print(dir(str))

['\_\_add\_\_', '\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getitem\_\_', '\_\_getnewargs\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format\_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']

In [3]:

print(dir(int))

['\_\_abs\_\_', '\_\_add\_\_', '\_\_and\_\_', '\_\_bool\_\_', '\_\_ceil\_\_', '\_\_class\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_divmod\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_float\_\_', '\_\_floor\_\_', '\_\_floordiv\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getnewargs\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_index\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_int\_\_', '\_\_invert\_\_', '\_\_le\_\_', '\_\_lshift\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_neg\_\_', '\_\_new\_\_', '\_\_or\_\_', '\_\_pos\_\_', '\_\_pow\_\_', '\_\_radd\_\_', '\_\_rand\_\_', '\_\_rdivmod\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rfloordiv\_\_', '\_\_rlshift\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_ror\_\_', '\_\_round\_\_', '\_\_rpow\_\_', '\_\_rrshift\_\_', '\_\_rshift\_\_', '\_\_rsub\_\_', '\_\_rtruediv\_\_', '\_\_rxor\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_sub\_\_', '\_\_subclasshook\_\_', '\_\_truediv\_\_', '\_\_trunc\_\_', '\_\_xor\_\_', 'as\_integer\_ratio', 'bit\_length', 'conjugate', 'denominator', 'from\_bytes', 'imag', 'numerator', 'real', 'to\_bytes']

In [4]:

print(dir(float))

['\_\_abs\_\_', '\_\_add\_\_', '\_\_bool\_\_', '\_\_class\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_divmod\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_float\_\_', '\_\_floordiv\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getformat\_\_', '\_\_getnewargs\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_int\_\_', '\_\_le\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_neg\_\_', '\_\_new\_\_', '\_\_pos\_\_', '\_\_pow\_\_', '\_\_radd\_\_', '\_\_rdivmod\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rfloordiv\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_round\_\_', '\_\_rpow\_\_', '\_\_rsub\_\_', '\_\_rtruediv\_\_', '\_\_set\_format\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_sub\_\_', '\_\_subclasshook\_\_', '\_\_truediv\_\_', '\_\_trunc\_\_', 'as\_integer\_ratio', 'conjugate', 'fromhex', 'hex', 'imag', 'is\_integer', 'real']

### String

In [7]:

print(dir(str))

['\_\_add\_\_', '\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getitem\_\_', '\_\_getnewargs\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format\_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']

In [10]:

s**=**"python programming"

s**.**capitalize() **\***makes first letter capital

Out[10]:

'Python programming'

In [21]:

s1**=**"PYTHON"

s2**=**"python"

s1**.**casefold() **\***converts to lowercase

Out[21]:

'python'

In [23]:

s1**.**lower() **\***converts to lowercase

Out[23]:

'python'

In [19]:

s2**.**upper() **\***converts to uppercase

Out[19]:

'PYTHON'

In [26]:

s3**=**"PrAthyusha" **\***converts lower to upper **and** upper to lower

s3**.**swapcase()

Out[26]:

'pRaTHYUSHA'

In [28]:

s**=**"Python programmming 123 #$@"

print(s)

Python programmming 123 #$@

In [31]:

p**=**"123"

p**.**isdigit()

Out[31]:

True

In [32]:

s**.**isdigit()

Out[32]:

False

In [33]:

s**.**isnumeric()

Out[33]:

False

In [36]:

print(s**.**isdigit())

False

In [38]:

r**=**"pythonprogramming" **\***space **is** **not** treated **as** alphabet

r**.**isalpha()

Out[38]:

True

In [40]:

q**=**"python programming" **\***spaceis there

q**.**isalpha()

Out[40]:

False

In [47]:

a**=**"pythonprogramming123"**\***alphabets **and** numericals

a**.**isalnum()

Out[47]:

True

In [52]:

s**.**split() **\***split into list

Out[52]:

['Python', 'programmming', '123', '#$@']

In [54]:

s**.**split("o")

Out[54]:

['Pyth', 'n pr', 'grammming 123 #$@']

In [70]:

**for** i **in** s:

**if** i**.**isalnum():

print(i,end**=**" ")

P y t h o n p r o g r a m m m i n g 1 2 3

In [77]:

**for** i **in** s():

**if** i**.**isdigit():

print(i,end**=**" ")

**---------------------------------------------------------------------------**

**TypeError** Traceback (most recent call last)

**<ipython-input-77-bcacd1a9a99f>** in <module>

**----> 1 for** i **in** p**():**

2 **if** i**.**isdigit**():**

3 print**(**i**,**end**=" ")**

**TypeError**: 'str' object is not callable

In [80]:

a**=**" Python programming " **\***leftside space **is** removed

a**.**lstrip()

Out[80]:

'Python programming '

In [81]:

a**.**rstrip() **\***rightside space **is** removed

Out[81]:

' Python programming'

In [82]:

a**.**strip() **\***removes both the spaces

Out[82]:

'Python programming'

In [91]:

a**.**replace("p","#")

Out[91]:

' Python #rogramming '

In [92]:

a**.**replace(" ","\*")

Out[92]:

'\*\*\*\*\*\*Python\*programming\*\*\*\*'

In [97]:

s**=**"python programming"

"@"**.**join(s)

" ECE "**.**join(s)

Out[97]:

'p ECE y ECE t ECE h ECE o ECE n ECE ECE p ECE r ECE o ECE g ECE r ECE a ECE m ECE m ECE i ECE n ECE g'

In [105]:

print(s**.**count("mm"))

print(s**.**count("p"))

s**.**count("i")

1

2

Out[105]:

1

In [115]:

print(s**.**index("r"))

s**.**index("p")

8

Out[115]:

0

In [112]:

s**.**istitle()

Out[112]:

False

In [113]:

s**.**title()

Out[113]:

'Python Programming'

In [117]:

s**.**startswith("p")

Out[117]:

True

In [118]:

s**.**endswith("g")

Out[118]:

True

In [140]:

p**=**"prathyusha"

In [147]:

p**.**center(20," ")

Out[147]:

' prathyusha '

In [148]:

p**.**center(30,"a")

Out[148]:

'aaaaaaaaaaprathyushaaaaaaaaaaa'

In [158]:

s**=**"python python python programming"

**for** i **in** s:

**if** i**==**"p":

print(s**.**index("p"))

print(i,end**=**" ")

print(i**.**index("p"))

0

p 0

0

p 0

0

p 0

0

p 0

In [159]:

**for** i **in** range(len(s)):

**if** s[i]**==**"p":

print(i)

