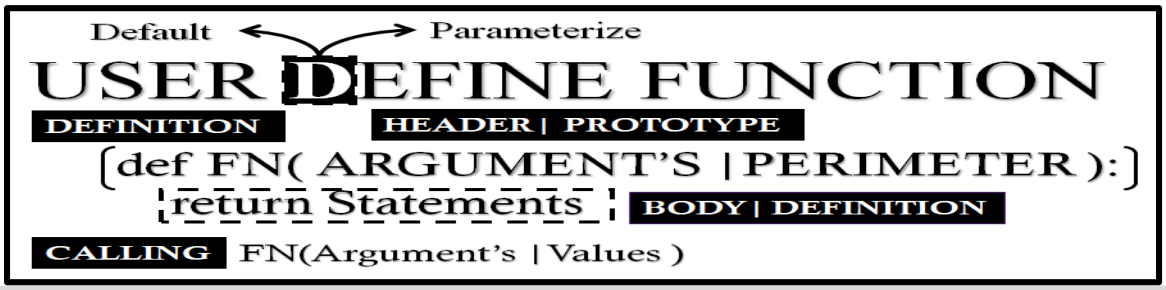
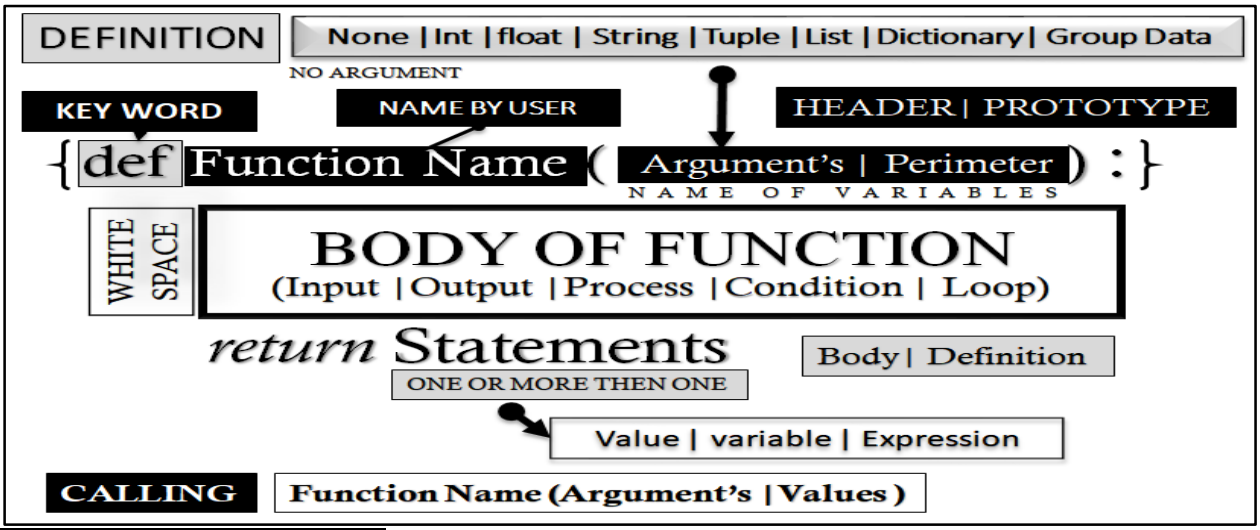
2 Python Function User Define Methods



Function Methods

- *Python function/methods*
- *Defining a python function*
- *Function with arguments*
- *Python function with the argument and return value*
- *Python function with default argument*
- *Python function with variable-length arguments*
- *Keyword variable length of arguments*
- *Argument pass by reference or value*
- *Call by reference*
- *Call by reference*
- *Local variable*
- *Global variable*
- *Global reassigning in local*
- *Local variable accessing from outside the function*
- *Global variable*
- ***Programs***
- **New Assignments Functions using Python**



RULES FOR FUNCTION

1. PER DEFINE FUNCTIONS LIKE: **len() | sum() |max() | min() | str()…**
2. CREATE FUNCTION NAME LIKE A VARIABLE OR IDENTIFIER : **FUN() | SUM() |CAL()**
3. BODY OF A FUNCTION CALL ACCORDING TO FUNCTION CALL: **loop | if else |input |output**
4. IN CALLING NEED TO PASS VALUE OR IDENTIFIER: **FUN(3,4) |FUN(A,B)**
5. FUNCTION CAN RETURN ONE OR MORE THEN ONE VALUE AT A TIME, MULTIPLE VALUES ARE CONVERTED AS TUPLE AUTOMATICALLY
6. FUNCTION NAME SHOULD BE SAME IN DEFINITION OR IN CALLING

DEFAULT & PARAMETERIZE

DEFAULT — NO ARGUMENTS

```
def Function Name ():
    BODY OF FUNCTION
    return Statements
```

PARAMETERIZE — WITH ARGUMENTS

```
def Function Name ( Argument's ):
    BODY OF FUNCTION
    return Statements
```

	DEFINITION	CALLING	OUTPUT
1.	def Display( ): print("Default")	Display( )	Default
2.	def Show( ): return " Hi "	print(Show( ))	Hi
3.	def MSG( ): print(" Hi! Ravi") print(" How are") print(" You now.")	MSG()	Hi! Ravi How are You now.
4.	def Menu( ): print("1. ADD") print("2. MUL") print("3. AVG")	Menu()	1. ADD 2. MUL 3. AVG
5.	def OPERATIONS( ): print("Create") print("DELETE") print("EDIT") print("MODIFY")	OPERATIONS()	Create DELETE EDIT MODIFY

	DEFINITION	CALLING	OUTPUT
1.	def Sum(*a* , *b*): print((a + b))	Sum(*3,3*)	6
2.	def Mul(*n* , *m*): return n * m	Mul(*3,3*)	9
3.	def Sum(*a* , *b*): print(a**b)	Sum(*3,3*)	27
4.	def Sum(*x* , *y*): return x+2, y+3	print(Sum(*3,3*))	(5,6)
5.	def Sum(a , b): return a + b	print(Sum(3,7))	10
6.	def Sum(a , b): return a+2, b+3	print(Sum(3,7))	(5,10)
7.	def Sum(a , b): return a + b	print(Sum('A','B'))	'AB'

A FUNCTION IS A MECHANISM THAT GENERATE RESULT |OUTPUT ACCORDING TO ITS BODY

FUNCTION	OPERATION
1. FUN(a)	= a**a
2. FUN(X)	= X+2
3. FUN(X,Y)	= X*2+Y*3
4. FUN(X,Y)	= X*Y
5. FUN(X,Y)	= X**Y
6. FUN(X,Y)	= X+X*Y
7. FUN(X,Y)	= X^Y
8. FUN(X,Y)	= (X+Y)/2
9. FUN(X,Y)	= $(X+Y)^2$
10. FUN(X,Y)	= X^2+Y^2
11. FUN(X,Y)	= X^2+Y^2+3
12. FUN(X,Y,N)	= X^N+Y^N
13. FUN(X,Y,N)	= $(X+Y)^N$
14. FUN(X,Y,Z)	= (X+Y)/Z

