000\n',
 '456,kumar,sales,bglore,3000\n',
 '106,bibu,sales,bglore,1450\n',
 '107,theeb,sales,noida,4590\n',
 '108,bibu,sales,bglore,5000\n']
```

In [38]:
```
list(map(lambda a:a.strip(),list(filter(lambda a:'sales' in a,open('E:\\emp.csv
```

Out[38]:
```
['101,raj,sales,pune,1000',
 '456,kumar,sales,bglore,3000',
 '106,bibu,sales,bglore,1450',
 '107,theeb,sales,noida,4590',
 '108,bibu,sales,bglore,5000']
```

https://www.kdnuggets.com/2020/02/audio-file-processing-ecg-audio-python.html

In [40]:
```
#help(enumerate)
enumerate('python')
```

Out[40]:
```
<enumerate at 0x2012f8abce0>
```

In [41]:
```
for var in enumerate('python'):
    print(var)
```

```
(0, 'p')
(1, 'y')
(2, 't')
(3, 'h')
(4, 'o')
(5, 'n')
```

In [42]:
```
for var in enumerate({'K1':'V1','K2':'V2','K3':'V3'}):
    print(var)
```

```
(0, 'K1')
(1, 'K2')
(2, 'K3')
```

In [43]:
```
for var in enumerate('python',1):
    print(var)
```

```
(1, 'p')
(2, 'y')
(3, 't')
(4, 'h')
(5, 'o')
(6, 'n')
```

In [44]:
```
for var in enumerate('python',50):
    print(var)
```

```
(50, 'p')
(51, 'y')
(52, 't')
(53, 'h')
(54, 'o')
(55, 'n')
```

In [46]:
```
for var in enumerate(open('E:\\emp.csv'),1):
    print(var)
```

```
(1, 'eid,ename,edept,eplace,ecost\n')
(2, '101,raj,sales,pune,1000\n')
(3, '102,leo,prod,bglore,2000\n')
(4, '103,paul,HR,chennai,3000\n')
(5, '104,anu,hr,hyderabad,4000\n')
(6, '456,kumar,sales,bglore,3000\n')
(7, '105,zion,Hr,mumbai,5000\n')
(8, '106,bibu,sales,bglore,1450\n')
(9, '107,theeb,sales,noida,4590\n')
(10, '108,bibu,sales,bglore,5000\n')
(11, '113,kumar,prod,hyderabad,5423DATA')
```

In [48]:
```
for var in enumerate(open('E:\\emp.csv'),1):
    line,data = var
    if(line >5 and line <12):
        print(line,data.strip())
```

```
6 456,kumar,sales,bglore,3000
7 105,zion,Hr,mumbai,5000
8 106,bibu,sales,bglore,1450
9 107,theeb,sales,noida,4590
10 108,bibu,sales,bglore,5000
11 113,kumar,prod,hyderabad,5423DATA
```

In [49]:
```
def fx(a):
    class cname:
        def __init__(self,a):
            self.a=a
        def method1(self):
            return str(self.a+100)
    obj = cname(a)
    return obj
```

In [50]:
```
import sqlite3
sqlite3.connect
```

Out[50]:
```
<function _sqlite3.connect>
```

In [51]:
```
import re
re.search
```

Out[51]:
```
<function re.search(pattern, string, flags=0)>
```

In [ ]:
```
re.search()
re.findall()
------------------//search

re.sub()
-----------//substitue

re.split()
----------------//formatted style of report
```

In [ ]:
```
re.search('pattern_string','input_string',re.I) ->><ack>/None
Vs
re.findall('pattern_string','input_string',re.I) ->[results]/[]
```

In [52]:
```
help(re.search)
```

```
Help on function search in module re:

search(pattern, string, flags=0)
    Scan through string looking for a match to the pattern, returning
    a Match object, or None if no match was found.
```

In [53]:
```
re.search('sales','101,raj,sales,pune,1000')
```

Out[53]:
```
<re.Match object; span=(8, 13), match='sales'>
```

In [54]:
```
bool(re.search('sales','101,raj,sales,pune,1000'))
```

Out[54]:
```
True
```

In [55]:
```
re.search('prod','101,raj,sales,pune,1000')
```

In [56]:
```
bool(re.search('prod','101,raj,sales,pune,1000'))
```

Out[56]:
```
False
```

In [58]:
```
re.search("sales","101,raj,SAles,pune,1000",re.I)
```

Out[58]:
```
<re.Match object; span=(8, 13), match='SAles'>
```

In [59]:
```
if(re.search('sales','101,raj,sales,pune,1000',re.I)):
    print('Yes - pattern is matched')
else:
    print('Not-matched')
```