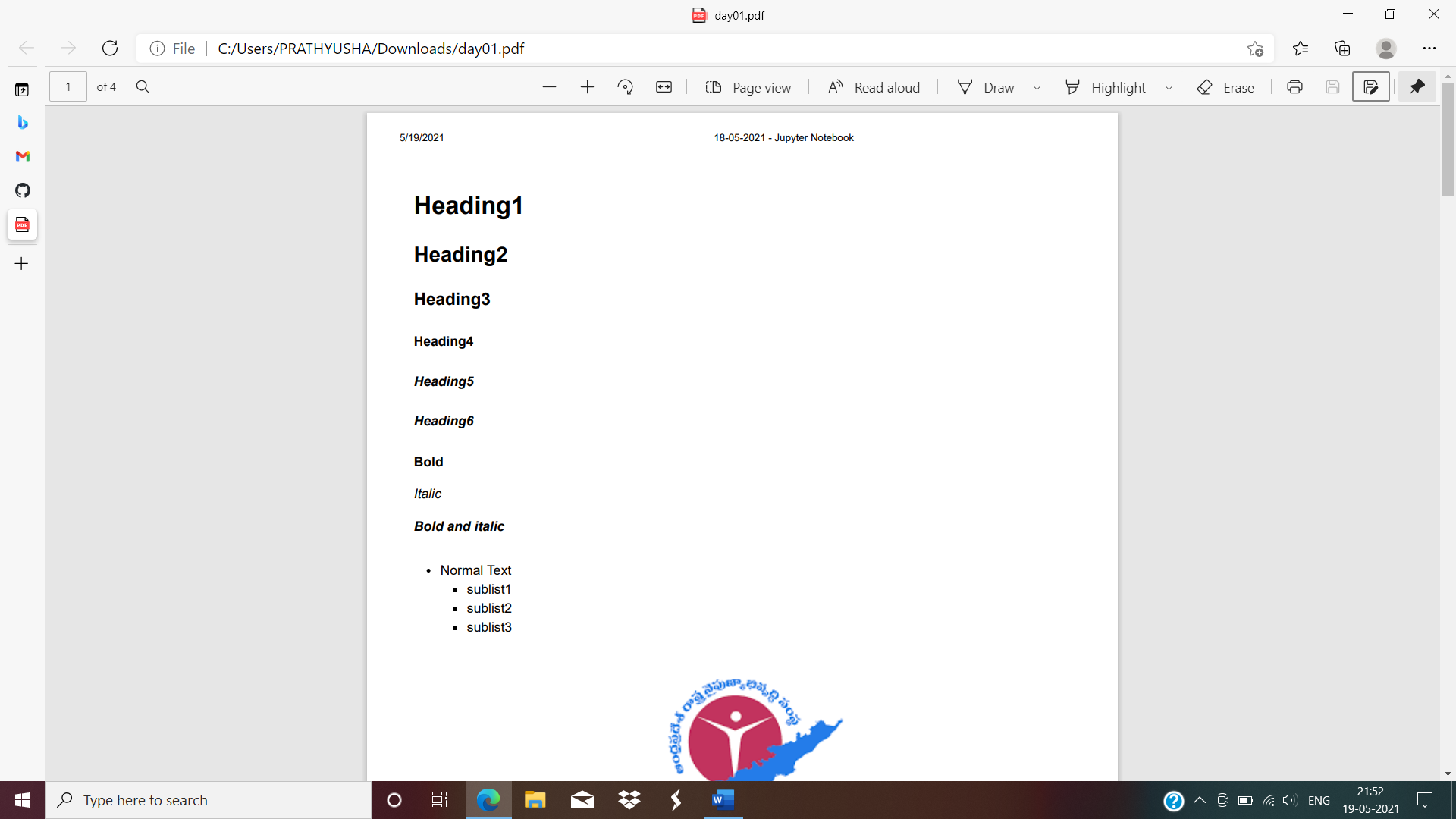
0

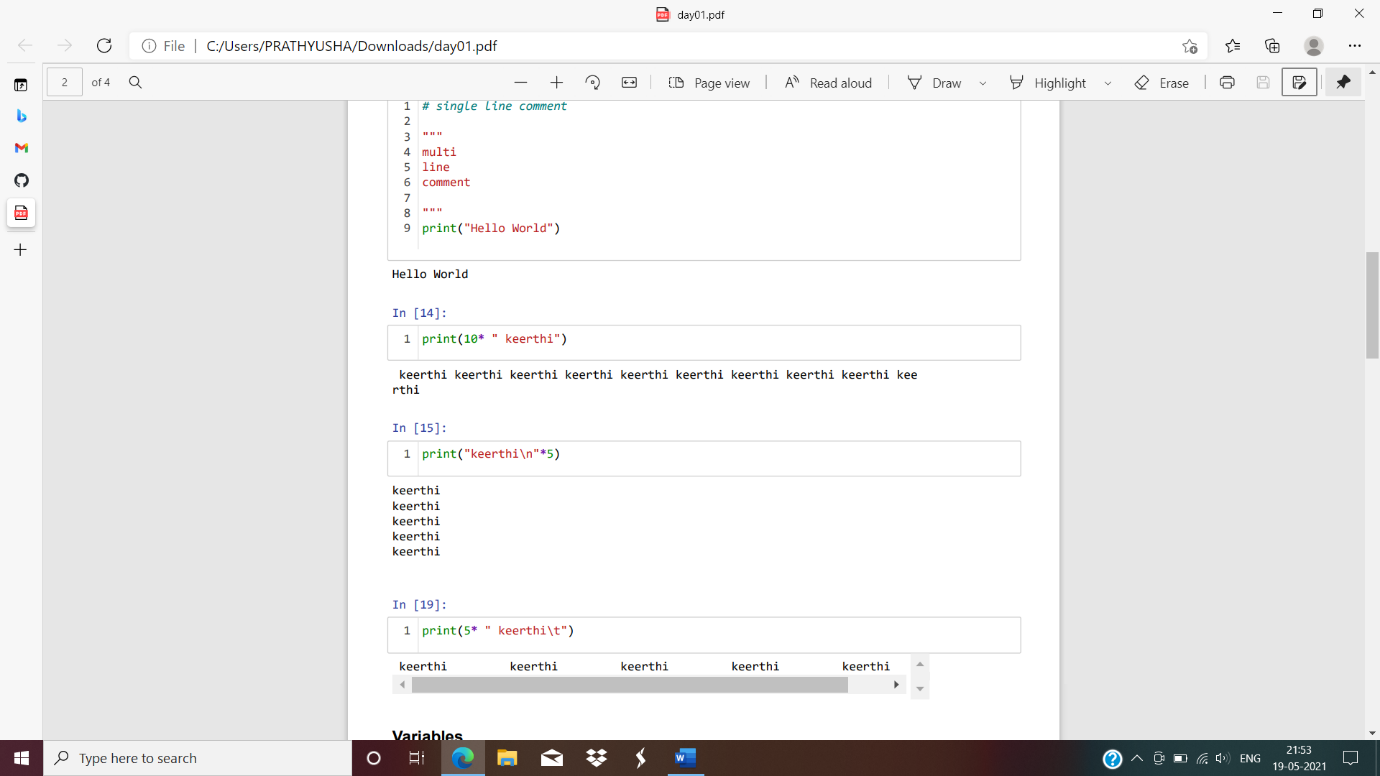
7

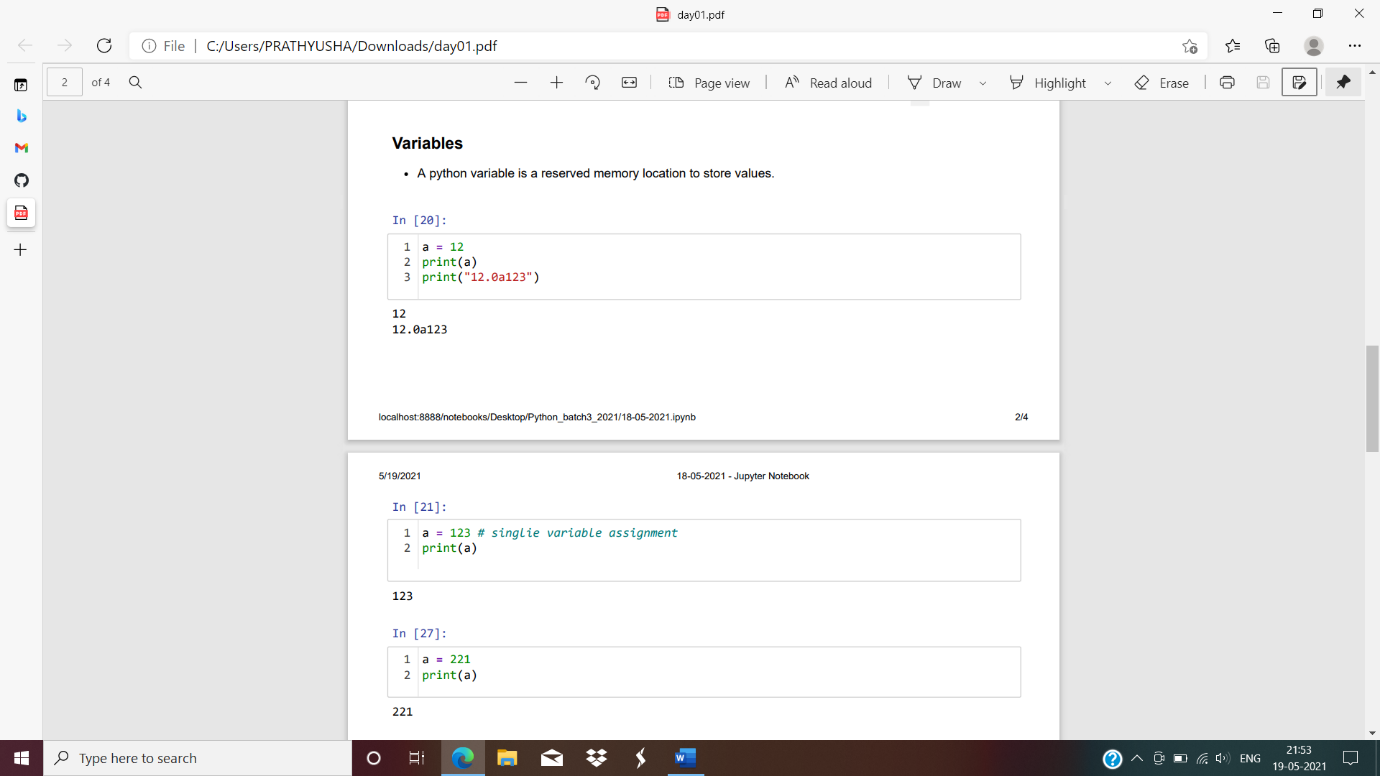
14

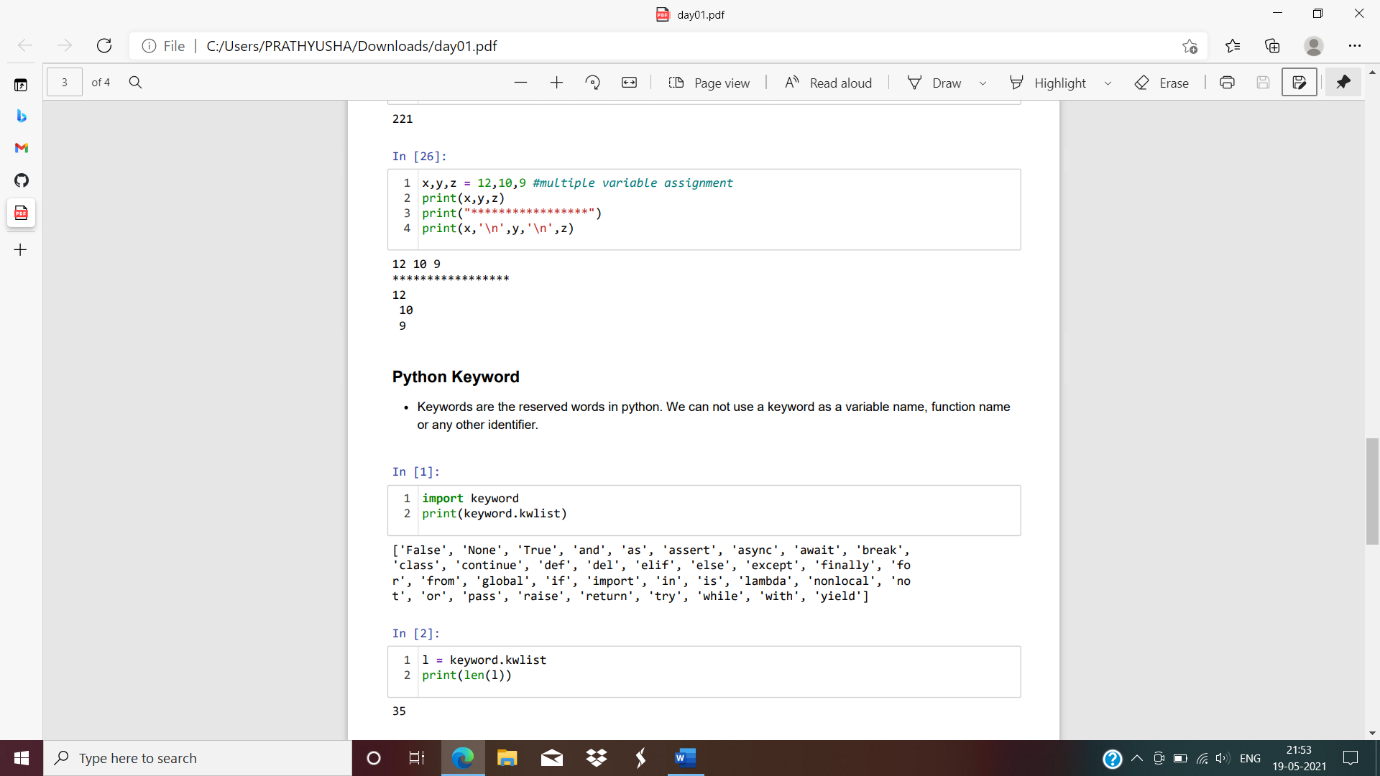
21

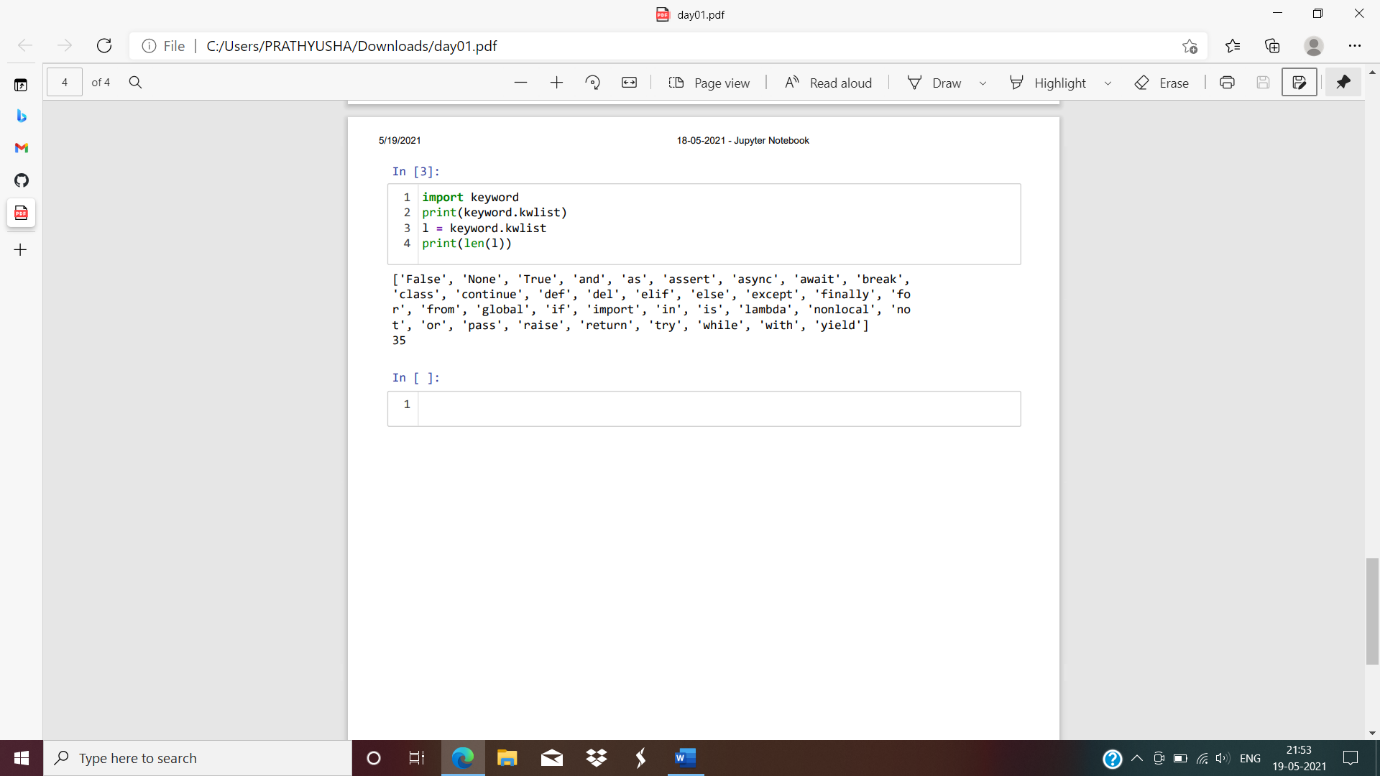
In [ ]:

****

****

****

****

****

**Data types and Conversions**

1.int 2.float 3.string

In [14]:

n1**=**20

print("n1=",n1)

type(n1)

n1= 20

Out[14]:

int

In [11]:

n2**=**13.56

type(n2)

Out[11]:

float

In [8]:

n3**=**"prathyusha"

type(n3)

Out[8]:

str

In [16]:

n**=**3

m**=**4

print(type(m))

print(type(n))

<class 'int'>

<class 'int'>

In [24]:

n**=**13

print(type(n))

print(type(str(n)))

<class 'int'>

<class 'str'>

In [25]:

n1**=**23

s**=**str(n1)

print(type(s))

<class 'str'>

num1=12 num2=15 print(num1,num2)

In [44]:

num1**=**18

num2**=**17

print(num1**+**num2)

35

In [29]:

s1**=**"swarna"

s2**=**"latha"

print(s1**+**s2)

swarnalatha

**Indentation**

In [43]:

n1,n2**=**13,12

**if**(n1**<**n2):

print("n1 greater than n2")

**else**:

print("wrong statement")

wrong statement

**Reading output dynamically**

In [49]:

**---------------------------------------------------------------------------**

**KeyboardInterrupt** Traceback (most recent call last)

**<ipython-input-49-dfc53c458c4a>** in <module>

**----> 1** n**=**input**()**

2 print**(**n**)**

3 print**(**type**(**x**))**

**C:\Anaconda\lib\site-packages\ipykernel\kernelbase.py** in raw\_input**(self, prompt)**

858 **"raw\_input was called, but this frontend does not support input requests."**

859 )

**--> 860 return self.\_input\_request(str(prompt),**

861 self**.**\_parent\_ident**,**

862 self**.**\_parent\_header**,**

**C:\Anaconda\lib\site-packages\ipykernel\kernelbase.py** in \_input\_request**(self, prompt, ident, parent, password)**

902 **except** KeyboardInterrupt**:**

903 **# re-raise KeyboardInterrupt, to truncate traceback**

**--> 904 raise** KeyboardInterrupt**("Interrupted by user")** **from** **None**

905 **except** Exception **as** e**:**

906 self**.**log**.**warning**("Invalid Message:",** exc\_info**=True)**

**KeyboardInterrupt**: Interrupted by user

In [50]:

x**=**input()

print(x)

print(type(x))

int

int

<class 'str'>

In [52]:

n**=**int(input("enter a value:"))

print(n)

print(type(n))

enter a value:15.5

**---------------------------------------------------------------------------**

**ValueError** Traceback (most recent call last)

**<ipython-input-52-bab35289e821>** in <module>

**----> 1** n**=**int**(**input**("enter a value:"))**

2 print**(**n**)**

3 print**(**type**(**n**))**

**ValueError**: invalid literal for int() with base 10: '15.5'

**Operators**

**Operators**

1.Arithmetic 2.Assignment 3.Comparision 4.Logical 5.Identity 6.Membership 7.Bitwise

**1.Arithmetic Operators**

-,+,\*,/,%,//,\*\*

In [60]:

a,b**=**15,8

print("a+b=",a**+**b)

print("a-b=",a**-**b)

print("a/b",a**/**b)

print("a\*b=",a**\***b)

print("a%b=",a**%b**)

print("a//b=",a**//**b)

print("a\*\*b=",a**\*\***b)

a+b= 23

a-b= 7

a/b 1.875

a\*b= 120

a%b= 7

a//b= 1

a\*\*b= 2562890625

**2.Assignment operator**

+,+=,-=,\*= etc....

In [68]:

a**=**5

print(a)

5

In [70]:

a **-=** 1

print(a)

4

In [71]:

a**-=**2

print(a)m**=**5

2

In [74]:

a**\*=**5

print(a)

10

**3.Comparision operators**

==,>,<,<=,>=,!=

In [78]:

n1,n2**=**5,2

print(n1**==**n2)

print(n1**>**n2)

print(n1**<=**n2)

print(n1**>=**n2)

print(n1**!=**n2)

False

True

False

True

True

**Logical operators**

and,not,or

In [85]:

a**=**5

print(a**<**6 **and** a**>**5)

print(a**<**6 **or** a**>**5)

False

True

In [89]:

res **=**a**<**6 **or** a**>**2

print(**not**(res))

False

In [90]:

a**=** 6 **or** 5

print(**not**(a))

False

**Identity operators**

is ,isnot

In [92]:

x,y**=**4,3

print(x **is** y)

False

In [93]:

x,y**=**4,4

print(x **is** y)

print(x **is** **not** y)

True

False

**6 . Membership Operators**

in, not in

In [95]:

fruits**=**"apple,mango,grapes"

print("apple"**in** fruits)

True

In [96]:

fruits**=**"apple,mango,grapes"

print("banana"**in** fruits)

False

In [99]:

fruits**=**"apple,mango,grapes"

print("apple" **not** **in** fruits)

print("banana"**not** **in** fruits)

False

Tru

**7.Bitwise operators**

&,|,^,>>,<<,~

In [106]:

print(5**&**3)

print(2**^**3)

print(5**>>**2)

print(6**<<**2)

print(6**|**2)

print(**~**2)

1

1

1

24

## static and dynamic

In [2]:

x**=**input()

print(x)

print(type(x))

123

123

<class 'str'>

In [4]:

a**=**1

print(type(a))

f**=**float(a)

print(type(f))

print(a)

print(f)

<class 'int'>

<class 'float'>

1

1.0

In [5]:

1

Out[5]:

1

## Conditional Statements

if elif else Nested if

In [9]:

a,b**=**1,2

**if**(a**>**b): *#colon we have to put after else or if*

print("a is greatest")

**else**:

print("b is greatest")

b is greatest

In [17]:

m**=**int(input("enter 1st number"))

n**=**int(input("enter 2nd number"))

0**=**int(input("enter 3rd number"))

**if**(m**>**n **and** m**>**o):

print("mis greatest")

**elif**(n**>**o):

print("n is greatest")

**else**:

print("o is greatest")

**File "<ipython-input-17-b4b49ab7d86c>", line 6**

**elif(n>o):**

**^**

**SyntaxError:** invalid syntax

In [ ]:

In [ ]:

In [ ]:

In [22]:

a**=**int(input("enter a mumber"))

**if**(a**%2**==0):

print("even")

**else**:

print("odd")

**File "<ipython-input-22-0b08a57f3c49>", line 4**

**else:**

**^**

**SyntaxError:** invalid syntax

In [ ]:

*## Nested if*

In [2]:

n**=**10

**if**(n**==**10):*#10==10*

**if**(n**%2**==0):

print(n,"n is even")

**if**(n**%2**!=0):

print(n,"n is odd")

**else**:

print(n,**not** equal to 10)

**File "<ipython-input-2-6f21823ec3d7>", line 8**

**else:**

**^**

**SyntaxError:** invalid syntax

In [ ]:

l**=**[1,2,3,4,5,6,4,5,6]

In [6]:

**for** i **in** l:

**if**(i**==**4):

print("4 is present")

4 is present

4 is present

In [7]:

**for** i **in** l:

**if**(i**==**4):

print("4 is present")

**break**

4 is present

In [12]:

**for** i **in** l:

**if**(i**==**4):

**continue**

print(i)

1

2

3

5

6

5

6

In [13]:

**for** i **in** l:

**if**(i**==**4):

print("4 is present")

**pass**

print(i)

1

2

3

4 is present

4

5

6

4 is present

4

5

6

## While loop

In [20]:

**for** i **in** range(1,11):

print("python",end**=**" ")

python python python python python python python python python python

syntax while codition: set of statements increment/decrement

In [5]:

n**=**1

**while**(n**<=**10):

print(n**\***n**\***n,end**=**" ")

n**=**n**+**1

1 8 27 64 125 216 343 512 729 1000

In [1]:

i**=**1

**while**(i**<=**6):

**if** (i**==**3):

**break**

print(i)

i**+=**1

1

2

In [6]:

**for** i **in** range(1,10):

print(i**\***i,end**=**" ")

1 4 9 16 25 36 49 64 81

In [17]:

n**=**int(input("enter a number"))

**while**(n**<=**50):

print(n,end**=**" ")

n**+=**1

print(n,end**=**" ")

n**+=**2

enter a number30

30 31 33 34 36 37 39 40 42 43 45 46 48 49

In [31]:

m**=**int(input("enter the amount in the account"))

n**=**int(input("enter the withdrawn amount"))

amount**=**m**-**1000

balance**=**m**-**n

**while**(**True**):

**if**(amount**>**n):

print("Take ur money")

**else**:

print("amount exceeded")

**break**

enter the amount in the account10000

enter the withdrawn amount8000

Take ur money

In [2]:

m**=**int(input())

n**=**int(input())

amount**=**m**-**1000

bal**=**m**-**n

print(amount)

print(bal)

5000

500

4000

4500

### Strings

In [14]:

s**=**'python programming 1234'

print(s)

python programming 1234

In [15]:

*###Positive isndexing*

*##negative indexing*

In [37]:

s**=**"Python programming 1234"

s[**-**18]

Out[37]:

'n'

### Slicing

varname[startindex: end index:step count]

In [39]:

s[19:22]

Out[39]:

'123'

In [45]:

s[4: :2]

Out[45]:

'o rgamn 24'

In [42]:

print(s)

Python programming 1234

In [51]:

print(s[0::5])

print(s[0::6])

Pngi2

P a

In [57]:

s[**-**6:**-**10:**-**1]

Out[57]:

'gnim'

In [58]:

s[**-**6:**-**2]

Out[58]:

'g 12'

In [60]:

s[::**-**1]

Out[60]:

'4321 gnimmargorp nohtyP'

In [61]:

s[::]

Out[61]:

'Python programming 1234'

In [67]:

print(len(s))

print(s)

len(s)

23

Python programming 1234

Out[67]:

23

In [73]:

print(s[len(s)**//**2])

print(len(s)**/**2)

r

11.5

In [74]:

m**=**4535567295747019

print(m**+**234)

4535567295747253

In [75]:

2**==**0

Out[75]:

False

In [ ]:

### String methods

In [1]:

*# Prime number*

*#17 (1,2,3.......17)*

*#17(2,3...............16)*

In [29]:

m**=**int(input("enter a number"))

count**=**0

**for** i **in** range(2,m**+**1):

**if**(m**%i**==0):

print(i)

enter a number100

2

4

5

10

20

25

50

100

In [41]:

m**=**int(input("enter a number"))

count**=**0

**for** i **in** range(2,m**+**1):

**if**(m**%i**==0):

count**=**count**+**1

print(count)

enter a number100

8

In [2]:

m**=**int(input("enter a number"))

**for** i **in** range(2,m**+**1):

**if**(m**%i**==0):

print("It is not a prime number")

**break**

**else**:

print("it is a prime number")

**break**

enter a number767

it is a prime number

In [4]:

m**=**int(input("enter a number"))

count**=**0

**for** i **in** range(1,m**+**1):

**if**(m**%i**==0):

count**+=**1

**if**(count**==**2):

print("n is a prime number")

**else**:

print("n is not a prime number")

enter a number100

n is not a prime number

In [6]:

enter a number0

enter m number12

**---------------------------------------------------------------------------**

**TypeError** Traceback (most recent call last)

**<ipython-input-6-194b75c15250>** in <module>

2 n**=**int**(**input**("enter m number"))**

3 **for** i **in** range**(**m**,**n**):**

**----> 4** i**[**m**]=0**

5 i**[**n**]=**i**[**m**+1]**

6 i**[**m**]=**i**[**n**]**

**TypeError**: 'int' object does not support item assignment

In [10]:

m**=**int(input())

n**=**int(input())

m**=**m**+**n

n**=**m**-**n

m**=**m**-**n

print("afrer swap=",m,n)

1

2

afrer swap= 2 1

In [11]:

**---------------------------------------------------------------------------**

**NameError** Traceback (most recent call last)

**<ipython-input-11-4bfafe6e2db6>** in <module>

1 m**=1.55**

**----> 2** ceil**(**m**)**

**NameError**: name 'ceil' is not defined

In [ ]:

print(dir(str))

['\_\_add\_\_', '\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getitem\_\_', '\_\_getnewargs\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format\_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']

In [3]:

print(dir(int))

['\_\_abs\_\_', '\_\_add\_\_', '\_\_and\_\_', '\_\_bool\_\_', '\_\_ceil\_\_', '\_\_class\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_divmod\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_float\_\_', '\_\_floor\_\_', '\_\_floordiv\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getnewargs\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_index\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_int\_\_', '\_\_invert\_\_', '\_\_le\_\_', '\_\_lshift\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_neg\_\_', '\_\_new\_\_', '\_\_or\_\_', '\_\_pos\_\_', '\_\_pow\_\_', '\_\_radd\_\_', '\_\_rand\_\_', '\_\_rdivmod\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rfloordiv\_\_', '\_\_rlshift\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_ror\_\_', '\_\_round\_\_', '\_\_rpow\_\_', '\_\_rrshift\_\_', '\_\_rshift\_\_', '\_\_rsub\_\_', '\_\_rtruediv\_\_', '\_\_rxor\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_sub\_\_', '\_\_subclasshook\_\_', '\_\_truediv\_\_', '\_\_trunc\_\_', '\_\_xor\_\_', 'as\_integer\_ratio', 'bit\_length', 'conjugate', 'denominator', 'from\_bytes', 'imag', 'numerator', 'real', 'to\_bytes']

In [4]:

print(dir(float))

['\_\_abs\_\_', '\_\_add\_\_', '\_\_bool\_\_', '\_\_class\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_divmod\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_float\_\_', '\_\_floordiv\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getformat\_\_', '\_\_getnewargs\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_int\_\_', '\_\_le\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_neg\_\_', '\_\_new\_\_', '\_\_pos\_\_', '\_\_pow\_\_', '\_\_radd\_\_', '\_\_rdivmod\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rfloordiv\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_round\_\_', '\_\_rpow\_\_', '\_\_rsub\_\_', '\_\_rtruediv\_\_', '\_\_set\_format\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_sub\_\_', '\_\_subclasshook\_\_', '\_\_truediv\_\_', '\_\_trunc\_\_', 'as\_integer\_ratio', 'conjugate', 'fromhex', 'hex', 'imag', 'is\_integer', 'real']

### String

In [7]:

print(dir(str))

['\_\_add\_\_', '\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getitem\_\_', '\_\_getnewargs\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format\_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']

In [10]:

s**=**"python programming"

s**.**capitalize() **\***makes first letter capital

Out[10]:

'Python programming'

In [21]:

s1**=**"PYTHON"

s2**=**"python"

s1**.**casefold() **\***converts to lowercase

Out[21]:

'python'

In [23]:

s1**.**lower() **\***converts to lowercase

Out[23]:

'python'

In [19]:

s2**.**upper() **\***converts to uppercase

Out[19]:

'PYTHON'

In [26]:

s3**=**"PrAthyusha" **\***converts lower to upper **and** upper to lower

s3**.**swapcase()

Out[26]:

'pRaTHYUSHA'

In [28]:

s**=**"Python programmming 123 #$@"

print(s)

Python programmming 123 #$@

In [31]:

p**=**"123"

p**.**isdigit()

Out[31]:

True

In [32]:

s**.**isdigit()

Out[32]:

False

In [33]:

s**.**isnumeric()

Out[33]:

False

In [36]:

print(s**.**isdigit())

False

In [38]:

r**=**"pythonprogramming" **\***space **is** **not** treated **as** alphabet

r**.**isalpha()

Out[38]:

True

In [40]:

q**=**"python programming" **\***spaceis there

q**.**isalpha()

Out[40]:

False

In [47]:

a**=**"pythonprogramming123"**\***alphabets **and** numericals

a**.**isalnum()

Out[47]:

True

In [52]:

s**.**split() **\***split into list

Out[52]:

['Python', 'programmming', '123', '#$@']

In [54]:

s**.**split("o")

Out[54]:

['Pyth', 'n pr', 'grammming 123 #$@']

In [70]:

**for** i **in** s:

**if** i**.**isalnum():

print(i,end**=**" ")

P y t h o n p r o g r a m m m i n g 1 2 3

In [77]:

**for** i **in** s():

**if** i**.**isdigit():

print(i,end**=**" ")

**---------------------------------------------------------------------------**

**TypeError** Traceback (most recent call last)

**<ipython-input-77-bcacd1a9a99f>** in <module>

**----> 1 for** i **in** p**():**

2 **if** i**.**isdigit**():**

3 print**(**i**,**end**=" ")**

**TypeError**: 'str' object is not callable

In [80]:

a**=**" Python programming " **\***leftside space **is** removed

a**.**lstrip()

Out[80]:

'Python programming '

In [81]:

a**.**rstrip() **\***rightside space **is** removed

Out[81]:

' Python programming'

In [82]:

a**.**strip() **\***removes both the spaces

Out[82]:

'Python programming'

In [91]:

a**.**replace("p","#")

Out[91]:

' Python #rogramming '

In [92]:

a**.**replace(" ","\*")

Out[92]:

'\*\*\*\*\*\*Python\*programming\*\*\*\*'

In [97]:

s**=**"python programming"

"@"**.**join(s)

" ECE "**.**join(s)

Out[97]:

'p ECE y ECE t ECE h ECE o ECE n ECE ECE p ECE r ECE o ECE g ECE r ECE a ECE m ECE m ECE i ECE n ECE g'

In [105]:

print(s**.**count("mm"))

print(s**.**count("p"))

s**.**count("i")

1

2

Out[105]:

1

In [115]:

print(s**.**index("r"))

s**.**index("p")

8

Out[115]:

0

In [112]:

s**.**istitle()

Out[112]:

False

In [113]:

s**.**title()

Out[113]:

'Python Programming'

In [117]:

s**.**startswith("p")

Out[117]:

True

In [118]:

s**.**endswith("g")

Out[118]:

True

In [140]:

p**=**"prathyusha"

In [147]:

p**.**center(20," ")

Out[147]:

' prathyusha '

In [148]:

p**.**center(30,"a")

Out[148]:

'aaaaaaaaaaprathyushaaaaaaaaaaa'

In [158]:

s**=**"python python python programming"

**for** i **in** s:

**if** i**==**"p":

print(s**.**index("p"))

print(i,end**=**" ")

print(i**.**index("p"))

0

p 0

0

p 0

0

p 0

0

p 0

In [159]:

**for** i **in** range(len(s)):

**if** s[i]**==**"p":

print(i)

0

7

14

21

In [ ]:

In [ ]:

### What is datastructures?

Datastructures are used to store a collection of related data. we have four built-in datastructures in python-list,tuple,dictionary and set.

### List

-It is a collection of different datastructures. -It is mutable(changeable) -Allow duplicate members. -In python lidtd are written with square brackets.

In [2]:

lst**=**["spmvv",12,23,30.8,"apsssdc",20]

print(lst)

['spmvv', 12, 23, 30.8, 'apsssdc', 20]

In [6]:

print(len(lst))

6

In [13]:

print(lst[0])

print(lst[5])

spmvv

20

In [14]:

*#Indexing or Slicing*

print(lst[0:5])

['spmvv', 12, 23, 30.8, 'apsssdc']

In [21]:

print(lst[0:6:2])

['spmvv', 23, 'apsssdc']

In [23]:

print(lst[1:3])

[12, 23]

In [30]:

print(lst[1:6:2])

[12, 30.8, 20]

In [36]:

lst2**=**["cse",5.8,14,18,[13,15.3,"ece"],'mech',9.0]

In [37]:

print(lst2)

['cse', 5.8, 14, 18, [13, 15.3, 'ece'], 'mech', 9.0]

In [38]:

print(len(lst2))

7

In [39]:

lst2[4]

Out[39]:

[13, 15.3, 'ece']

In [50]:

lst2[4][1]

Out[50]:

15.3

In [56]:

lst**=**[1,2,[3,4,5,[6,7,8,9],10,'cse'],9.8,13]

print(lst)

[1, 2, [3, 4, 5, [6, 7, 8, 9], 10, 'cse'], 9.8, 13]

In [57]:

print(len(lst))

5

In [58]:

lst[2:6]

Out[58]:

[[3, 4, 5, [6, 7, 8, 9], 10, 'cse'], 9.8, 13]

In [63]:

lst[2][3]

Out[63]:

[6, 7, 8, 9]

In [65]:

lst[2][3][2]

Out[65]:

8

In [68]:

print(len(lst))

5

prathyusha prathyusha prathyusha prathyusha prathyusha prathyusha prathyusha prathyusha prathyusha prathyusha prathyusha

In [69]:

print(dir(list))

['\_\_add\_\_', '\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_delitem\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getitem\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_iadd\_\_', '\_\_imul\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_reversed\_\_', '\_\_rmul\_\_', '\_\_setattr\_\_', '\_\_setitem\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', 'append', 'clear', 'copy', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse', 'sort']

In [93]:

s**=**["p","q","r","s","t","u","v","w"]

s[4]**=**"a"

In [94]:

print(s)

['p', 'q', 'r', 's', 'a', 'u', 'v', 'w']

In [95]:

s**.**append("a")

In [96]:

print(s)

['p', 'q', 'r', 's', 'a', 'u', 'v', 'w', 'a']

In [101]:

s**.**clear()

print(s)

[]

In [109]:

s**=**['q','w','e','r','t','y','u']

print(s)

['q', 'w', 'e', 'r', 't', 'y', 'u']

In [120]:

print(s[**-**5:**-**1:2])

['e', 't']

In [122]:

print(dir(list))

['\_\_add\_\_', '\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_delitem\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getitem\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_iadd\_\_', '\_\_imul\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_reversed\_\_', '\_\_rmul\_\_', '\_\_setattr\_\_', '\_\_setitem\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', 'append', 'clear', 'copy', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse', 'sort']