Things Working Like Function

#1
```
# DEFINITION
def FUN(a):
    print(a**a)
```

#2
```
def FUN(X):
    print(X+2)
X=int(input("Enter X:"))
FUN(X)
```

#3
```
def FUN(X,Y):
    print(X*2+Y*3)
```

#5
```
def FUN(X,Y):
    print(X**Y)
```

#4
```
def FUN(X,Y):
    print(X*Y)
```

 # CALLING

CASE 1	CASE 2	CASE 3
FUN(5)	a=5 FUN(a)	a=int(input("Enter a:")) FUN(a)

I
```
X=int(input("Enter X:"))
Y=int(input("Enter Y:"))
FUN(X,Y)
```

x**y=pow(x,y)

#6
```
def FUN(X,Y):
    print(X+X*Y)
```

#7
```
def FUN(X,Y):
    print(X**Y)
```

#8
```
def FUN(X,Y):
    print((X+Y)/2)
```

#9
```
def FUN(X,Y):
    print((X+Y)**2)
```

#10
```
def FUN(X,Y):
    print((X**2+Y**2))
```

#11
```
def FUN(X,Y):
    print((X*X+Y*Y+3))
```

#print((X*X+Y*Y))

#12
```
N=int(input("Enter N:"))
def FUN(X,Y,N):
    print(X**N+Y**N)
```

#13
```
N=int(input("Enter N:"))
def FUN(X,Y,N):
    print((X+Y)*N)
```

#14
```
Z=int(input("Enter Z:"))
def FUN(X,Y,Z):
    print((X+Y)/N)
```

Convert Normal Code Into Function

Direct

1. `print("Hi! Python")`

Hi! Python

Direct

2. `a , b=3,5`
   `print((a + b))`

8

Direct

3. `MSG= "Hi! Python"`
   `print(MSG)`

Hi! Python

1. DEFAULT

```
def Display():
    print("Hi! Python")
Display()
```
CALLING

2.

```
def Sum( a , b ):
    print((a + b))
a , b=3,5
Sum(a , b)
```
CALLING

3.

```
def Display( Str ):
    print(Str)
MSG= "Hi! Python"
Display(MSG)
```
CALLING

PARAMETERIZE

FUNCTION

Things Working Like Function

4. Direct

```
a=int(input("Enter a: ")
b=int(input("Enter b: ")
print((a + b))
```

4.

```
def Sum( a , b ):
    return a + b
a=int(input("Enter a: ")
b=int(input("Enter b: ")
print(Sum(a , b))
```
CALLING

Enter a: 2
Enter b: 4

6

DIFFERENT KIND OF FUNCTION

TOP

Bottom

1 Without Return

```
2 def Display():
3     print("Hi! Python")
1 Display()
```
OUTPUT Hi! Python

2 Without Return

```
3 def Sum(a , b):
4     print((a + b))
1 a , b=3,5
2 Sum(a , b)
```
OUTPUT 8

3 Without Return — Hi! Python

```
3 def Display( Str ):
4     print(Str*2)
1 MSG= "Hi! Python"
2 Display(MSG)
```
OUTPUT Hi! Python Hi! Python

4 With Return

```
3     def Sum(a , b):
4         return a + b
1     a , b=3,5
2, 5  print(Sum(a , b))
```
(2 → 5)

OUTPUT 8

5 With Return

```
4     def Sum(a , b):
5         return a + b
1     a=int(input("Enter a: ")
2     b=int(input("Enter b: ")
3, 6  print(Sum(a , b))
```
(3 → 6)

OUTPUT
Enter a: 12
Enter b: 23
SUM is 25

LAB WORK 1

1

```
4   def Sum(a , b):
5       return a + b
Start 1 a =int(input("Enter a: "))
    2   b =int(input("Enter b: "))
3,6     print("R:",Sum(a , b))
```
(4 → 9)

3 → 6

OUTPUT
Enter a: 4
Enter b: 9
R: 13

2 WRONG PROGRAM

```
Sum(a , b)      #Error
def Sum(a , b):
    print(a + b)
a =int(input("Enter a: ")
b =int(input("Enter b: ")
Sum(a , b)
```
CALLING

CALLING BEFORE DEFINITION

3 RIGHT

```
6   def Sum(a , b):
7       print("R:",a + b)
Start 1 Sum(4 , 7)
    2   Sum(12, 8)
    3   a =int(input("Enter a: "))
    4   b =int(input("Enter b: "))
    5   Sum(a , b)
```

OUTPUT
R: 11
R: 20
Enter a: 3
Enter b: 9
R: 12

4 MISSED STATEMENT

```
    def Sum(a , b):
2       print("A :" , a+b)
3       print("B :" , a*b)
4       return a**b
5   X   print("C", a % b)
Start 1, 6 R=Sum(8 , 2)
7   print("Result : " , R)
```
(8 → 2)

Missed

OUTPUT
A : 10
B : 16
Result : 64

LAB WORK 2

SINGLE VALUE RETURN

```
4   def Sum(a , b):
5       return a + b
1   a=int(input("Enter a : "))
2   b=int(input("Enter b : "))
3,6 Result= Sum(a , b)
7   print("R : ",Result)
```

Value 1

OUTPUT

```
Enter a: 12
Enter b: 9
R :  21
```

MULTI VALUE RETURN AS TUPLE

```
4   def Sum(a , b):
5       return (a + 2), (b+3)
1   a=int(input("Enter a: "))
2   b=int(input("Enter b: "))
3,6 Result= Sum(a , b)
7   print("R : ", Result)
```

Value 1, Value 2

OUTPUT

```
Enter a : 11
Enter b : 12
R : (13, 15)
```

```
7   3,6 R1,R2= Sum(a , b)
    7   print(R1," , ",R2)
    8   print("R1 : ",R1)
    9   print("R2 : ",R2)
```

OUTPUT

```
Enter a: 11
Enter b: 14
13 , 17
R1 :  13
R2 :  17
```

```
8   3,6 Result= Sum( a , b)
    7   R1, R2 = Result
    8   print(R1)
    9   print(R2)
```

OUTPUT

```
Enter a: 11
Enter b: 15
R1 :  13
R2 :  18
```

FIND OUTPUT SET 1

PROBLEM 1

```
2   def SHOW1( a , b , c ):
3   I   return (a+b)**c, (a+b)*c
4   def SHOW2( a , b , n ):
5   II  return a**n +b*n, a*b**n
1,4 print(SHOW1( 3 , 2 , 2 ))
3,6 print(SHOW2( 3 , 2 , 2 ))
```

OUTPUT

I (25, 10)
II (13, 12)

DISPLAY CODE 1

1. print(SHOW1(3,4,2))
2. print(SHOW1(2,2,3))
3. print(SHOW1(2,2,4))
4. print(SHOW1(3,3,3))
5. print(SHOW1(3,5,2))
6. print(SHOW1(4,4,2))
7. print(SHOW1(2,1,3))
8. print(SHOW1(2,3,2))

DISPLAY CODE 2

1. print(SHOW2(3,4,2))
2. print(SHOW2(2,2,3))
3. print(SHOW2(2,2,4))
4. print(SHOW2(3,3,3))
5. print(SHOW2(3,5,2))
6. print(SHOW2(4,4,2))
7. print(SHOW2(2,1,3))
8. print(SHOW2(2,3,2))