```
Yes - pattern is matched
```

In [60]:
```
for var in open('E:\\emp.csv'):
    if(re.search('sales',var,re.I)):
        print(var.strip())
```

```
101,raj,sales,pune,1000
456,kumar,sales,bglore,3000
106,bibu,sales,bglore,1450
107,theeb,sales,noida,4590
108,bibu,sales,bglore,5000
```

In [ ]:
```
Basic Regular Expression (BRE) - Single pattern
Extended Regular Expression (ERE) - Multiple pattern

Basic Regular Expression (BRE) - Single pattern
-----------------------------------------------
^   ====>  ^pattern
$   ====>   pattern$
^pattern$ ============> pattern only
. (dot) ----->Match any Single Char except \n

.* ===>list of all

[] - character class -> [Aav]run ==> Arun arun vrun ; [Aa][Rr]un
                                                      ARun Arun aRun arun
[a-z] ->match any lowercase chars
[A-Z] ->match any uppercase chars
[a-zA-Z] ->match any alpha
[0-9] -->match any digits
[a-zA-Z0-9] ->match any alpha number

    ^[A-Za-z].*[a-z]$


^[] - starts with
[]$ - ends with

[^] ==============>Not matching a pattern
       [^a-z] => Not matching any lowercase  chars
       [^a-zA-Z] =>Not matching any alpha
       [^a-zA-Z0-9] =>Match space and specialchars

space - \s  => ^\s - line starts with space ; \s$ - ends with space

[0-9]  -> \d
[a-zA-Z0-9] --> \w

[^\w\s] --->match any specialchars <-- [^a-zA-Z0-9\s]

^$ ---> empty line

Extended Regular Expression - ERE
---------------------------------
| ()  + {}

pattern1|pattern2 ->Any one pattern is matched ; any order ->True
```

In [61]:
```
IP="127.0.0.1"
re.findall(".",IP)
```

Out[61]:
```
['1', '2', '7', '.', '0', '.', '0', '.', '1']
```

In [62]:
```
re.findall("\.",IP)
```

Out[62]:
```
['.', '.', '.']
```

In [63]:
```
re.findall('sales|DBA','101,raj,sales,pune')
```

Out[63]:
```
['sales']
```

In [ ]:
```
re.findall('^[a-zA-Z].*[0-9]$|^[A-Z].*[a-z]$|^\s.*[0-9]$',Input_string)
```

In [ ]:
```
pattern1 | pattern2  - like logical OR
---------------------------------------- any one pattern is matched anyorder
(pattern1)(pattern2) - like logical AND
........................................ both pattern should match - same order

<Pattern>+  <== 1 or more

ab+c
 |------->abc abbbbbbbbbbbbbbbbbbc //OK  abbbbbbbbbb,bbbbbbbbc //Not-matched
```

In [64]:
```
data="Sample REPORT Generated On 15th Sep 2024 Time:15:23:44:0T:42updated"
```

In [65]:
```
re.findall("[0-9]",data)
```

Out[65]:
```
['1', '5', '2', '0', '2', '4', '1', '5', '2', '3', '4', '4', '0', '4', '2']
```

In [66]:
```
re.findall("[0-9]+",data)
```

Out[66]:
```
['15', '2024', '15', '23', '44', '0', '42']
```

In [ ]:
```
^\s+ <== line begings with 1 or more space
```

In [ ]:
```
{} <== IRE - range based

<pattern>{n} - n times
ab{3}c ------------------>abbbc //OK

<pattern>{n,} - minimum n times
ab{3,}c ---> abbbc abbbbbbbbbbbbbbbbbbbbbbbbbbc //OK

ab+c  same as --> ab{1,}c

<pattern>{n,m} - minimum n times and maximum m times
ab{3,6}c -> abbbc abbbbc abbbbbc abbbbbbc //OK
```

In [ ]:
```
^[A-Za-z][A-Za-z][A-Za-z][0-9][0-9][0-9][0-9][0-9][a-z][a-z]$ <== BRE