In [127]:

print(s)

['q', 'w', 'e', 'r', 't', 'y', 'u', 'q', 'q', 'w', 'a']

In [130]:

s**.**extend("prathyusha")

lst**.**extend([a,bprint(s)

['q', 'w', 'e', 'r', 't', 'y', 'u', 'q', 'q', 'w', 'a', 'p', 'r', 'a', 't', 'h', 'y', 'u', 's', 'h', 'a']

In [139]:

print()

**---------------------------------------------------------------------------**

**TypeError** Traceback (most recent call last)

**<ipython-input-139-bc7bea678d72>** in <module>

**----> 1** s**.**insert**('p')**

**TypeError**: insert expected 2 arguments, got 1

In [162]:

lst3**=**lst**.**copy() **\***copy

In [160]:

print(lst3)

[1, 2, [3, 4, 5, [6, 7, 8, 9], 10, 'cse'], 9.8, 13]

In [172]:

print(lst)

[1, 2, [3, 4, 5, [6, 7, 8, 9], 10, 'cse'], 9.8, 13, 'cse', 'cse', 'cse', 'cse', 'cse', 'prathyusha', 'prathyusha']

In [175]:

*#extend*

lst**.**extend(['a','b']) **\***In extend it takes **as** multiple

print(lst)

[1, 2, [3, 4, 5, [6, 7, 8, 9], 10, 'cse'], 9.8, 13, 'cse', 'cse', 'cse', 'cse', 'cse', 'prathyusha', 'prathyusha', 'a', 'b']

In [179]:

*#append*

lst**.**append(['1','2','3']) **\*in** append it takes **as** a single element

print(lst)

[1, 2, [3, 4, 5, [6, 7, 8, 9], 10, 'cse'], 9.8, 13, 'cse', 'cse', 'cse', 'cse', 'cse', 'prathyusha', 'prathyusha', 'a', 'b', ['1', '2', '3']]

In [180]:

*#index*

lst**.**index("cse")

Out[180]:

5

In [207]:

**for** i **in** range(len(lst)):

print(i,"=",lst[i])

0 = 1

1 = 2

2 = [3, 4, 5, [6, 7, 8, 9], 10, 'cse']

3 = 9.8

4 = 13

5 = cse

6 = cse

7 = cse

8 = cse

9 = cse

10 = prathyusha

11 = prathyusha

12 = a

13 = b

14 = ['1', '2', '3']

In [198]:

**for** i **in** range(len(lst)):

**if**(lst[i]**==**'cse'):

print(i)

5

6

7

8

9

In [ ]:

print(dir(list))

['\_\_add\_\_', '\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_delitem\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getitem\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_iadd\_\_', '\_\_imul\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_reversed\_\_', '\_\_rmul\_\_', '\_\_setattr\_\_', '\_\_setitem\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', 'append', 'clear', 'copy', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse', 'sort']

In [6]:

lst**=**["a","b","c","d","f","g","h"]

print(lst)

['a', 'b', 'c', 'd', 'f', 'g', 'h']

In [7]:

lst**.**remove("f")

print(lst)

['a', 'b', 'c', 'd', 'g', 'h']

In [12]:

lst**.**reverse()

print(lst)

['h', 'g', 'd', 'c', 'b', 'a']

In [13]:

lst**.**sort()

In [57]:

lst**=**["r","1","a","2","B","\*","A"]

print(lst)

['r', '1', 'a', '2', 'B', '\*', 'A']

In [58]:

lst**.**sort()

print(lst)

['\*', '1', '2', 'A', 'B', 'a', 'r']

In [59]:

lst**.**reverse()

print(lst)

['r', 'a', 'B', 'A', '2', '1', '\*']

In [60]:

lst**.**insert(2,"spmvv")

print(lst)

['r', 'a', 'spmvv', 'B', 'A', '2', '1', '\*']

In [61]:

lst**.**pop(2)

print(lst)

['r', 'a', 'B', 'A', '2', '1', '\*']

In [62]:

lst**.**clear()

print(lst)

[]

In [65]:

p**=**("a","p","r")

print(p)

('a', 'p', 'r')

In [71]:

print(p)

('a', 'p', 'r')

In [3]:

lst**=**[12,9.5,'cse','ece',15,8.5,3]

nlst**=**[]

slst**=**[]

flst**=**[]

**for** item **in** lst:

**if**(type(item)**==**int):

nlst**.**append(item)

**elif**(type(item)**==**(float)):

flst**.**append(item)

**else**:

slst**.**append(item)

print(nlst,'\n',slst,'\n',flst)

[12, 15, 3]

['cse', 'ece']

[9.5, 8.5]

**Tuple**

* A tuple is a collection which is ordered and immutable.
* In python tuples are written with round brackets.
* iteration in tuple is faster than list.

In [14]:

t**=**(12,9.5,'cse','ece',15,8.5,3)

print(len(t))

7

In [15]:

t[**-**1::**-**1]

Out[15]:

(3, 8.5, 15, 'ece', 'cse', 9.5, 12)

In [16]:

t['ece']**=**'ww'

**---------------------------------------------------------------------------**

**TypeError** Traceback (most recent call last)

**<ipython-input-16-41e687eb978e>** in <module>

**----> 1** t**['ece']='ww'**

**TypeError**: 'tuple' object does not support item assignment

In [17]:

**del**(t[2])

print(t)

**---------------------------------------------------------------------------**

**TypeError** Traceback (most recent call last)

**<ipython-input-17-2142a5d7178f>** in <module>

**----> 1 del(**t**[2])**

2 print**(**t**)**

**TypeError**: 'tuple' object doesn't support item deletion

In [29]:

tup**=**(1,2,3,(4,5,6,7,(8,9)),10,11)

tup

Out[29]:

(1, 2, 3, (4, 5, 6, 7, (8, 9)), 10, 11)

In [30]:

print(len(tup))

6

In [43]:

tup[3][4][1]

Out[43]:

9

In [44]:

t**.**count(3)

Out[44]:

1

In [55]:

a**=**(1,2,3,4)

a**.**index(2)

Out[55]:

1

In [ ]:

### Dictionaries

-It stores collection of various types of data. -Dictionaries are unchangeable. -Dictionaries have pair keys and values which separated with ":". -it is represented as flower brackets--> {}. -Keys are act as index of values in dictionary. -Keys in dictionary

In [11]:

dic**=**{"name":"prathyusha","id":221,"grade":"A"}

In [12]:

print(dic)

{'name': 'prathyusha', 'id': 221, 'grade': 'A'}

In [37]:

*#Mutable*

print(dic["name"])

print(dic["id"])

print(dic["grade"])

prathyusha

221

A

In [38]:

dic["name"]**=**"jaan"

dic **\***dictionary **is** mutable

Out[38]:

{'name': 'jaan', 'id': 221, 'grade': 'A'}

di

In [39]:

dic["name"]

Out[39]:

'jaan'

In [40]:

lst**=**{1,2,3,4,5}

lst

Out[40]:

{1, 2, 3, 4, 5}

In [48]:

print(dir(dict))

['\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_delitem\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getitem\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_reversed\_\_', '\_\_setattr\_\_', '\_\_setitem\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys', 'pop', 'popitem', 'setdefault', 'update', 'values']

In [49]:

*#Items #pair of key and value*

dic**.**items()

Out[49]:

dict\_items([('name', 'jaan'), ('id', 221), ('grade', 'A')])

In [50]:

dic**.**keys()

Out[50]:

dict\_keys(['name', 'id', 'grade'])

In [52]:

dic**.**values()

Out[52]:

dict\_values(['jaan', 221, 'A'])

In [59]:

*#update*

dic**.**update({"class":8,"section":"B"})

dic

Out[59]:

{'name': 'jaan', 'id': 221, 'grade': 'A', 'class': 8, 'section': 'B'}

In [74]:

*#pop*

dic**.**pop("name")

dic**.**pop("id")

**---------------------------------------------------------------------------**

**KeyError** Traceback (most recent call last)

**<ipython-input-74-def2b91cec48>** in <module>

1 **#pop**

**----> 2** dic**.**pop**("name")**

3 dic**.**pop**("id")**

**KeyError**: 'name'

In [73]:

dic

Out[73]:

{'grade': 'A', 'class': 8, 'section': 'B'}

In [75]:

dic**.**pop("class")

Out[75]:

8

In [77]:

dic**.**pop("section")

dic**.**pop("grade")

Out[77]:

'A'

In [79]:

dic

Out[79]:

{}

In [80]:

dic**=**{'name': 'jaan', 'id': 221, 'grade': 'A', 'class': 8, 'section': 'B'}

dic

Out[80]:

{'name': 'jaan', 'id': 221, 'grade': 'A', 'class': 8, 'section': 'B'}

In [81]:

*#popitem()*

dic**.**popitem() **\***last item gets deleted **in** popitem()

Out[81]:

('section', 'B')

In [82]:

dic

Out[82]:

{'name': 'jaan', 'id': 221, 'grade': 'A', 'class': 8}

In [91]:

*#setdefault()*

dic**.**setdefault("D")

dic

Out[91]:

{'name': 'jaan', 'id': 221, 'grade': 'A', 'class': 8, 'D': None}

In [94]:

dic**.**update({"D":"dell"})

dic

Out[94]:

{'name': 'jaan', 'id': 221, 'grade': 'A', 'class': 8, 'D': 'dell'}

In [97]:

dic**.**setdefault("A")

dic

Out[97]:

{'name': 'jaan', 'id': 221, 'grade': 'A', 'class': 8, 'D': 'dell', 'A': None}

In [101]:

dic**.**update()

dic

Out[101]:

{'name': 'jaan', 'id': 221, 'grade': 'A', 'class': 8, 'D': 'dell', 'A': None}

In [113]:

**File "<ipython-input-113-cc5ce43c9d79>", line 1**

**dic.(["A"])**

**^**

**SyntaxError:** invalid syntax

In [109]:

dic["A"]**=**"acer"

dic

Out[109]:

{'name': 'jaan', 'id': 221, 'grade': 'A', 'class': 8, 'D': 'dell', 'A': 'acer'}

In [115]:

dic**.**setdefault("L","Lenovo")

Out[115]:

'Lenovo'

In [116]:

dic

Out[116]:

{'name': 'jaan',

'id': 221,

'grade': 'A',

'class': 8,

'D': 'dell',

'A': 'acer',

'L': 'Lenovo'}

In [129]:

dic**.**setdefault("P","LLLL")

Out[129]:

'LLLL'

In [130]:

dic

Out[130]:

{'name': 'jaan',

'id': 221,

'grade': 'A',

'class': 8,

'D': 'dell',

'A': 'acer',

'L': 'Lenovo',

'l': 'LLLL',

'P': 'LLLL'}

In [131]:

*#get()*

dic**.**get("name")

Out[131]:

'jaan'

In [133]:

dic**.**get("id")

Out[133]:

221

In [137]:

dic**.**get

Out[137]:

'jaan'

In [150]:

dic2**=**dic**.**copy()

In [151]:

dic2

Out[151]:

{'name': 'jaan',

'id': 221,

'grade': 'A',

'class': 8,

'D': 'dell',

'A': 'acer',

'L': 'Lenovo',

'l': 'LLLL',

'P': 'LLLL'}

In [153]:

*#clear*

dic**.**clear()

dic

Out[153]:

{}

In [154]:

dic2

Out[154]:

{'name': 'jaan',

'id': 221,

'grade': 'A',

'class': 8,

'D': 'dell',

'A': 'acer',

'L': 'Lenovo',

'l': 'LLLL',

'P': 'LLLL'}

In [164]:

*#Fromkeys()*

x**=**("key1","key2","key3")

y**=**"a"

dict**.**fromkeys(x,y)

Out[164]:

{'key1': 'a', 'key2': 'a', 'key3': 'a'}

In [180]:

dict2**=**dic**.**fromkeys(x)

dict2

Out[180]:

{'key1': None, 'key2': None, 'key3': None}

In [173]:

dict2**=**dict**.**fromkeys(x,y)

dict2

Out[173]:

{'key1': 'a', 'key2': 'a', 'key3': 'a'}

In [181]:

dict2["key2"] **=** 12

dict2

Out[181]:

{'key1': None, 'key2': 12, 'key3': None}

In [198]:

dic1**=**{"student1":["a",100,"cse"],"student2":["b",121,"mech"]}

In [199]:

dic1

Out[199]:

{'student1': ['a', 100, 'cse'], 'student2': ['b', 121, 'mech']}

In [200]:

dic1**.**get("student1")

Out[200]:

['a', 100, 'cse']

In [201]:

dic1["student2"]

Out[201]:

['b', 121, 'mech']

In [203]:

dic1['student1'][2]

Out[203]:

'cse'

In [204]:

lst**=**[3,5,3,2,2,6,6,6,3]

In [206]:

lst

Out[206]:

[3, 5, 3, 2, 2, 6, 6, 6, 3]

In [214]:

dic5**.**update({3:2,5:1,2:2,6:3})

dic5

Out[214]:

{3: 2, 5: 1, 2: 2, 6: 3}

In [215]:

**---------------------------------------------------------------------------**

**TypeError** Traceback (most recent call last)

**<ipython-input-215-d07838f1b526>** in <module>

1 lst**=[3,5,3,2,2,6,6,6,3]**

**----> 2** dic**.**update**(**lst**)**

**TypeError**: cannot convert dictionary update sequence element #0 to a sequence

In [ ]:

lst**=**{lst**=**}

In [2]:

print(dir(dict))

['\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_delitem\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getitem\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_reversed\_\_', '\_\_setattr\_\_', '\_\_setitem\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys', 'pop', 'popitem', 'setdefault', 'update', 'values']

In [12]:

lst**=**[3,5,3,2,2,6,6,6,3,7]

dic**=**{}

**for** i **in** lst:

dic[i]**=**lst**.**count(i)

print(dic)

{3: 3, 5: 1, 2: 2, 6: 3, 7: 1}

In [31]:

lst**=**[3,5,3,2,2,6,6,6,3,7]

dic**=**{}

**for** i **in** lst:

**if** i **not** **in** dic:

dic[i]**=**1

**else**:

dic[i]**=**dic[i]**+**1

print(dic)

{3: 3, 5: 1, 2: 2, 6: 3, 7: 1}

In [47]:

dic[5]**=**1

dic

Out[47]:

{3: 3, 5: 1, 2: 2, 6: 3, 7: 1}

In [52]:

lst**=**[3,5,3,2,2,6,6,6,3,7]

folt(i,lst**.**count(i))

print(dic)

{3: 3, 5: 1, 2: 2, 6: 3, 7: 1}

### Sets

{} sets will remove the repitition\duplication

In [72]:

m**=**{1,2,3,4,2,1,2}

m

Out[72]:

{1, 2, 3, 4}

In [58]:

Out[58]:

set

In [68]:

Out[68]:

{1, 2, 3, 4}

### Sets

{} sets will remove the repitition\duplication

In [100]:

k**=**[5,1,2,3,1,2,3,5,6]

print(set(k))

list(set(k))

{1, 2, 3, 5, 6}

Out[100]:

[1, 2, 3, 5, 6]

In [101]:

type(k)

Out[101]:

list

In [109]:

print(k)

l1**=**[]

**for** i **in** k:

**if** i **not** **in** l1:

l1**.**append(i)

l1**.**sort()

print(l1)

len(l1)

[5, 1, 2, 3, 1, 2, 3, 5, 6]

[1, 2, 3, 5, 6]

Out[109]:

5

In [106]:

print(dir(set))

['\_\_and\_\_', '\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_iand\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_ior\_\_', '\_\_isub\_\_', '\_\_iter\_\_', '\_\_ixor\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_or\_\_', '\_\_rand\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_ror\_\_', '\_\_rsub\_\_', '\_\_rxor\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_sub\_\_', '\_\_subclasshook\_\_', '\_\_xor\_\_', 'add', 'clear', 'copy', 'difference', 'difference\_update', 'discard', 'intersection', 'intersection\_update', 'isdisjoint', 'issubset', 'issuperset', 'pop', 'remove', 'symmetric\_difference', 'symmetric\_difference\_update', 'union', 'update']

In [118]:

l**=**[1,2,"a",3,5,"a","6",'7',"b","c"]

In [119]:

set(l)

Out[119]:

{1, 2, 3, 5, '6', '7', 'a', 'b', 'c'}

In [155]:

s**=**{1,2,3,4,5}

s**.**add(10)

In [156]:

s

Out[156]:

{1, 2, 3, 4, 5, 10}

In [159]:

p**=**s**.**copy()

p

Out[159]:

{1, 2, 3, 4, 5, 10}

In [160]:

s\_copy**=**s**.**copy()

s\_copy

Out[160]:

{1, 2, 3, 4, 5, 10}

In [178]:

s1**=**{1,2,3,4,5}

s2**=**{4,5,6,3}

print(s1**.**difference(s2))

s2**.**difference(s1)

{1, 2}

Out[178]:

{6}

In [179]:

print(s1)

print(s2)

{1, 2, 3, 4, 5}

{3, 4, 5, 6}

In [181]:

print(s1)

print(s2)

print()

print(s1**.**difference\_update(s2))

print()

print(s1)

print(s2)

{1, 2, 3, 4, 5}

{3, 4, 5, 6}

None

{1, 2}

{3, 4, 5, 6}

In [183]:

s1**=**{1,2,3,6,7,8}

s2**=**{3,2,4,1,2,7}

s1**.**intersection(s2)

Out[183]:

{1, 2, 3, 7}

In [184]:

s1

Out[184]:

{1, 2, 3, 6, 7, 8}

In [185]:

s2

Out[185]:

{1, 2, 3, 4, 7}

In [189]:

s1**.**intersection\_update(s2)

In [190]:

s1

Out[190]:

{1, 2, 3, 7}

In [191]:

s2

Out[191]:

{1, 2, 3, 4, 7}

In [194]:

s1**=**{1,2,3,6,7,8}

s2**=**{3,2,4,1,2,7}

s1**.**isdisjoint(s2)

Out[194]:

False

In [196]:

s1**=**{1,2,3}

s2**=**{4,5,6}

s1**.**isdisjoint(s2)

Out[196]:

True

In [201]:

s1**=**{1,2,3,4,5}

s2**=**{1,2,3,4,5,6,7}

print(s2**.**issuperset(s1))

s1**.**issubset(s2)

True

Out[201]:

True

In [212]:

s1**=**{1,2,3,4,5,30,40,"q"}

s2**=**{1,2,3,4,5,6,7}

print(s2**.**difference(s1))

s2**.**symmetric\_difference(s1)

{6, 7}

Out[212]:

{30, 40, 6, 7, 'q'}

In [213]:

print(s1)

print(s2)

{1, 2, 3, 4, 5, 40, 'q', 30}

{1, 2, 3, 4, 5, 6, 7}

In [214]:

s1**.**union(s2)

Out[214]:

{1, 2, 3, 30, 4, 40, 5, 6, 7, 'q'}

In [215]:

s1

Out[215]:

{1, 2, 3, 30, 4, 40, 5, 'q'}

In [216]:

s2

Out[216]:

{1, 2, 3, 4, 5, 6, 7}

In [217]:

s1**.**update(s2) *# to join two sets*

s1

Out[217]:

{1, 2, 3, 30, 4, 40, 5, 6, 7, 'q'}

In [219]:

s1**.**pop()

Out[219]:

1

In [220]:

s1

Out[220]:

{2, 3, 30, 4, 40, 5, 6, 7, 'q'}

In [223]:

s1**.**pop() *#In sets first element gets removed*

s1

Out[223]:

{30, 4, 40, 5, 6, 7, 'q'}

In [224]:

s1**.**clear()

In [225]:

s1

Out[225]:

set()

In [233]:

s1**=**{1,2,3,4,5}

print(s1**.**discard(3))

None

In [234]:

s1**=**{1,2,3,4,5}

print(s1**.**discard(3))

None

In [236]:

s1**.**discard(2)

In [237]:

s1

Out[237]:

{1, 4, 5}

In [238]:

s1**.**remove(1)

In [239]:

s1

Out[239]:

{4, 5}

In [240]:

print(s1**.**remove(1))

**---------------------------------------------------------------------------**

**KeyError** Traceback (most recent call last)

**<ipython-input-240-22938132aa80>** in <module>

**----> 1** print**(**s1**.**remove**(1))**

**KeyError**: 1

In [241]:

print(s1**.**discard(1))

None

In [242]:

s1

Out[242]:

{4, 5}

In [244]:

s1**.**clear()

print(s1)

type(s)

set()

Out[244]:

set

In [249]:

p**=**{1,2,3,4,5}

**del** p

In [252]:

p **\***difference between delete **and** clear **is** In delete the entire data removes

**in** clear it removes the data but it defined **as** the set

**---------------------------------------------------------------------------**

**NameError** Traceback (most recent call last)

**<ipython-input-252-6c10289a8da7>** in <module>

**----> 1** p

**NameError**: name 'p' is not defined

In [ ]:

**REGULAR EXPRESSION**

* First we have to import regular expression package **re**
* \*\*re.methodname(pattern/pattern variable,string/string variable) ### Methods in re
* search()
* match()
* findall()

In [3]:

**import** math

math**.**sqrt(36)

Out[3]:

6.0

In [12]:

36**\*\***2

Out[12]:

1296

In [52]:

**import** re

print(re**.**search("SS","APSSDCSS"))

print(re**.**search("pyt","APSSDC"))

<re.Match object; span=(2, 4), match='SS'>

None

In [26]:

n**=**input()

print(re**.**search("@",n))

#212@

<re.Match object; span=(4, 5), match='@'>

In [53]:

*# Match method*

print(re**.**match("pss","Apssdcpss")) *#check character by character*

print(re**.**match("Apssd","Apssdc"))

None

<re.Match object; span=(0, 5), match='Apssd'>

In [51]:

print(re**.**findall("ss","Apssdcsasss"))

print(re**.**findall("s","apssdc"))

['ss', 'ss']

['s', 's']

In [56]:

print(re**.**search("@","apssdc@")) **\*** string takes through character throughout the string

print(re**.**match("@","apssdc@")) **\*** match takes **from** starting character throughout the string

<re.Match object; span=(6, 7), match='@'>

None

**Symbols in Re**

In [151]:

img src**=**"https://miro.medium.com/max/1200/1\*hjsbL45MhT2Tw5DGAYoAUg.png"

**File "<ipython-input-151-e9425ea71aba>", line 1**

**img src="https://miro.medium.com/max/1200/1\*hjsbL45MhT2Tw5DGAYoAUg.png"**

**^**

**SyntaxError:** invalid syntax

In [86]:

*#*

print(re**.**search(".","@apssdc"))

print(re**.**search("...","apssdc"))

print(re**.**search(".a",'apssdc'))

print(re**.**search("a","apssdc"))

<re.Match object; span=(0, 1), match='@'>

<re.Match object; span=(0, 3), match='aps'>

None

<re.Match object; span=(0, 1), match='a'>

In [63]:

*# '\*'Symbol*

print(re**.**search("^A","APSSDC"))

print(re**.**search("^AP","APSSDC"))

print(re**.**search("^AP","SAPssdc"))

<re.Match object; span=(0, 1), match='A'>

<re.Match object; span=(0, 2), match='AP'>

None

In [65]:

*# '$'Symbol*

print(re**.**search("DC$","APSSDC"))

print(re**.**search("C$","APSSDC"))

print(re**.**search("DC$","APSSDCS"))

print(re**.**match("DC$","APSSDCS"))

<re.Match object; span=(4, 6), match='DC'>

<re.Match object; span=(5, 6), match='C'>

None

None

In [92]:

*# '\*'Symbol*

print(re**.**search("A\*","AAAAAAAAAAAAAAAAAPSSDC"))

print(re**.**search("A\*","PSSDC"))

print(re**.**search("A","PSSDC"))

<re.Match object; span=(0, 17), match='AAAAAAAAAAAAAAAAA'>

<re.Match object; span=(0, 0), match=''>

None

In [97]:

*# '+'Symbol*

print(re**.**search("A+","AAAAAAAAAAAAAAAAAPSSDC"))

print(re**.**search("A+","PSSDC"))

print(re**.**search("A+","APSSDC"))

<re.Match object; span=(0, 17), match='AAAAAAAAAAAAAAAAA'>

None

<re.Match object; span=(0, 1), match='A'>

In [112]:

'{}'Symbol

print(re**.**search("A{1,3}","AAAAAAAAAAAAAAAAAPSSDC"))

print(re**.**search("P{2,5}","PPPSSDC"))

print(re**.**search("A{0,2}","APSSDC"))

print(re**.**search("SS{0,1}","APSSDC"))

<re.Match object; span=(0, 3), match='AAA'>

<re.Match object; span=(0, 3), match='PPP'>

<re.Match object; span=(0, 1), match='A'>

<re.Match object; span=(2, 4), match='SS'>

In [126]:

*#'[]'Symbol*

print(re**.**search("[ST]","APSSDC"))

print(re**.**search("[TSV]","PSSDC")) **\*** T **or** V **or** S

print(re**.**search("[Tv]","APSSDC"))

print(re**.**match("[ASP]","APSSDC"))

print(re**.**match("[APS]","APSSDC"))

print(re**.**match("TV","APSSDC"))

<re.Match object; span=(2, 3), match='S'>

<re.Match object; span=(1, 2), match='S'>

None

<re.Match object; span=(0, 1), match='A'>

<re.Match object; span=(0, 1), match='A'>

None

In [142]:

<re.Match object; span=(4, 5), match='1'>

<re.Match object; span=(5, 7), match='12'>

None

<re.Match object; span=(2, 3), match='P'>

None

### Pattern writing using regular expressions

##### Name Validation

* M Prathyusha
* prathyusha M
* M.Prathusha
* M Prathy Usha
* Prathy Usha Mamidi

In [1]:

[A**-**Z][A**-**Za**-**z]{0,10}[**.**]{0,10}[ ][A**-**Z][A**-**Za**-**z]{0,10}[ ]{0,1}[A**-**Z]{0,1}[A**-**Za**-**z]{0,8}

**File "<ipython-input-1-bfa3ad51b4e4>", line 1**

**[A-Z][A-Za-z]{0,10}[.]{0,10}[ ][A-Z][A-Za-z]{0,10}[ ]{0,1}[A-Z]{0,1}[A-Za-z]{0,8}**

**^**

**SyntaxError:** invalid syntax

In [7]:

**import** re

pattern**=**"[A-Z][A-Za-z]{0,10}[.]{0,10}[ ][A-Z][A-Za-z]{0,10}[ ]{0,1}[A-Z]{0,1}[A-Za-z]{0,8}"

n**=**input()

re**.**match(pattern,n)

prathyusha

In [6]:

**import** re

pattern**=**"[+]{0,1}[9][1][6-9][0-9]{9}|[0]{0,1}[6-9][0-9]{9}" **\*** 8500272314

n**=**input() **\*** 08500272314

re**.**match(pattern,n) **\*** **+**9185002723314

**\*** 918500272314

08694919885

Out[6]:

<re.Match object; span=(0, 11), match='08694919885'>

### Email Valiadtor

* prathyusha11123.m@ddjk.com
* all letter must be lowwercase alphabets===> 6 max 14
* contains some numbers===>1 to 6
* it may contain special characters(optional)===>[&,-,\_]
* after special character must contain numbers or digits
* must contains @
* contains some alphabets ===> length 4-8
* must contain .
* must contains some alphabets ===> length 2-4

### Comprehensions

**Comprehensions are used to reduce the number of lines of code**

### Types

* list comprehension
* set comprehension
* Dictionary comprehension

#### List comprehension

Syntax:

* if only if conditon is there===>[output loop condition]
* if contains both if and else ===>[output(if)condition loop]

In [15]:

l**=**[1,2,3,4,5,6,7,8,9]

l1**=**[] *# output : EVen numbers*

**for** i **in** l:

**if** i**%2**==0:

l1**.**append(i)

print("EVEN NUMBERS :",l1)

EVEN NUMBERS : [2, 4, 6, 8]

In [18]:

*### list comprehension*

S**=**[i **for** i **in** l **if** i**%2**==0]

print("EVEN NUMBERS",S)

EVEN NUMBERS [2, 4, 6, 8]

In [19]:

["EVEN" **if** i**%2**==0 else "odd" for i in l]

Out[19]:

['odd', 'EVEN', 'odd', 'EVEN', 'odd', 'EVEN', 'odd', 'EVEN', 'odd']

In [20]:

l**=**[1,2,3,4,5,6,"a","b"]

[i **for** i **in** l **if**(str(i)**.**isalpha())]

Out[20]:

['a', 'b']

In [ ]:

**Numpy**

* Numpy stands for Numerical python.
* Numpy is a python library used for working arrays.

In [9]:

**import** numpy **as** np

In [11]:

np**.**\_\_version\_\_ *#To see the version of nump*

Out[11]:

'1.20.1'

In [20]:

*# Create 1D array*

a1**=**np**.**array([1,2,3,4,5])

print(a1)

print(a1**.**ndim,"dimension")

[1 2 3 4 5]

1 dimension

In [60]:

*# Create 2D Array*

a2**=**np**.**array([[1,2,3],[4,5,6],[7,8,9],[4,1,2]])

print(a2)

print(a2**.**ndim,"dimension")

print(a2**.**shape)

print(a2**.**size)

print(a2**.**itemsize)

[[1 2 3]

[4 5 6]

[7 8 9]

[4 1 2]]

2 dimension

(4, 3)

12

4

In [48]:

a3**=**np**.**array([[[1,3,2],[1,3,2],[1,3,5]]])

print(a3)

print(a3**.**ndim)

print(a3**.**shape)

[[[1 3 2]

[1 3 2]

[1 3 5]]]

3

(1, 3, 3)

In [58]:

a4**=**np**.**array([[[1,"a"],[3,4]],[[4,5],[5,6]]])

print(a4)

print(a4**.**ndim,"dimension")

print(a4**.**shape)

print(a4**.**size)

print(a4**.**itemsize)

[[['1' 'a']

['3' '4']]

[['4' '5']

['5' '6']]]

3 dimension

(2, 2, 2)

8

44

**Creating an array using range()**

* np.array(range(start,end,step))
* np.array(range(start,end,step)).reshape(rows,columns)

In [63]:

q1**=**np**.**array(range(**-**11,10))

print(q1)

print(a1**.**ndim)

[-11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6

7 8 9]

1

In [76]:

q1**=**np**.**array(range(**-**11,10))**.**reshape(1,3,7)

q2**=**np**.**array(range(**-**1,11))**.**reshape(4,3)

print(q1)

print(q1**.**ndim)

print(q1**.**shape)

print(q1**.**size)

print(q1**.**itemsize)

print(q2)

print(q2**.**ndim)

print(q2**.**shape)

print(q2**.**size)

print(q2**.**itemsize)

[[[-11 -10 -9 -8 -7 -6 -5]

[ -4 -3 -2 -1 0 1 2]

[ 3 4 5 6 7 8 9]]]

3

(1, 3, 7)

21

4

[[-1 0 1]

[ 2 3 4]

[ 5 6 7]

[ 8 9 10]]

2

(4, 3)

12

4

**Create an array using range()**

* np.arrange(start,end.step).reshape(rows,columns)

In [84]:

q3**=**np**.**arange(1,40,2)**.**reshape(5,4)

print(q3)

print(q3**.**size)

print(q3**.**shape)

print(q3**.**ndim,"dimension")

[[ 1 3 5 7]

[ 9 11 13 15]

[17 19 21 23]

[25 27 29 31]

[33 35 37 39]]

20

(5, 4)

2 dimension

In [91]:

*# Zero matrix*

z**=**np**.**zeros((5,5))

print(z)

print(z[1][1])

print(type(z[1][1]))

[[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]]

0.0

<class 'numpy.float64'>

In [98]:

*# Zero matrix*

z**=**np**.**zeros((5,5),dtype**=**'int')

print(z)

print(z[1][1])

print(type(z[1][1]))

[[0 0 0 0 0]

[0 0 0 0 0]

[0 0 0 0 0]

[0 0 0 0 0]

[0 0 0 0 0]]

0

<class 'numpy.int32'>

In [103]:

*# Ones matrix*

z**=**np**.**ones((5,5),dtype**=**"int")

print(z)

print(type(z))

[[1 1 1 1 1]

[1 1 1 1 1]

[1 1 1 1 1]

[1 1 1 1 1]

[1 1 1 1 1]]

<class 'numpy.ndarray'>

In [107]:

print(z**-**5)

print(z**+**4)

print(z**\***5)

print(z**/**5)

print(z**%5**)

[[-4 -4 -4 -4 -4]

[-4 -4 -4 -4 -4]

[-4 -4 -4 -4 -4]

[-4 -4 -4 -4 -4]

[-4 -4 -4 -4 -4]]

[[5 5 5 5 5]

[5 5 5 5 5]

[5 5 5 5 5]

[5 5 5 5 5]

[5 5 5 5 5]]

[[5 5 5 5 5]

[5 5 5 5 5]

[5 5 5 5 5]

[5 5 5 5 5]

[5 5 5 5 5]]

[[0.2 0.2 0.2 0.2 0.2]

[0.2 0.2 0.2 0.2 0.2]

[0.2 0.2 0.2 0.2 0.2]

[0.2 0.2 0.2 0.2 0.2]

[0.2 0.2 0.2 0.2 0.2]]

[[1 1 1 1 1]

[1 1 1 1 1]

[1 1 1 1 1]

[1 1 1 1 1]

[1 1 1 1 1]]

In [109]:

print(dir(np))