OUTPUT 1

1. (49, 14)
2. (64, 12)
3. (256, 16)
4. (216, 18)
5. (64, 16)
6. (64, 16)
7. (27, 9)
8. (25, 10)

OUTPUT 2

1. (17, 48)
2. (14, 16)
3. (24, 32)
4. (36, 81)
5. (19, 75)
6. (24, 64)
7. (11, 2)
8. (10, 18)

PROBLEM 1

```
2   def FUN1( a , b , c ):
3   I   return (a+3)**c, (b+5)*c
4   def FUN2( a , b , n ):
5   II  return a*n +b**n, a**b*n
1,4 print(FUN1( 3 , 2 , 2 ))
3,6 print(FUN2( 3 , 2 , 2 ))
```

OUTPUT

I (36, 14)
II (10, 18)

DISPLAY CODE 1

1. print(FUN1(3,4,2))
2. print(FUN1(2,2,3))
3. print(FUN1(2,2,4))
4. print(FUN1(3,3,3))
5. print(FUN1(3,5,2))
6. print(FUN1(4,4,2))
7. print(FUN1(2,1,3))
8. print(FUN1(2,3,2))

DISPLAY CODE 2

1. print(FUN2(3,4,2))
2. print(FUN2(2,2,3))
3. print(FUN2(2,2,4))
4. print(FUN2(3,3,3))
5. print(FUN2(3,5,2))
6. print(FUN2(4,4,2))
7. print(FUN2(2,1,3))
8. print(FUN2(2,3,2))

OUTPUT 1

1. (36, 18)
2. (125, 21)
3. (625, 28)
4. (216, 24)
5. (36, 20)
6. (49, 18)
7. (125, 18)
8. (25, 16)

OUTPUT 2

1. (22, 162)
2. (14, 12)
3. (24, 16)
4. (36, 81)
5. (31, 486)
6. (24, 512)
7. (7, 6)
8. (13, 16

Problem 1

```
4 8
def FUN( a , b ):
    a+=2 → 6
    b*=5 → 40
    print(a, ":",b)
a, b=4 , 8
print(a, ":",b)
4 8
FUN( a , b )
print(a, ":",b)
```

II 6 : 40
I 4 : 8
III 4 : 8

OUTPUT

I 4 : 8
II 6 : 40
III 4 : 8

Problem 2

```
def FUN( a , b ):
    a+=2
    b*=5
    print(a, ":",b)
a, b=4 , 8
print(a, ":",b)
FUN( a , a )
print(a, ":",b)
```

Problem 3

```
def FUN( a , b ):
    a+=2
    b*=5
    print(a, ":",b)
a, b=4 , 8
print(a, ":",b)
FUN( b , b )
print(a, ":",b)
```

Problem 4

```
def FUN( a , b ):
    a+=2
    b*=5
    print(a, ":",b)
a, b=4 , 8
print(a, ":",b)
FUN( b , a )
print(a, ":",b)
```

OUTPUT

Problem 1:	Problem 2:	Problem 3:
4 : 8	4 : 8	4 : 8
6 : 20	10 : 40	10 : 20
4 : 8	4 : 8	4 : 8

FUNCTION WITH DEFAULT VALUE

CODE 1

```
2,6,10  def DEFAULT( a=6 ):
3,7,11      a+=3
4,8,12      print(a)
1           DEFAULT(3)
5           DEFAULT(13)
9           DEFAULT()
```

{ 6 }
{ 16 }
{ 9 }

CODE 2

```
2,6,10  def DEFAULT(a , b=6 ):
3,7,11      a+=3; b+=a
4,8,12      print(a,',',b)
1           DEFAULT(4,3)
5           DEFAULT(5)
9           DEFAULT(7,8)
```

{ 7 , 10 }
{ 8 , 14 }
{ 10 , 18 }

CODE 3

```
2,6,10  def DEFAULT( a=5 , b=6 ):
3,7,11      a+=3;b+=a
4,8,12      print(a,',',b)
1           DEFAULT(4,3)
5           DEFAULT(5)
9           DEFAULT()
```

{ 7 , 10 }
{ 8 , 14 }
{ 8 , 14 }

Can't Use	Can Use		Calling
X 1. def sum();	1. def sum():	✓	1) sum()
X 2. def sum(a=4 ,b=6 ,c):	2. def sum(a , b , c=5):	✓	2) sum(4,6) | sum(7,5,8)
X 3. def sum(a , b=6 ,c):	3. def sum(a , b=6 ,c=5):	✓	3) sum(4) | sum(7,5,8) | sum(7,5)
X 4. def sum(a=4 ,b ,c):	4. def sum(a=4 ,b=5 ,c=3):	✓	4) sum(7,5,8) | sum(7,5) | sum(7) |sum()

PROBLEM 1

```
3 2
2   def SUM( a , b=7 , c=2 ):
3   I   print (a*b, " : ",b*c, " : ",b**c)
4   II  return (a+b)*c
1,5     print(SUM( 3 , 2 ))
```

OUTPUT

Line I 6 : 4 : 4
Line II 10

CODE

1. print(SUM(2,3))
2. print(SUM(4,2,3))
3. print(SUM(2,2,2))
4. print(SUM(2,2))
5. print(SUM(2))
6. print(SUM(2,4))
7. print(SUM(3))
8. print(SUM(2,7))
9. print(SUM(1,2,3))
10. print(SUM(2,2,3))
11. print(SUM(5))
12. print(SUM(3,4))
13. print(SUM(1,3))

OUTPUT ?

1. 6 : 6 : 9 ,10
2. 8 : 6 : 8 ,18
3. 4 : 4 : 4 ,8
4. 4 : 4 : 4 ,8
5. 14 : 14 : 49 ,18
6. 8 : 8 : 16 ,12
7. 21 : 14 : 49 ,20
8. 14 : 14 : 49 ,18
9. 2 : 6 : 8 ,9
10. ?
11. ?
12. ?
13. ?

PROBLEM 1

```
                    (3)   (2)
2  def FUN( a=2 , b=7 ):
3        a+=2
4        b*=5
5      print (a, " @ ", b)
1  FUN( 3 , 2 )
```

OUTPUT I [5 @ 10]

CODE	OUTPUT ?
1. FUN(3 , 4)	1. 5 @ 20
2. FUN(7)	2. 9 @ 35
3. FUN(16,10)	3. 18 @ 50
4. FUN(8)	4. 10 @ 35
5. FUN( )	5. 4 @ 35
6. FUN(3,3)	6. 5 @ 15
7. FUN(6,6)	7. 8 @ 30
8. FUN(0,1)	8. 2 @ 5
9. FUN(16)	9. 18 @ 35
10. FUN(1,2)	10. 3 @ 10
11. FUN(4,"CD")	11. 6 @ CDCDCDCDCD
12. FUN(3, "A")	12. 5 @ AAAAA
13. FUN(2, "Ravi")	13. 4 @ RaviRaviRaviRaviRavi
14. FUN(3,1)	14. ?
15. FUN(1,1)	15. ?
16. FUN(0, len("SUMAN"))	16. ?
17. FUN(-10,-11)	17. ?
18. FUN(len("LAN", len("MAN")	18. ?
19. FUN(13,-3)	19. ?
20. FUN(len("XYZ"), "XYZ")	20. ?