^[A-Za-z]{3}[0-9]{5}[a-z]{2} <== ERE
```

In [ ]:
```
re.sub('oldPattern','replaceString','inputString') ->output_str

output_str
   |->replaced string if old pattern is matched
   |->inputString if old pattern is not matched
```

In [67]:
```
re.sub('sales','ADMIN','101,raj,sales,pune,1000')
```

Out[67]: '101,raj,ADMIN,pune,1000'

In [68]:
```
re.sub('sales','ADMIN','101,raj,prod,pune,1000')
```

Out[68]: '101,raj,prod,pune,1000'

In [69]:
```
re.subn('sales','ADMIN','101,raj,sales,pune')
```

Out[69]: ('101,raj,ADMIN,pune', 1)

In [70]:
```
re.sub('sales','ADMIN','sales,raj,sales,pune,sales')
```

Out[70]: 'ADMIN,raj,ADMIN,pune,ADMIN'

In [71]:
```
re.sub('sales','ADMIN','sales,raj,sales,pune,sales',1)
```

Out[71]: 'ADMIN,raj,sales,pune,sales'

In [72]:
```
re.sub('sales','ADMIN','sales,raj,sales',1,re.I)
```

Out[72]: 'ADMIN,raj,sales'

In [73]:
```
var="root:x:bin:bash:/root"

re.sub("bash","ksh",var)
```

Out[73]: 'root:x:bin:ksh:/root'

In [74]:
```
re.sub("bash","",var)
```

Out[74]: 'root:x:bin::/root'

In [75]:
```
re.sub("bash.","",var)
```

Out[75]: 'root:x:bin:/root'

In [76]:
```
L=['120GB','110GB','150Gb','Gb100','200']

for var in L:
    print(re.sub('[a-zA-Z]','',var))
```

```
120
110
150
100
200
```

In [80]:
```
s1="root:x:bin:bash"
s2="root:x,,bin(bash"
s1.split(":")
```

Out[80]:
```
['root', 'x', 'bin', 'bash']
```

In [78]:
```
re.split(":",s1)
```

Out[78]:
```
['root', 'x', 'bin', 'bash']
```

In [81]:
```
re.split("[^\w\s]",s2)
```

Out[81]:
```
['root', 'x', '', 'bin', 'bash']
```

In [84]:
```
re.split("[^\w\s]+",s2)
```

Out[84]:
```
['root', 'x', 'bin', 'bash']
```

In [85]:
```
d={'app':['flask','django','FastApi'],'service':['demo1','demo2','demo3']}
print(type(d))
```

```
<class 'dict'>
```

In [86]:
```
import json
json.dumps(d) # convert to json
```

Out[86]:
```
'{"app": ["flask", "django", "FastApi"], "service": ["demo1", "demo2", "demo
3"]}'
```

In [87]:
```
json.dumps(d,indent=2)
```

Out[87]:
```
'{\n  "app": [\n    "flask",\n    "django",\n    "FastApi"\n  ],\n  "servic
e": [\n    "demo1",\n    "demo2",\n    "demo3"\n  ]\n}'
```

In [88]:
```
print(json.dumps(d,indent=2))
```

```
{
  "app": [
    "flask",
    "django",
    "FastApi"
  ],
  "service": [
    "demo1",
    "demo2",
    "demo3"
  ]
}
```

In [89]:
```
wobj = open("E:\\test.json",'w')
json.dump(d,wobj)
wobj.close()
```

In [90]:
```
fobj = open('E:\\test.json')
pd = json.load(fobj)
fobj.close()
pd
```

Out[90]:
```
{'app': ['flask', 'django', 'FastApi'], 'service': ['demo1', 'demo2', 'demo
3']}
```

In [91]:
```
import requests
```

In [92]:
```
requests.get('https://api.github.com/users/hadley/orgs')
```

Out[92]:
```
<Response [200]>
```

In [93]:
```
r = requests.get('https://api.github.com/users/hadley/orgs')
r.headers
```

Out[93]:
```
{'Date': 'Thu, 24 Oct 2024 10:42:49 GMT', 'Content-Type': 'application/json;
charset=utf-8', 'Cache-Control': 'public, max-age=60, s-maxage=60', 'Vary':
'Accept,Accept-Encoding, Accept, X-Requested-With', 'ETag': 'W/"61f25e2a55611
f7eaf75a20ed5fe501245c5af99bc383854f5442975318fad26"', 'X-GitHub-Media-Type':
'github.v3; format=json', 'x-github-api-version-selected': '2022-11-28', 'Acc
ess-Control-Expose-Headers': 'ETag, Link, Location, Retry-After, X-GitHub-OT
P, X-RateLimit-Limit, X-RateLimit-Remaining, X-RateLimit-Used, X-RateLimit-Re
source, X-RateLimit-Reset, X-OAuth-Scopes, X-Accepted-OAuth-Scopes, X-Poll-In
terval, X-GitHub-Media-Type, X-GitHub-SSO, X-GitHub-Request-Id, Deprecation,
Sunset', 'Access-Control-Allow-Origin': '*', 'Strict-Transport-Security': 'ma
x-age=31536000; includeSubdomains; preload', 'X-Frame-Options': 'deny', 'X-Co
ntent-Type-Options': 'nosniff', 'X-XSS-Protection': '0', 'Referrer-Policy':
'origin-when-cross-origin, strict-origin-when-cross-origin', 'Content-Securit
y-Policy': "default-src 'none'", 'Content-Encoding': 'gzip', 'Server': 'githu
b.com', 'X-RateLimit-Limit': '60', 'X-RateLimit-Remaining': '59', 'X-RateLimi
t-Reset': '1729770169', 'X-RateLimit-Resource': 'core', 'X-RateLimit-Used':
'1', 'Accept-Ranges': 'bytes', 'Content-Length': '1008', 'X-GitHub-Request-I
d': 'D744:1DE9F7:63BA02:69C70B:671A24A9'}
```

In [94]:
```
r.headers['Content-Type']
```

Out[94]:
```
'application/json; charset=utf-8'
```

In [95]:
```
web_content = r.text
```

In [96]:
```
pd = json.loads(web_content)
print(type(pd))
```