['ALLOW\_THREADS', 'AxisError', 'BUFSIZE', 'Bytes0', 'CLIP', 'ComplexWarning', 'DataSource', 'Datetime64', 'ERR\_CALL', 'ERR\_DEFAULT', 'ERR\_IGNORE', 'ERR\_LOG', 'ERR\_PRINT', 'ERR\_RAISE', 'ERR\_WARN', 'FLOATING\_POINT\_SUPPORT', 'FPE\_DIVIDEBYZERO', 'FPE\_INVALID', 'FPE\_OVERFLOW', 'FPE\_UNDERFLOW', 'False\_', 'Inf', 'Infinity', 'MAXDIMS', 'MAY\_SHARE\_BOUNDS', 'MAY\_SHARE\_EXACT', 'MachAr', 'ModuleDeprecationWarning', 'NAN', 'NINF', 'NZERO', 'NaN', 'PINF', 'PZERO', 'RAISE', 'RankWarning', 'SHIFT\_DIVIDEBYZERO', 'SHIFT\_INVALID', 'SHIFT\_OVERFLOW', 'SHIFT\_UNDERFLOW', 'ScalarType', 'Str0', 'Tester', 'TooHardError', 'True\_', 'UFUNC\_BUFSIZE\_DEFAULT', 'UFUNC\_PYVALS\_NAME', 'Uint64', 'VisibleDeprecationWarning', 'WRAP', '\_NoValue', '\_UFUNC\_API', '\_\_NUMPY\_SETUP\_\_', '\_\_all\_\_', '\_\_builtins\_\_', '\_\_cached\_\_', '\_\_config\_\_', '\_\_deprecated\_attrs\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_expired\_functions\_\_', '\_\_file\_\_', '\_\_getattr\_\_', '\_\_git\_revision\_\_', '\_\_loader\_\_', '\_\_mkl\_version\_\_', '\_\_name\_\_', '\_\_package\_\_', '\_\_path\_\_', '\_\_spec\_\_', '\_\_version\_\_', '\_add\_newdoc\_ufunc', '\_distributor\_init', '\_financial\_names', '\_globals', '\_mat', '\_pytesttester', 'abs', 'absolute', 'add', 'add\_docstring', 'add\_newdoc', 'add\_newdoc\_ufunc', 'alen', 'all', 'allclose', 'alltrue', 'amax', 'amin', 'angle', 'any', 'append', 'apply\_along\_axis', 'apply\_over\_axes', 'arange', 'arccos', 'arccosh', 'arcsin', 'arcsinh', 'arctan', 'arctan2', 'arctanh', 'argmax', 'argmin', 'argpartition', 'argsort', 'argwhere', 'around', 'array', 'array2string', 'array\_equal', 'array\_equiv', 'array\_repr', 'array\_split', 'array\_str', 'asanyarray', 'asarray', 'asarray\_chkfinite', 'ascontiguousarray', 'asfarray', 'asfortranarray', 'asmatrix', 'asscalar', 'atleast\_1d', 'atleast\_2d', 'atleast\_3d', 'average', 'bartlett', 'base\_repr', 'binary\_repr', 'bincount', 'bitwise\_and', 'bitwise\_not', 'bitwise\_or', 'bitwise\_xor', 'blackman', 'block', 'bmat', 'bool8', 'bool\_', 'broadcast', 'broadcast\_arrays', 'broadcast\_shapes', 'broadcast\_to', 'busday\_count', 'busday\_offset', 'busdaycalendar', 'byte', 'byte\_bounds', 'bytes0', 'bytes\_', 'c\_', 'can\_cast', 'cast', 'cbrt', 'cdouble', 'ceil', 'cfloat', 'char', 'character', 'chararray', 'choose', 'clip', 'clongdouble', 'clongfloat', 'column\_stack', 'common\_type', 'compare\_chararrays', 'compat', 'complex128', 'complex64', 'complex\_', 'complexfloating', 'compress', 'concatenate', 'conj', 'conjugate', 'convolve', 'copy', 'copysign', 'copyto', 'core', 'corrcoef', 'correlate', 'cos', 'cosh', 'count\_nonzero', 'cov', 'cross', 'csingle', 'ctypeslib', 'cumprod', 'cumproduct', 'cumsum', 'datetime64', 'datetime\_as\_string', 'datetime\_data', 'deg2rad', 'degrees', 'delete', 'deprecate', 'deprecate\_with\_doc', 'diag', 'diag\_indices', 'diag\_indices\_from', 'diagflat', 'diagonal', 'diff', 'digitize', 'disp', 'divide', 'divmod', 'dot', 'double', 'dsplit', 'dstack', 'dtype', 'e', 'ediff1d', 'einsum', 'einsum\_path', 'emath', 'empty', 'empty\_like', 'equal', 'errstate', 'euler\_gamma', 'exp', 'exp2', 'expand\_dims', 'expm1', 'extract', 'eye', 'fabs', 'fastCopyAndTranspose', 'fft', 'fill\_diagonal', 'find\_common\_type', 'finfo', 'fix', 'flatiter', 'flatnonzero', 'flexible', 'flip', 'fliplr', 'flipud', 'float16', 'float32', 'float64', 'float\_', 'float\_power', 'floating', 'floor', 'floor\_divide', 'fmax', 'fmin', 'fmod', 'format\_float\_positional', 'format\_float\_scientific', 'format\_parser', 'frexp', 'frombuffer', 'fromfile', 'fromfunction', 'fromiter', 'frompyfunc', 'fromregex', 'fromstring', 'full', 'full\_like', 'gcd', 'generic', 'genfromtxt', 'geomspace', 'get\_array\_wrap', 'get\_include', 'get\_printoptions', 'getbufsize', 'geterr', 'geterrcall', 'geterrobj', 'gradient', 'greater', 'greater\_equal', 'half', 'hamming', 'hanning', 'heaviside', 'histogram', 'histogram2d', 'histogram\_bin\_edges', 'histogramdd', 'hsplit', 'hstack', 'hypot', 'i0', 'identity', 'iinfo', 'imag', 'in1d', 'index\_exp', 'indices', 'inexact', 'inf', 'info', 'infty', 'inner', 'insert', 'int0', 'int16', 'int32', 'int64', 'int8', 'int\_', 'intc', 'integer', 'interp', 'intersect1d', 'intp', 'invert', 'is\_busday', 'isclose', 'iscomplex', 'iscomplexobj', 'isfinite', 'isfortran', 'isin', 'isinf', 'isnan', 'isnat', 'isneginf', 'isposinf', 'isreal', 'isrealobj', 'isscalar', 'issctype', 'issubclass\_', 'issubdtype', 'issubsctype', 'iterable', 'ix\_', 'kaiser', 'kron', 'lcm', 'ldexp', 'left\_shift', 'less', 'less\_equal', 'lexsort', 'lib', 'linalg', 'linspace', 'little\_endian', 'load', 'loads', 'loadtxt', 'log', 'log10', 'log1p', 'log2', 'logaddexp', 'logaddexp2', 'logical\_and', 'logical\_not', 'logical\_or', 'logical\_xor', 'logspace', 'longcomplex', 'longdouble', 'longfloat', 'longlong', 'lookfor', 'ma', 'mafromtxt', 'mask\_indices', 'mat', 'math', 'matmul', 'matrix', 'matrixlib', 'max', 'maximum', 'maximum\_sctype', 'may\_share\_memory', 'mean', 'median', 'memmap', 'meshgrid', 'mgrid', 'min', 'min\_scalar\_type', 'minimum', 'mintypecode', 'mkl', 'mod', 'modf', 'moveaxis', 'msort', 'multiply', 'nan', 'nan\_to\_num', 'nanargmax', 'nanargmin', 'nancumprod', 'nancumsum', 'nanmax', 'nanmean', 'nanmedian', 'nanmin', 'nanpercentile', 'nanprod', 'nanquantile', 'nanstd', 'nansum', 'nanvar', 'nbytes', 'ndarray', 'ndenumerate', 'ndfromtxt', 'ndim', 'ndindex', 'nditer', 'negative', 'nested\_iters', 'newaxis', 'nextafter', 'nonzero', 'not\_equal', 'numarray', 'number', 'obj2sctype', 'object0', 'object\_', 'ogrid', 'oldnumeric', 'ones', 'ones\_like', 'os', 'outer', 'packbits', 'pad', 'partition', 'percentile', 'pi', 'piecewise', 'place', 'poly', 'poly1d', 'polyadd', 'polyder', 'polydiv', 'polyfit', 'polyint', 'polymul', 'polynomial', 'polysub', 'polyval', 'positive', 'power', 'printoptions', 'prod', 'product', 'promote\_types', 'ptp', 'put', 'put\_along\_axis', 'putmask', 'quantile', 'r\_', 'rad2deg', 'radians', 'random', 'ravel', 'ravel\_multi\_index', 'real', 'real\_if\_close', 'rec', 'recarray', 'recfromcsv', 'recfromtxt', 'reciprocal', 'record', 'remainder', 'repeat', 'require', 'reshape', 'resize', 'result\_type', 'right\_shift', 'rint', 'roll', 'rollaxis', 'roots', 'rot90', 'round', 'round\_', 'row\_stack', 's\_', 'safe\_eval', 'save', 'savetxt', 'savez', 'savez\_compressed', 'sctype2char', 'sctypeDict', 'sctypes', 'searchsorted', 'select', 'set\_numeric\_ops', 'set\_printoptions', 'set\_string\_function', 'setbufsize', 'setdiff1d', 'seterr', 'seterrcall', 'seterrobj', 'setxor1d', 'shape', 'shares\_memory', 'short', 'show\_config', 'sign', 'signbit', 'signedinteger', 'sin', 'sinc', 'single', 'singlecomplex', 'sinh', 'size', 'sometrue', 'sort', 'sort\_complex', 'source', 'spacing', 'split', 'sqrt', 'square', 'squeeze', 'stack', 'std', 'str0', 'str\_', 'string\_', 'subtract', 'sum', 'swapaxes', 'sys', 'take', 'take\_along\_axis', 'tan', 'tanh', 'tensordot', 'test', 'testing', 'tile', 'timedelta64', 'trace', 'tracemalloc\_domain', 'transpose', 'trapz', 'tri', 'tril', 'tril\_indices', 'tril\_indices\_from', 'trim\_zeros', 'triu', 'triu\_indices', 'triu\_indices\_from', 'true\_divide', 'trunc', 'typeDict', 'typecodes', 'typename', 'ubyte', 'ufunc', 'uint', 'uint0', 'uint16', 'uint32', 'uint64', 'uint8', 'uintc', 'uintp', 'ulonglong', 'unicode\_', 'union1d', 'unique', 'unpackbits', 'unravel\_index', 'unsignedinteger', 'unwrap', 'use\_hugepage', 'ushort', 'vander', 'var', 'vdot', 'vectorize', 'version', 'void', 'void0', 'vsplit', 'vstack', 'warnings', 'where', 'who', 'zeros', 'zeros\_like']

In [123]:

r**=**np**.**random**.**randint(10,20)

print(r)

15

In [128]:

r**=**np**.**random**.**randint(10,20,5)

r

Out[128]:

array([15, 11, 18, 19, 13])

In [141]:

r**=**np**.**random**.**randint(10,50,24)**.**reshape(2,3,2,2)

print(r)

print(r**.**ndim)

[[[[34 42]

[47 30]]

[[48 17]

[44 10]]

[[36 26]

[26 25]]]

[[[39 36]

[14 38]]

[[40 11]

[47 20]]

[[49 44]

[11 23]]]]

4

In [146]:

np**.**random**.**random()

Out[146]:

0.7238468363005259

In [147]:

np**.**random**.**random((2,3))

Out[147]:

array([[0.89059598, 0.15913787, 0.95788168],

[0.53414254, 0.46156296, 0.82017617]])

In [152]:

print(np**.**mean(r))

31.541666666666668

In [154]:

print(np**.**log(3))

1.0986122886681098

In [ ]: