python operators

operators are used to perform operations on variables and values

arthimetic operators

assignment operator

comparision operator

logical operator

bitwise operator

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [1]:

```
print(10**5)
```

```
100000
```

In [2]:

```
print(10-5)
```

```
5
```

In [3]:

```
print(10+5)
```

```
15
```

In [4]:

```
print(10/5)
```

```
2.0
```

In [5]:

```
x=4
x*5
print(x)
```

```
4
```

In [6]:

```
x=4
x*=5
print(x)
```

```
20
```

In [7]:

```
x=4
x/=5
print(x)
```

```
0.8
```

In [8]:

```
x=80
y=90
if(x==y):
    print("yes")
    else:
        print("no")
```

```
  File "<ipython-input-8-2b322950c318>", line 5
    else:
    ^
SyntaxError: invalid syntax
```

In [9]:

```
x=80
y=90
if(x==y):
    print("yes")
else:
        print("no")
```

```
no
```

comparision operator

In [ ]:

```
# == equal to
# !=equal to
# > greater than
# < lessthan
# >=greater than equal to
# <= lessthan equal to
```

In [10]:

```
x=5
y=3
print(x>=y)
```

```
True
```

In [11]:

```
x=5
y=3
print(x==y)
```

```
False
```

In [12]:

```
x=5
y=3
print(x!=y)
```

```
True
```

In [13]:

```
x=5
y=3
print(x>=y)
```

```
True
```

In [15]:

```
x=5
y=3
print(x>=y)
```

```
True
```

In [16]:

```
x=5
print (x>3 and x<10)
type(x)
```

```
True
```

Out[16]:

```
int
```

In [17]:

```
x=5
print (x>3 and x<10)
type(x)
```

```
True
```

Out[17]:

```
int
```

28/09/2022

In [ ]:

```
types of comments
1.single line comments
2. multi line comments

1.single line comments
>with the help of single line comments to display the title of page
>a single line comment denoted the symbol as #
syntax:
    # title of the page corresponding to markdown format.
    2.multiline comments:
        * a multi line comments to display the multiple lines of title to display the markd
        1st syntax:
           ''''''  ....................
            .........''''''
            2nd syntax
          """"""""  ....................
            .....................""""""""
```

python data types:

integer-int()

> it holds the integer values

string-str()

>it holds the string values

float-float()

>it holds thye floating type of data values

In [1]:

```
# convert the integer to string
m=1234
n=str(m)
type(n)
```

Out[1]:

```
str
```

In [3]:

```
# integer to float
h=9
print(type(h))
h1 =flaot(h)
print(h1)
print(type(h1))
```

```
<class 'int'>

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-3-3194c5d88181> in <module>
      2 h=9
      3 print(type(h))
----> 4 h1 =flaot(h)
      5 print(h1)
      6 print(type(h1))

NameError: name 'flaot' is not defined
```

In [5]:

```
# integer to float
h=9
print(type(h))
h1 = flaot(h)
print(h1)
print(type(h1))
```

```
<class 'int'>

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-5-b05c3e9f098b> in <module>
      2 h=9
      3 print(type(h))
----> 4 h1 = flaot(h)
      5 print(h1)
      6 print(type(h1))

NameError: name 'flaot' is not defined
```

In [6]:

```
# integer to float
h=9
print(type(h))
h1 = float(h)
print(h1)
print(type(h1))
```

```
<class 'int'>

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-6-b05c3e9f098b> in <module>
      2 h=9
      3 print(type(h))
----> 4 h1 = flaot(h)
      5 print(h1)
      6 print(type(h1))

NameError: name 'flaot' is not defined
```

In [7]:

```
# integer to float
h=9
print(type(h))
h1 = float(h)
print(h1)
print(type(h1))
```

```
<class 'int'>
9.0
<class 'float'>
```

key words python

In [9]:

```
#keywords
import keyword
print(keyword.kwlist)
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break',
'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'fo
r', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'no
t', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

In [12]:

```
# biggest of two numbers
a=299
b=1000
if (a>b):
    print("b is bigger than a")
else:
     print ("a is less than b")
```

```
a is less than b
```

write a program to find biggest of two numbers

write a program to check even or not

write a program to check given age is eligible for vote or not

In [13]:

```
n1=int (input("enter n1 value..."))
n2 =int(input("enter n2 value..."))
if(n1>n2):
    print(n1,"is greater value..")
else :
    print(n2,"is greater value..")
```

```
enter n1 value...100
enter n2 value...150
150 is greater value..
```

In [14]:

```
print("hai,1234")
```

```
hai,1234
```

In [15]:

```
print(12345)
```

```
12345
```

In [16]:

```
print(11+23)
```

```
34
```

In [17]:

```
age=int(input("enter age..."))
if (age>=18):
    print(age,"is eligible value..")
else:
    print(age,"is not eligible value..")
```

```
enter age...17
17 is not eligible value..
```

In [ ]:

```
n=int(input("enter a number"))
# even-divisible by-2
# 2,4,6,8,10,12,14,16
if(n%2==0):
    print("even")
else:
    print("odd")
```

In [1]:

```
# to check the given character is vowels or constant?
#vowels :a,e,i,o
ch=str(input("enter character..."))#ch=i
if(ch== 'a' or ch=='e'or ch=='i'or ch=='0' or ch=='u'):
    print(ch,"it is vowel")
else:
    print(ch,"it is constant")
```

```
enter character...a
a it is vowel
```

In [ ]:

```
n1=int(input("enter n1 value..."))
n2=int(input("enter n2 value..."))
n3=int(input("enter n3 value..."))
if(n1==n2 and n2==n3):
    print("is three ")
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```
#to print the 1 to 10 natural numbers for using for loop
```

In [1]:

```
for i in range(11):
    print(i,end="")
```

```
012345678910
```

In [ ]:

```
to give step value to print the odd numbers from starting number 1 ending number 100
```

to print the odd numbers from 1 to 100 by using for loop

In [2]:

```
for i in range(1,100,2):
    print (i,end=" ")
```

```
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53
55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

to print the 0 to 50 elements

In [3]:

```
for i in range(0,50,3):
    print(i,end=" ")
```

```
0 3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48
```

to print the 1 to n natural numbers in ascending order

In [4]:

```
n=int(input("enter a natural number size"))
for i in range (1,n+1):
    print(i,end=" ")
```

```
enter a natural number size660
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 2
9 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54
55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 8
0 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 1
04 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 1
23 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 1
42 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 1
61 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 1
80 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 1
99 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 2
18 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 2
37 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 2
56 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 2
75 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 2
94 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 3
13 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 3
32 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 3
51 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 3
70 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 3
89 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 4
08 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 4
27 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 4
46 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 4
65 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 4
84 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 5
03 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 5
22 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 5
41 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 5
60 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 5
79 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 5
98 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 6
17 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 6
36 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 6
55 656 657 658 659 660
```

In [5]:

```
n=int(input("enter a natural number size"))
for i in range (1,n+1):
    print(i,end=" ")
```

```
enter a natural number size2000
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 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78
79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102
103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 12
1 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139
140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 15
8 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176
177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 19
5 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213
214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 23
2 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250
251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 26
9 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287
288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 30
6 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324
325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 34
3 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361
```

In [1]:

```python
n=int(input("enter a natural number size"))
for i in range (1,n+1):
    print(i,end=" ")
```

```
enter a natural number size1000
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 2
9 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54
55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 8
0 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 1
04 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 1
23 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 1
42 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 1
61 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 1
80 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 1
99 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 2
18 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 2
37 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 2
56 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 2
75 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 2
94 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 3
13 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 3
32 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 3
51 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 3
70 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 3
89 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 4
08 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 4
27 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 4
46 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 4
65 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 4
84 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 5
03 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 5
22 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 5
41 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 5
60 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 5
79 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 5
98 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 6
17 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 6
36 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 6
55 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 6
74 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 6
93 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 7
12 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 7
31 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 7
50 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 7
69 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 7
88 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 8
07 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 8
26 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 8
45 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 8
64 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 8
83 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 9
02 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 9
21 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 9
40 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 9
59 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 9
78 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 9
97 998 999 1000
```

In [5]:

```
for i in 'apssdc':
    if i=='d':
        break
    else:
        print(i,end=" ")
```

```
a p s s
```

In [6]:

```
for i in '1234567890':
    if i=='5':
        break
    else:
        print(i,end=" ")
```

```
1 2 3 4
```

In [8]:

```
for i in range (1,10):
    if i==5:
        break
    else:
            print(i,end=" ")
```

```
1 2 3 4
```

In [ ]:

```
to print the even numbers in between 1 to 20 using continue keyword
```

In [ ]:

```
for i in range(1,41):
    if (i%)
```

In [9]:

```
# swap between two numbers
a=10
b=5
print("after swapping:a value",a)
print("before swapping:b value,b")
temp =a
#temp =10
a=b
# b=5
b=temp
#b=10
print ("after swapping:a value",a)
print ("after swapping:b value",b)
```

```
after swapping:a value 10
before swapping:b value,b
after swapping:a value 5
after swapping:b value 10
```

In [10]:

```
# how to generate a random number in python
import random
print(random.randint(0,8))
```

```
7
```

In [12]:

```
import string
for letter in string.ascii_lowercase:
    print (letter,end=" ")
for letter in string.ascii_uppercase:
    print(letter,end= " ")
```

```
a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L
M N O P Q R S T U V W X Y Z
```

In [13]:

```
import calendar
year =1999
month=8
print(calendar.month(year,month))
```

```
    August 1999
Mo Tu We Th Fr Sa Su
                   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
```

In [1]:

```
n1=int(input("enter n1 values"))
n2=int(input("enter n2 values"))
def add(a,b):
    c=a+b
    return c
add(n1,n2)
```

```
enter n1 values25
enter n2 values10
```

Out[1]:

```
35
```

In [3]:

```
n1=int(input("enter n1 values"))
n2=int(input("enter n2 values"))
def sub(a,b):
    c=a-b
    print(c)
sub(n1,n2)
```

```
enter n1 values70
enter n2 values21
49
```

In [7]:

```
def adding():
    a=20
    b=30
    sum=a+b
    print("after calling:",sum)
    adding()
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

30-09-2022

In [ ]:

```
> a list is a ordered type of data
> a list as denoted as[]
>
```

In [3]:

```
#accessing a last element in a list
print (li[-1])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-3-d9289670f69a> in <module>
      1 #accessing a last element in a list
----> 2 print (li[-1])

NameError: name 'li' is not defined
```

In [ ]:

```
# accessing the item in a list or not
if
```

In [ ]:

```
# example for the list
li=
```

In [ ]:

```
> a list is a collection of characters variables,and number variables and boolen values dat
>a list
```

In [4]:

```
# example for the list
li=["apple","banana","orange" ,"graphs","milk"]
li
```

Out[4]:

```
['apple', 'banana', 'orange', 'graphs', 'milk']
```

In [5]:

```
# type of the list
print(type(li))
```

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

In [6]:

```
# length if the list
print (len(li))
```

```
5
```

In [7]:

```
# length if the list
print(type(li))
```

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

In [8]:

```
# assesing first element in a list
print(li[0])
```

```
apple
```

In [9]:

```
# assesing first element in a list
print(li[-1])
```

```
milk
```

In [ ]:

```
# accessing the item in a list or not
```

In [10]:

```
li.insert(1,"nani")
li
```

Out[10]:

```
['apple', 'nani', 'banana', 'orange', 'graphs', 'milk']
```

In [11]:

```
li1=["suresh","venu","balram"]
li1
```

Out[11]:

```
['suresh', 'venu', 'balram']
```

In [13]:

```
li[2:5]
```

Out[13]:

```
['banana', 'orange', 'graphs']
```

In [14]:

```
li[2:]
```

Out[14]:

```
['banana', 'orange', 'graphs', 'milk']
```

In [15]:

```
li1=["sbi","cbi"]
li+li1
```

Out[15]:

```
['apple', 'nani', 'banana', 'orange', 'graphs', 'milk', 'sbi', 'cbi']
```

In [16]:

```
li1
```

Out[16]:

```
['sbi', 'cbi']
```

In [17]:

```
li.sort()
li
```

Out[17]:

```
['apple', 'banana', 'graphs', 'milk', 'nani', 'orange']
```

In [19]:

```
del li[1]
li
```

Out[19]:

```
['apple', 'milk', 'nani', 'orange']
```

In [20]:

```
# create tuple
t1=(10,20,30)
t1
print(type(t1))
```

```
<class 'tuple'>
```

In [21]:

```
# single value tuple
t2=(10)
print (type(t2))
t3=(20,)
print(type(t3))
```

```
<class 'int'>
<class 'tuple'>
```

In [22]:

```
t3
```

Out[22]:

```
(20,)
```

In [23]:

```
t2
```

Out[23]:

```
10
```

In [24]:

```
t2=(10,20,10,20,20,30,20,20,30,10)
# to count the number of occurences
t2.count(30)
```

Out[24]:

```
2
```

In [25]:

```
t2=(10,20,10,20,20,30,20,20,30,10)
# to count the number of occurences
t2.count(20)
```

Out[25]:

```
5
```

In [26]:

```
#index
t2.index(20)
```

Out[26]:

```
1
```

In [27]:

```
t2.index(30)
```

Out[27]:

```
5
```

In [28]:

```
t2.index(10)
```

Out[28]:

```
0
```

In [31]:

```
tuple1= ("abc",34,"true",40,"male")
print(tuple1)
```

```
('abc', 34, 'true', 40, 'male')
```

In [1]:

```
# to create a dictionaries with values
d1={'a':10,'b':34,'c':45}
print(d1)
print(type(d1))
```

```
{'a': 10, 'b': 34, 'c': 45}
<class 'dict'>
```

In [2]:

```
# to create a dictionaries with different data types..
d2={'a':100,'name':'RAVINDRANADH.GADDE','branch':'cse','b':45.8}
print(d2)
```

```
{'a': 100, 'name': 'RAVINDRANADH.GADDE', 'branch': 'cse', 'b': 45.8}
```

In [ ]:

```
# update the dictionary values
```

In [3]:

```
print(dir(dict))
```

```
['__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__do
c__', '__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 [5]:

```
# pop item
print(d2)
print(d2.clear())
```

```
{'a': 100, 'name': 'RAVINDRANADH.GADDE', 'branch': 'cse', 'b': 45.8}
None
```

In [7]:

```
# pop item
print(d2)
print(d2.popitem())
```

```
{}

---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-7-b6aa6d3e0831> in <module>
      1 # pop item
      2 print(d2)
----> 3 print(d2.popitem())

KeyError: 'popitem(): dictionary is empty'
```

In [ ]:

```
# git is a local system
# git hub is a remote system
```