PROBLEM 2

```
                    (3)   (2)    (4)
2     def FUN( X , Y=7 , Z=2):
3           X+=Y
4           Y*=Z
5           Z=X+Y
6         print (a, " # ", b, " @ ", Z )
1     FUN( 3 , 2, 4 )
```

OUTPUT I [5 # 8 @ 13]

CODE	OUTPUT ?
1. FUN(7 , 8, 4)	1. 15 # 32 @ 47
2. FUN(2 , 3, 4)	2. 5 # 12 @ 17
3. FUN(0 , 1, 4)	3. 1 # 4 @ 5
4. FUN(2 , 3, 4)	4. 5 # 12 @ 17
5. FUN(1 , 2, 4)	5. 3 # 8 @ 11
6. FUN(2)	6. 9 # 14 @ 23
7. FUN( )	7. #ERROE
8. FUN(2 , 2, 2)	8. 4 # 4 @ 8
9. FUN(2 , 2)	9. 4 # 4 @ 8
10. FUN(7 , 2, 1)	10. 9 # 2 @ 11
11. FUN(1 , 10, 2)	11. 11 # 20 @ 31
12. FUN(1 , 1, 1)	12. 2 # 1 @ 3
13. FUN(1 , 0, 0)	13. 1 # 0 @ 1
14. FUN(0 , 11, 2)	14. 11 # 22 @ 33
15. FUN(7 , 1)	15. 8 # 2 @ 10
16. FUN(11 , 2, 8)	16. ?
17. FUN(4 , 3, 7)	17. ?
18. FUN(2 ,4, 6)	18. ?
19. FUN(5, 5, 5)	19. ?
20. FUN(10, 6, 1)	20. ?

Find Output Set 3 (Function Arguments as default value) *Arguments as default*

Example *Arguments as default*

```
4,10 def FUN ( a , b=7 ):
5,11     a+=2
6,12     b*=5
7,13     print(a, ":", b)
1    a, b= 4, 8
2    print(a, ":",b)
3    FUN(a, b)
8    print(a, ":",b)
9    FUN(a)
14   print(a, ":",b)
```

II IV I III V

OUTPUT

I	4 : 8
II	6 : 40
III	4 : 8
IV	6 : 35
V	4 : 8

Problem 1

```
def FUN(a=2, b=7 ):
    a+=2
    b*=5
    print(a, ":", b)
a, b = 4 , 8
print(a, ":", b)
FUN(a, b)
print(a , ":", b)
FUN(a)
FUN(b)
FUN()
print(a , ":", b)
```

Problem 2

```
def FUN ( a=5 , b=7 ):
    a*=b
    b=a + b
    print(a, ":", b)
a, b=4,8
print(a, ":",b)
FUN(a, b)
FUN(a)
FUN(b)
FUN()
```

Problem 3

```
def s( a=6 , b=7 ):
    a+=2
    b*=5
    print(a, ":", b)
a, b=4,8
print(a, ":",b)
s(a, b)
print(a, ":",b)
s(a)
print(a, ":",b)
s(b)
print(a, ":",b)
s()
print(a , ":", b)
```

Problem 4

```
def func ( x, y=10 ):
    if x%y==0 :
        return x+2
    else:
        return y-1
p , q=20,23;
print(p ,",", q)
q= func (p , q)
print(p ,",", q)
```

Problem 5

```
def FUN(a, b=10 ):
    print(a , ",", b)
a, b=4,13
print(a, " " ,b)
FUN(a , b)
FUN(a)
FUN(b)
FUN(b , a)
```

OUTPUT

Problem 1:	4:35	35 : 42	4 : 8	20 , 23	13 , 10
4: 8	Problem 2:	Problem 3:	10 : 35	20 , 22	13 , 4
6:40	4 : 8	4 : 8	4 : 8	Problem 5:	
4: 8	32 : 40	6 : 40	8 : 35	4 , 13	
6:35	28 : 35	4 : 8	4 : 8	4 , 13	
10:35	56 : 63	6 : 35	Problem 4:	4 , 10	

Find Output Unsolved 1

Q1.

```
def func(x, y=10 ):
    if x%y==0 :
        return x+2
    else:
        return y-1
p , q=20,23;
print(p ,",",q)
q=func(p , q)
print(p ,",",q)
p=func(q)
print(p ,",", q)
```

Q2.

```
def func(x, y=10 ):
    if x%y==0:
        return x+2
    else:
        return y-1
p , q=20,23;
print(p ,",",q)
q=func(p , q)
print(p ,",",q)
p=func(q)
print(p ,",",q)
q=func(p)
print(p , ",", q)
```

Q3.

```
def execute(x , y=200 ):
    temp =x + y
    x+=temp
    if(y!=200):
        print(temp ,",", x , ",", y)
a , b =50,20
execute(b)
print(a ,",", b)
execute(a , b)
print(a ,",", b)
```

Q4.

```
def execute(x , y=15 ):
    temp=x + y
    x+=temp
    if(y<=20):
        print(temp ,",", x , ",", y)
a , b =50,20
execute(b)
print(a ,",", b)
execute(a , b)
print(a ,",", b)
```

Q5.

```
def execute(x , y=20  ):
    temp=x*y
    x+=temp
    if(y>=20):
    print(temp ,",", x , ",", y)
a , b =50,20
execute(b)
print(a ,",", b)
execute(a , b)
print(a ,",", b)
```

Q6.

```
def DEFAULT( a=6  ):
    a+=3
    print(a)
DEFAULT(3) ;DEFAULT(13)
DEFAULT()
```

Q7.

```
def DEFAULT(a , b=6 ):
    a+=3; b+=a
    print(a,', ',b)
DEFAULT(4,3)
DEFAULT(5);DEFAULT(7,8)
```

Q8.

```
def DEFAULT( a=5 , b=6 ):
    a+=3;b+=a
    print(a,', ',b)
DEFAULT(4,3);DEFAULT(5)
DEFAULT()
```

======OUTPUT=======

Ans. 1:
20 , 23
20 , 22
9 , 22

Ans. 2:
20 , 23
20 , 22
9 , 22
9 , 9

Ans. 3:
50 , 20
70 , 120 , 20
50 , 20

Ans. 4:
35 , 55 , 15
50 , 20
70 , 120 , 20
50 , 20

Ans. 5:
400 , 420 , 20
50 , 20
1000 , 1050 , 20
50 , 20

Ans. 6:
6
16
9

Ans. 7:
7 , 10
8 , 14
10 , 18

Ans. 8:
7 , 10
8 , 14
8 , 14

Find Output Unsolved 2

```
1. def Fun1():
       print('Python, let's fun with functions')
   Fun1()
2. def add(i):
       if(i*3%2==0):
           i*=i
       else:
           i*=4
       return i
   a=add(10)
   print(a)
3. import math
   def area(r):
       return math.pi*r*r
   a=int(area(10))
   print(a)
4.def fun1(x, y):
       x = x + y
       y = x – y
       x = x – y
       print('a=',x)
       print('b=',y)
   a = 5
   b = 3
   fun1(a,b)
5. def div5(n):
       if n%5==0:
           return n*5
       else:
           return n+5
   def output(m=5):
       for i in range(0,m):
           print(div5(i),'@',end=" ")
       print('n')
   output(7)
   output()
   output(3)
6.def sum(*n):
       total=0
       for i in n:
           total+=i
           print('Sum=', total)
   sum()
   sum(5)
   sum(10,20,30)
7.def func(b):
       global x
       print('Global x=', x)
       y = x + b
       x = 7
       z = x – b
       print('Local x = ',x)
       print('y = ',y)
       print('z = ',z)
   x=5
   func(10)
8. def func(x,y=100):
       temp = x + y
       x += temp
       if(y!=200):
           print(temp,x,x)
   a=20
   b=10
   func(b)
   print(a,b)
   func(a,b)
   print(a,b)
9. def get(x,y,z):
       x+=y
       y-=1
       z*=(x-y)
       print(x,'#',y,'#',z)
   def put(z,y,x):
       x*=y
       y+=1
       z*=(x+y)
       print(x,'$',y,'$',z)
   a=10
   b=20
   c=5
   put(a,c,b)
   get(b,c,a)
   put(a,b,c)
   get(a,c,b)
```

======OUTPUT=======

Ans. #1.
Python, lets fun with functions
Ans. # 2.
100
Ans. #3.
314
Ans. #4.
a= 3
b= 5
Ans. #5.
0 @ 6 @ 7 @ 8 @ 9 @
25 @ 11 @ n
0 @ 6 @ 7 @ 8 @ 9 @ n
0 @ 6 @ 7 @ n
Ans. #6.
Sum= 5
Sum= 10
Sum= 30
Sum= 60
Ans. #7.
Global x= 5
Local x = 7
y = 15
z = -3
Ans. #8.
110 120 120
20 10
30 50 50
20 10
Ans. #9.
100 $ 6 $ 1060
25 # 4 # 210
100 $ 21 $ 1210
15 # 4 # 220

DIFFERENT TYPE OF ARGUMENTS IN FUNCTION

LIST

```
def Sum(DL):
    return DL[0] + DL[1]
DL =[3,6]
print(Sum(DL))
```

9

LIST,TUPLE

```
def Join (DL,TL):
    DL+=TL
    print(DL)
DL =[3,6]
TL=(2,8)
Join (DL,TL)
```

[3, 6, 2, 8]

LIST MUTABLE

```
def Sum(DL):
    DL[0]+=2
    DL[1]+= DL[0]
    print(DL)
DL =[3,6]
print(DL)
Sum(DL)
print(DL)
```

[5, 11]
[3, 6]
[5, 11]

TUPLE IMMUTABLE

```
def CHA(TL):
    TL=list(TL)
    TL[0]=TL[0]+2
    TL[1]=TL[1]+3
    print(TL)
TL =(3,6)
print(TL)
CHA(TL)
print(TL)
```

[5, 9]
(3, 6)
(3, 6)

Crate Tuple By Default

```
def Group(*T):
    print(T)
a , b , c=12,23,21
print(a, ",", b,"," , c)
Group(a , b , c)
```

(12 , 23 , 21)
(12, 23, 21)

```
def Group(*T):
    print(T)
a , b , c=12,23,21
Group(a , b, c)
Group(12,"Pawan","10 A",17)
```

(12, 23, 21)
(12, ' Pawan ' ,'10 A', 17)

Problem 1

Single Arguments as LIST

```
def FUN (a, b):
    a+=2
    b[0]=b[0]+5
    print(a, ":",b)
a, b=4,[8]
print(a, ":",b)
FUN(a , b)
print(a, ":",b)
```

II
I
III

OUTPUT

I 4 : [8]
II 6 : [13]
III 4 : [13]

LIST *a= [4] b= [8]*

Problem 2

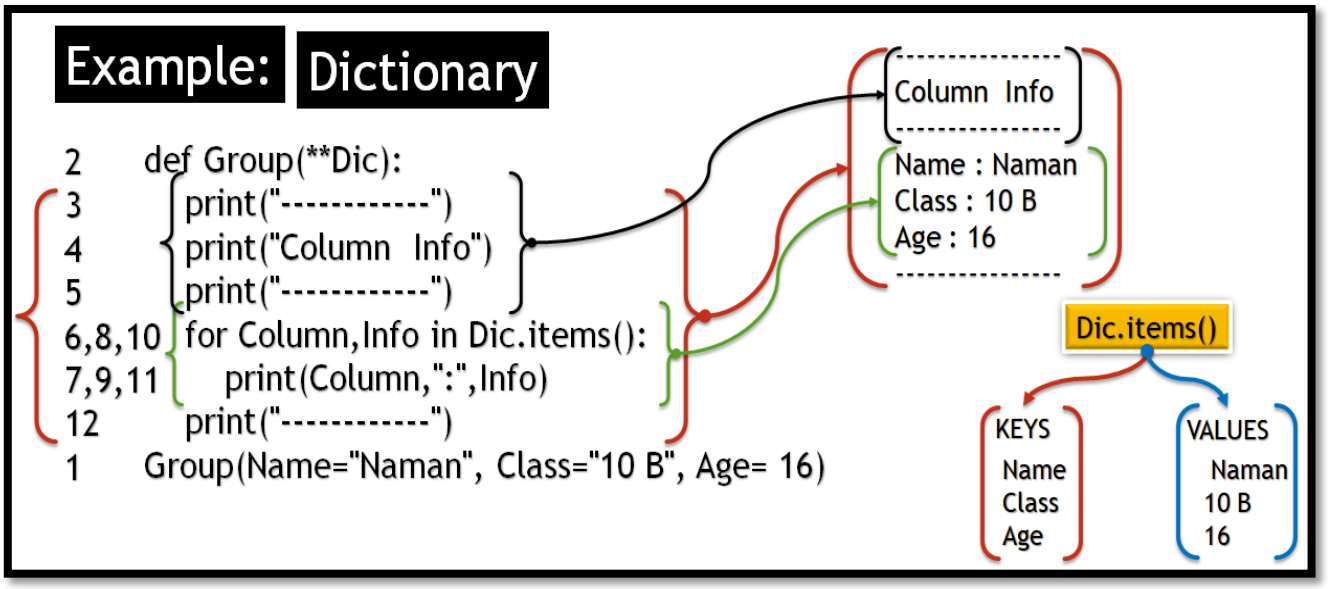
Both Arguments as LIST

```
def FUN (a, b):
    a[0]+=2
    b[0]+=5
    print(a , ":",b)
a , b= [4] , [8]
print(a, ":",b)
FUN(a , b)
print(a , ":",b)
```

II
I
III

OUTPUT

I [4] : [8]
II [6] : [13]
III [6] : [13]



You have seen several built-in function in python like len(), range() and cmp(). if you want to create own function, Python lets you to create own functions. So what is the advantage of creating own function ? well Function allows you to break up your long program into magangeable blocks, Easy to maintain and reuse. It eliminate redundancy. As we go further you will get to know the advantages of function.

Defining a Python function: In python , very simple rule to define a function.

- Use def keyword followed by functiona name with parentheses () .
- Any arguments to function must be placed within these parentheses.
- The code block must be start with : .
- The code with the function must be indented.

Syntax

```
def function_name(arguments):
code
    return value
```

Calling a Python function:

function_name():

Calling a Python function like same calling the built-in function. Use the name of the function followed by set of parentheses. So i can call this function many time with same or different arguments. You can execute it by calling it from another Python function. So in this way i can say function reduce the size of code as well as reduce redundancy.

let's learn from example.

```
def hello():
  """ This is Hello program"""
  print "You are in Hello"
hello()
```

hello function: Above program is very simple just define **hello()** function. The line **def hello():** and its block are a Python function definition. They define what the function does, but don't run. It will not run the function untill it sees a function call for it. Consider you want to document the function. Functions have special mechanism that allow you to document them. That is called docstring.

```
def fun(): #Method Declaration
   print("hi i am a default method...")#Method Body
fun() #Method Calling
def fun1(a):
   print("a : ",(a))
fun1(3)def fun2(a,b):
   print("Addition is : ",(a+b))
fun2(3,6)
def fun2(a,b):
   return (a+b) #method with return type
print("the result is ",fun2(3,6))
```

Function with arguments: As you've seen with built-in function, like len() function, you pass the sequence and the function return its length. Your own Python function also receive and return arguments. . When you use keyword arguments in a Python function call, the caller identifies the arguments by the parameter name. This allows you to place them out of order because the Python interpreter is able to use the keywords provided to match the values with parameters.

Let us discuss our example

```
def fun1(str1):
  print str1
```

```
str1 = "hello all"
fun1(str1)
```

hello function with argument : In above program argument "str" has been passed by calling Python function. The string **str** is printed with in the function. Consider a case when argument is not passed what happen ? Next example will clear the picture.

Showing error in function

Python Function with the argument and return value: Consider a Python function with return value is small department where some material goes and new processed material comes out.

Let us discuss next example

```
def sum(a, b):
  c = a+b
  return c
x = 10
y = 50
print "Send x and y to sum department"
print "value of addition ", sum(x,y)
```

In above example print statement is outside from the function. Print statement print the function**sum(x,y)** , so what is the value of **sum(x,y)** ? As function return **c** , that is the value of function.

Python function with default Argument: If you don't provide value to argument then the function take its default value defined in the function. Our next example will clear all the doubt.

```
def info(name, age=50):
  print "Name", name
  print "age", age
info("mohit", age=28)
info("Ravi")
```

Python function with Variable-length arguments: Sometime you need to pass more arguments than you specified while defining the Python function. These type of arguments are called variable-length arguments.

Syntax: def function_name(arg, *vartuple):
code block
return

Let's discuss our next example:

```
def variable( agr1, *vari):
  print "Out-put is",agr1
  for var in vari:
    print var
variable(6)
variable(90,9,4,5,6)
```

From output we deduce that first argument is handled by argument **arg1** rest arguments are handled by argument ***vari** .

keyword variable length of arguments: Consider a situation, when you want to pass argument as "key=value" format. See the example below.

Argument Pass by reference or value: Call by reference means passing the address of a variable where the actual value is stored. All arguments in the Python language are passed by reference. The called Python function uses the value stored in the passed address; any changes to it do affect the source variable.

Example:

```
def ref(list1):
  list1.extend([23,89])
  print ("list inside the function: ",list1 )
list1 = [12,67,90]
print("list before pass", list1)
ref(list1)
print ("list outside the function", list1)
```

Let's check the output with code

call by reference: List extends inside the Python function but the change also reflected back in the calling function. Another Example:

```
def fun(a):
  a=a+4
```

```
    print ("Inside the function", a )
a= 10
fun(a)
print ("Outside the function", a)
```

call by reference: After seeing above example you think it is the example of call by value because change inside the Python function did not reflect back to calling function. At this situation still I would say it is call by reference because inside the function I made new assignment that is a= a+4 . We imagine that a = a + 4 is changing the number stored in a, but it's really reassigning a to point to a new object

Scope of Variables :You may not access all variables of program from all locations. This depends upon where a variable has been declared. There are two basic scopes of variables in Python:

1. ***Local variables***
2. ***Global variables***

Local variableis defined inside the Python function. Local variables are only accessible with in their local scope.

Global variableis defined outside the Python function. Global variables are only accessible throughout the program.

Variable **a** is global variable its value remain same outside the function and inside the function. But if you reassign a inside the function ? Let us discuss with example

```
def fun():
  a =12
  print ("Inside the function", a)
a= 10
fun()
print ("Outside a",a)
```

Let us check out put with pictorial diagram.

```
# global in nested function
def add():
    x = 15
    def change():
        global x
        x = 20
    print("Before making changing: ", x)
    print("Making change")
    change()
    print("After making change: ", x)
add()
print("value of x",x)
```

```
Before making changing:  15
Making change
After making change:  20
value of x 20
```

Global reassigning in local: If you reassign global variable inside the function it does not reflect outside the Python function If local variable is accessed outside the function. Let us discuss in next example

```
def fun():
  a =12
  print ("a inside the function",a )
fun()
print ("Outside a",a)
```

Local variable accessing from outside the function: It give error if you access local variable from outside the function. Consider a situation when you feel that change inside the function should reflect outside the Python function. In this situation you would explicit define **global** keyword as shown in example below.

```
def ref():
  global k
  k=k+7
  print (k )
k=10
ref()
print ("k outside ",k)
```

global variable: In above example after calling function value of k changed because inside the function k is defined as gloabal variable. The statement **global VarName** tells Python that **VarName** is a **global** variable. Python stops searching the local namespace for the variable.

Unsolved Programs

1. Write a program using function to mul. 3 numbers.
2. Write a program using function to avg 5 numbers.
3. Write a program using function to calculate simple interest
4. Write a program using function to find no. Is –ve or +ve
5. Write a program using function to find grater no in x and y
6. Write a program using function to factorial of a given no.
7. Write a program using function to no is prime or not
8. Write a program using function to display result x^n.
9. Write a program using function to display result $(x+1)^n$
10. Write a program using function to display number from 1 to n.
11. Write a program using function to display number from n to m.
12. Write a program using function to display even/odd numbers from 1 to n.
13. Write a program using function to display table of a given numbers.
14. Write a menu driven program to calculate :
    Area of circle[$A=\pi r^2$]
    Area of squire [A=a*a]
    Area of rectangle[A=l*b]
15. Write a menu driven program to calculate:
    Volume of cube[$V=s^3$]
    Volume of sphere[$V=\frac{4}{3}\pi r^3$]
    Volume of cuboid[V=l*b*h]

#1. MULTIPLY 3 NUMBERS

```
def mul (a , b, c):
    return a*b*c
a=int(input("Enter a: "))
b=int(input("Enter b: "))
c=int(input("Enter c: "))
Result= mul (a , b, c)
print(Result)
```

```
Enter a: 3
Enter b: 2
Enter c: 3
18
```

#2.AVERAGE 5 NUMBERS

```
def Avg(a , b, c, d, e):
    return (a+ b+ c+ d+ e)/5
a=int(input("Enter a: "))
b=int(input("Enter b: "))
c=int(input("Enter c: "))
d=int(input("Enter d: "))
e=int(input("Enter e: "))
Result= Avg(a, b, c, d, e)
print(Result)
```

```
Enter a: 2
Enter b: 3
Enter c: 4
Enter d: 5
Enter e: 1
3.0
```

#4. NUMBER IS –VE OR +VE

```
def check(a):
    if a<0:
        print("-ve")
    else:
        print("+ve")
a=int(input("Enter a: "))
check(a)
```

```
Enter a: -9
-ve
```

CODE

##1.
```
def mul (a , b, c):
    return a*b*c
a=int(input("Enter a: "))
b=int(input("Enter b: "))
c=int(input("Enter c: "))
Result= mul (a , b, c)
print(Result)
```
##2.
```
def Avg(a , b, c, d, e):
    return (a+ b+ c+ d+ e)/5
a=int(input("Enter a: "))
b=int(input("Enter b: "))
c=int(input("Enter c: "))
d=int(input("Enter d: "))
e=int(input("Enter e: "))
Result= Avg(a, b, c, d, e)
print(Result)
```
##3.
```
def SI(p,r,t):
    return (p*r*t)/100
p=int(input("Enter P: "))
r=int(input("Enter R: "))
t=int(input("Enter T: "))
Result= SI(p,r,t)
print(Result)
```
##4.
```
def check(a):
    if a<0:
        print("-ve")
    else:
        print("+ve")
a=int(input("Enter a: "))
check(a)
```
##5.
```
def GN(a,b):
    if a<b:
        print(b," is G")
    else:
        print(a," is G")
a=int(input("Enter a: "))
b=int(input("Enter b: "))
GN(a,b)
```
##6.
```
def fact(a):
    fact=1
    for i in range(1,a+1):
        fact*=i
    print("Fact:",fact)
a=int(input("Enter a: "))
fact(a)
```
##7.
```
def prime(a):
    c=1
    for i in range(1,a+1):
        if a%i==0:
            c+=1
    if c==2:
        print(a," is Prime")
    else:
        print(a," is not Prime")
a=int(input("Enter a: "))
prime(a)
```
##8.
```
def POW(x,n):
    print(x**n)
x=int(input("Enter x: "))
n=int(input("Enter n: "))
POW(x,n)
```
##9.
```
def POW(x,n):
    print((x+1)**n)
x=int(input("Enter x: "))
n=int(input("Enter n: "))
POW(x,n)
```
##10.
```
def DIS(a):
    for i in range(1,a+1):
        print(i)
a=int(input("Enter a: "))
DIS(a)
```
##11.
```
def DIS(a,b):
    for i in range(a,b+1):
        print(i)
n=int(input("Enter n: "))
m=int(input("Enter m: "))
DIS(n,m)
```
##12.
```
def DIS(a):
    for i in range(1,a+1):
        if i%2==0:
            print(i, " EVEN")
        else:
            print(i, " ODD")
n=int(input("Enter n: "))
DIS(n)
```
##13.
```
def DIS(a):
    for i in range(1,11):
        print(i*a)
a=int(input("Enter a: "))
DIS(a)
```
##14.
```
def Area_of_circle(r):
    A=3.14*r**2
    return A
def Area_of_squire(a):
    A=a*a
    return A
def Area_of_rectangle(l,b):
    A=l*b
    return A
n=int(input("Enter your choice N: "))
if n==1:
    r=int(input("Enter r: "))
    print(Area_of_circle(r))
else if n==2:
    a=int(input("Enter a: "))
    print(Area_of_squire(a))
else if n==2:
    l=int(input("Enter l: "))
    b=int(input("Enter b: "))
    print(Area_of_rectangle(l,b))
else:
    print("Wrong Choice")
```
##15.
```
def Volume_of_cube(s):
    V=s**3
    return V
def Volume_of_cube(r):
    V=(4/3)*3.14*r**3]
    return V
def Volume_of_cuboid(l,b,h):
    V=l*b*h
    return V
n=int(input("Enter your choice N: "))
if n==1:
    s=int(input("Enter s: "))
    print(Volume_of_cube(s))
else if n==2:
    r=int(input("Enter r: "))
    print(Volume_of_cube(r))
else if n==2:
    l=int(input("Enter l: "))
    b=int(input("Enter b: "))
    h=int(input("Enter h: "))
    print(Volume_of_cuboid(l,b,h))
else:
    print("Wrong Choice")
```


New Assignments Functions using Python

INTRODUCTION AND TYPE AND CALLING OF FUNCTIUONS

#1.Default function used in : Display Static Data/Menu Display/Choice Display

```
def FUN():        #definartion
    print("Python")
    print("is")
    print("FUN!")
FUN()             #calling
```

```
#2. Parameterize/ Argumenta function used in : \nperform operation's")
def FUN(a):       #definartion
   print("a is %d"%a)
   print("a*a is %d"%(a*a))
   print("a**a is %d"%(a**a))
FUN(5)            #calling
#3. Return Single Argument 1
def FUN(a):   #definartion
   return a*a
print(FUN(5))         #calling
#4. Return Single Argument 2
def FUN(a):   #definartion
   return a*a
r=FUN(5)      #calling
print(r)
#5. Return Multiple Argument 1
def FUN(a):   #definartion
   return a*a,a**3
print(FUN(5))         #calling
#6. Return Multiple Argument 2
def FUN(a):   #definartion
   return a*a,a**3
r1,r2=FUN(5)
print(r1,", ",r2)         #calling
#7. Return Multiple Argument 2 using index
def FUN(a):   #definartion
   return a*a,a**3
r1=FUN(5)[0]
r1=FUN(5)[1]
print(r1,", ",r2)         #calling
#8. Function Arguments  LIST
def FUN(a):   #definartion
   print(a)
D=[1,2,3,5]
FUN(D)            #calling
#9. Function Arguments  Tuple
def FUN(a):   #definartion
   print(a)
D=(1,2,3,5)
FUN(D)            #calling
#10. Function Arguments  String
def FUN(a):   #definartion
   print(a)
D="DotPyEdu"
FUN(D)            #calling
print("11. Function Arguments  Dictionary
def FUN(a):   #definartion
   print(a)
D={1:"PYTHON",2:"is",3:"Fun",5:"Function's"}
FUN(D)            #calling
#12. Function with de  fult Argument
def FUN(a=10):   #definartion
   print(a)
a=4
print(a)
FUN(a)            #calling
print(a)
#13. Function with defult Argument
def FUN(a,b=10):   #definartion
   print(a,",",b)
a,b=4,13
print(a,",",b)
FUN(a,b)          #calling
FUN(a)            #calling
FUN(b)            #calling
FUN(b,a)          #calling
```

OUTPUT

```
1. Default function used in :
Display Static Data/Menu Display/Choice Display
Python
is
FUN!
2.  Parameterize/ Argumenta function used in :
perform operation's
a is 5
a*a is 25
a**a is 3125
3. Return Single Argument 1
25
4. Return Single Argument 2
25
4. Return Multiple Argument 1
(25, 125)
6. Return Multiple Argument 2
25 ,  125
7. Return Multiple Argument 2 using index
125 ,  125
8.  Function Arguments  LIST
[1, 2, 3, 5]
9. Function Arguments  Tuple
(1, 2, 3, 5)
10. Function Arguments  String
DotPyEdu
11. Function Arguments  Dictionary
{1: 'PYTHON', 2: 'is', 3: 'Fun', 5: "Function's"}
12. Function with defult Argument
4
4
4
13. Function with defult Argument
4 , 13
4 , 13
4 , 10
13 , 10
13 , 4
```

ASSIGNMENT OF FUNCTION

Q1:
```
def switchover(A,N, split):
    K=0
    while K<N:
        if K<split :
            A[K]+= K
        else:
            A[K]*= K
        K+=1
def display(A,N):
    K=0
    while K<N:
        if K%2== 0:
            print(A[K],end="%")
        else:
            print(A[K],"\n")
        K+=1
H= [30,40,50,20,10,5]
switchover(H,6,3)
print(H)
display(H,6)
```
Q2:
```
def s(a, b):
    a+=2
    b*=5
    print(a,":",b,"\n")
a,b=4,8
print(a,":",b,"\n")
s(a,b)
print(a,":",b,"\n")
```
Q3:
```
def s(a, b):
    a+=2
    b[0]=b[0]+5
    print(a,":",b,"\n")
a,b=4,[8]
print(a,":",b,"\n")
s(a,b)
print(a,":",b,"\n")
```
Q4:
```
def s(a, b):
    a[0]+=2
    b[0]+=5
    print(a,":",b,"\n")
a,b=[4],[8]
print(a,":",b,"\n")
s(a,b)
print(a,":",b,"\n")
```
Q5:
```
def s(a, b=7):
    a+=2
    b*=5
    print(a,":",b,"\n")
a,b=4,8
print(a,":",b,"\n")
s(a,b)
print(a,":",b,"\n")
s(a)
print(a,":",b,"\n")
```
Q6:
```
def s(a=6, b=7):
    a+=2
    b*=5
    print(a,":",b,"\n")
a,b=4,8
print(a,":",b,"\n")
s(a,b)
print(a,":",b,"\n")
s(a)
print(a,":",b,"\n")
s()
print(a,":",b,"\n")
```
Q7:
```
def func(x,y):
    if x[0]%y[0]==0 :
        return x[0]+2
    else:
        return y[0]-1
p,q=[20],[23];
print(p,",",q)
q=func(p,q)
print(p,",",q)
```
Q8:
```
def func(x,y=10):
    if x%y==0 :
        return x+2
    else:
        return y-1
p,q=20,23;
print(p,",",q)
q=func(p,q)
print(p,",",q)
```
Q9:
```
def func(x,y=10):
    if x%y==0 :
        return x+2
    else:
        return y-1
p,q=20,23;
print(p,",",q)
q=func(p,q)
print(p,",",q)
p=func(q)
print(p,",",q)
```
Q10:
```
def func(x,y=10):
    if x%y==0 :
        return x+2
    else:
        return y-1
p,q=20,23;
print(p,",",q)
q=func(p,q)
print(p,",",q)
p=func(q)
print(p,",",q)
q=func(p)
print(p,",",q)
```
Q11:
```
def execute(x,y=200):
    temp=x+y
    x+=temp
    if(y!=200):
        print(temp,",",x,",",y)
a,b =50,20
execute(b)
print(a,",",b)
execute(a,b)
print(a,",",b)
```
Q12:
```
def execute(x,y=15):
    temp=x+y
    x+=temp
    if(y<=20):
        print(temp,",",x,",",y)
a,b =50,20
execute(b)
print(a,",",b)
execute(b)
print(a,",",b)
```
Q13:
```
def execute(x,y=20):
    temp=x*y
    x+=temp
    if(y>=20):
        print(temp,",",x,",",y)
a,b =50,20
execute(b)
print(a,",",b)
execute(a)
print(a,",",b)
```

OUTPUT

Q1.
[30,41,52,60,40,25]
30%41
52%60
40%25
Q2.
4 : 8
6 : 40
4 : 8
Q3.
4 : [8]
6 : [13]
4 : [13]
Q4.
[4] : [8]
[6] : [13]
[6] : [13]
Q5.
4 : 8
6 : 40
4 : 8
6 : 35
4 : 8
Q6.
4 : 8
6 : 40
4 : 8
6 : 35
4 : 8
8 : 35
4 : 8
Q7.
[20] , [23]
[20] , 22
Q8.
20 , 23
20 , 22
Q9.
20 , 23
20 , 22
9 , 22
Q10.
20 , 23
20 , 22
9 , 22
9 , 9
Q11.
50 , 20
70 ,120 ,20,50 ,20
Q12.
35 , 55 , 15
50 , 20
35 , 55 , 15
50 , 20
Q13.
400,420 ,20,
50,20
1000 , 1050 ,20
50 , 20

===UNSOLVED ===

Q1.
```
def Changer(P,Q=10):
    P=P/Q
    Q=P%Q
    print (P,"#",Q)
    return P
A,B=200,20
A=Changer(A,B)
print (A,"$",B)
B=Changer(B)
print (A,"$",B)
A=Changer(A)
print (A,"$",B)
```

Q2. Find the output:

1.
```
import random
    n=random.randrange(10,3,-2)
    print (n )
```

2.
```
import re
    r1=re.finditer('sing{0,2}','singers singg well')
    for I in r1:
        print (I.group(), I.span() )
```

3.
```
str='Educating the world in different spheres'
    print ("The position of 'in' at:", str.find('in') )
```

4.
```
str='board exam'
    print (str.split('a') )
```

5.
```
p=[1,7,2,6,4,9]
    p.insert(3,2)
    print (p)
```

6.
```
n=[1,4,2,6,9,5]
    print (n[:4]) print n[-1:-4:-2]
```

Q3. Find the Output:

(i)
```
def small(a,b):
  if a<b:
    return a
  else:
    if a==b:
      return 0
    else:
      return b
def main():
  x,y=5,8
  z=small(x,y)
  print( "x=",x,"y=",y )
  y=z-1
  z=small(y,x)
  print (x,y )
main()
```

ii)
```
print int (7.0+0.1) (1 ½ )
print (str (1.2 * 3.4) )
print (float ("77"+".0") )
```

Q4. Find and write the output of the following python code:

```
def Change(P ,Q=30):
P=P+Q
Q=P-Q
print( P,"#",Q)
return (P)
R,S=150,100
R=Change(R,S)
print(R,"#",S)
S=Change(S)
```

=======OUTPUT=======

OUTPUT 1:
10.0 # 10.0
10.0 $ 20
2.0 # 2.0
10.0 $ 2.0
1.0 # 1.0
1.0 $ 2.0

OUTPUT 2:
10
sing (0, 4)
singg (8, 13)
The position of 'in' at: 6
['bo', 'rd ex', 'm']
[1, 7, 2, 2, 6, 4, 9]
[1, 4, 2, 6]

OUTPUT 3:
(i)
x= 5 y= 8
5 4
(ii)
7.1
4.08
77.0

OUTPUT 4:
250 # 150
250 # 100
130 # 100