```
<class 'list'>
```

In [97]:
```
print(type(pd[0]))
```

```
<class 'dict'>
```

In [98]:
```
pd[0]
```

Out[98]:
```
{'login': 'ggobi',
 'id': 423638,
 'node_id': 'MDEyOk9yZ2FuaXphdGlvbjQyMzYzOA==',
 'url': 'https://api.github.com/orgs/ggobi',
 'repos_url': 'https://api.github.com/orgs/ggobi/repos',
 'events_url': 'https://api.github.com/orgs/ggobi/events',
 'hooks_url': 'https://api.github.com/orgs/ggobi/hooks',
 'issues_url': 'https://api.github.com/orgs/ggobi/issues',
 'members_url': 'https://api.github.com/orgs/ggobi/members{/member}',
 'public_members_url': 'https://api.github.com/orgs/ggobi/public_members{/mem
ber}',
 'avatar_url': 'https://avatars.githubusercontent.com/u/423638?v=4',
 'description': ''}
```

In [99]:
```
r=requests.get('https://www.google.com')
r.headers['Content-Type']
```

Out[99]:
```
'text/html; charset=ISO-8859-1'
```

In [100]:
```
web_page=r.text
```

In [101]:
```
with open('E:\\test.html','w') as wobj:
    wobj.write(web_page)
```

In [ ]:
```
numpy    - computation
  |->index,reshape,slicing
  |->universal functions

pandas   - Data Analysis
matplotlib - Visualization
```

In [102]:
```
L=[10,20,30,40]
L+100
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[102], line 2
      1 L=[10,20,30,40]
----> 2 L+100

TypeError: can only concatenate list (not "int") to list
```

In [104]:
```
import array
#help(array.array)
arr = array.array('i')
arr
```

Out[104]:
```
array('i')
```

In [105]:
```
arr.append(10)
arr.append(20)
arr
```

Out[105]:
```
array('i', [10, 20])
```

In [106]:
```
arr.append(30.0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[106], line 1
----> 1 arr.append(30.0)

TypeError: 'float' object cannot be interpreted as an integer
```

In [107]:
```
arr+100
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[107], line 1
----> 1 arr+100

TypeError: can only append array (not "int") to array
```

In [108]:
```
import numpy
numpy.array([10,20,30])
```

Out[108]:
```
array([10, 20, 30])
```

In [109]:
```
arr = numpy.array([10,20,30])
arr
```

Out[109]:
```
array([10, 20, 30])
```

In [110]:
```
arr+100
```

Out[110]:
```
array([110, 120, 130])
```

In [111]:
```
arr=numpy.array([10,20,30])
arr.ndim
```

Out[111]:
```
1
```

In [112]:
```
arr.shape
```

Out[112]:
```
(3,)
```

In [113]:
```
arr=numpy.array([[10,20,30]])
print(arr.ndim,arr.shape)
```

```
2 (1, 3)
```

In [114]:
```
arr=numpy.array([[[10,20,30]]])
print(arr.ndim,arr.shape)
```

```
3 (1, 1, 3)
```

In [117]:
```
arr = numpy.array([[1,2,3],[4,5,6],[7,8,9]])
arr[1][2]
```

Out[117]:
```
6
```

In [118]:
```
arr[1,2]
```

Out[118]: 6