```
# loops - for loop and while loop,
# loops -
# for var in dict:
#    statements

# we can use conditional statements inside the loop
# loop inside loop (nested loops)
# break, continue , zip, enumerate

my_list1 = [10,12,19,20]
list(enumerate(my_list1))
```

```
[(0, 10), (1, 12), (2, 19), (3, 20)]
```

```
# zip
my_list2 = [1,2,3,4]
list(zip(my_list1, my_list2))
```

```
[(10, 1), (12, 2), (19, 3), (20, 4)]
```

```
x, y = (10,20)
x
```

```
10
```

```
y
```

```
20
```

```
for x,y in zip(my_list1,my_list2):
    print(x+y)
```

```
11
14
22
24
```

```
dict(zip(my_list1, my_list2))
```

```
{10: 1, 12: 2, 19: 3, 20: 4}
```

```
# While loop -
# while <condition> :
# statements



scores = [82, 30, 56, 23, 87]
p = 0
f = 0
for m in scores:
    if m >= 33:
        p = p + 1
```

```
    else:
        f = f+1

print('pass count', p)
print('fail count', f)
```

```
pass count 3
fail count 2
```

```
n = 3
factorial = 1

for i in range(1,n+1):
    factorial = factorial * i # fac = 1 # fac =  2 # fac = 6

print(factorial)
```

```
6
```

```
# list comprehension and one liners in python

# write a program to check if a number is even or odd
num = 23

if num%2 == 0 :
    print("even")

num = 22

"even" if num%2 == 0 else "odd"  # one liner conditional statements in
python
```

```
'even'
```

```
num = 23
"even" if num%2 == 0 # else part is compoulsory in one liner
conditional statements
```

```
  File
"C:\Users\maanz\AppData\Local\Temp/ipykernel_14828/3583273240.py",
line 2
    "even" if num%2 == 0
                       ^
SyntaxError: invalid syntax
```

```
num = 22
if num%2 ==0 :
```

```
    num = num + 10
    num = num - 2
    print(num)
```

```
30
```

```
num = 22
num+10  if num%2==0  else num-2  # in one liners only 1 operation or statements is possible
```

```
32
```

```
# write a one liner conditional statement to check if a student has passed or failed
# >= 33 marks is passed else fail
marks = 76
"pass" if marks>= 33 else "fail"
```

```
'pass'
```

```
# list comprehension

# write a program to find the list of all even number b/w 2-20

even_list = []
for i in range(2,21,2):

    even_list.append(i)

even_list
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
```

```
# list comprehension -
[ i     for i in range(2,21,2)  ]
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
```

```
scores
```

```
[82, 30, 56, 23, 87]
```

```
[ "pass" if m>=33 else "fail"   for m in scores]
```

```
['pass', 'fail', 'pass', 'fail', 'pass']
```

```
#[final_output       loop]

pass_fail_list = [ "pass" if m>=33 else "fail"   for m in scores]
pass_fail_list
```

```
['pass', 'fail', 'pass', 'fail', 'pass']
```

```
pass_fail_list.count('fail')
```

```
2
```

```
#
product_price = [1000,500, 2000, 2500, 650]

# apply discount of 15% on product whose price is more than 1500 and
# 10% discount on product whose price is less than 1500
# store the discounted price in a list

discounted_price = []

for x in product_price:
    if x > 1500:
        discounted_price.append(x - (x*0.15) )
    elif x < 1500:
        discounted_price.append(x - (x*0.1))

print(discounted_price)
```

```
[900.0, 450.0, 1700.0, 2125.0, 585.0]
```

```
[   i          for i in range(2,10, 2) ]
```

```
[2, 4, 6, 8]
```

```
# one liner - conditional statements
num = 31
if num%2 ==0 :
    print("even")
else:
    print("odd")
```

```
odd
```

```
"even" if num%2 ==0 else "odd"
```

```
'odd'
```

```
product_price = [1000,500, 2000, 2500, 650]
product_price
```

```
[1000, 500, 2000, 2500, 650]
```

```
[ price - (price*0.15) if price >=1500  else price - (price *0.1)
for price in product_price]
```

```
[900.0, 450.0, 1700.0, 2125.0, 585.0]
```

```
# filtering the values
state_name = ['Karnataka', 'TamilNadu', 'Maharashtra', 'Gujarat',
'Goa', 'Kerala']

# filter the state names which starts with 'G'

states_with_g = []

for state in state_name:
    if state.startswith('G'):
        states_with_g.append(state)
states_with_g
```

```
['Gujarat', 'Goa']
```

```
# [  final_output    for loop       filter operation]

[  state      for state in state_name        if state.startswith('G')]
```

```
['Gujarat', 'Goa']
```

```
# using list comprehension print the list of numbers b/w 3,30 which
are divisible by both 3 and 5

number_list = []

for x in range(3,31):
    if x%3 ==0 and x%5 == 0:
        number_list.append(x)

print(number_list)
```

```
[15, 30]
```

```
[  x     for x in range(3,31)   if x%3 ==0 and x%5 == 0         ]
```

```
[15, 30]
```

```
state_name = ['Karnataka', 'TamilNadu', 'Maharashtra', 'Gujarat',
'Goa', 'Kerala']

# filter the state names whose name's length is >7

state = []
for s in state_name:
    if len(s) > 7:
        state.append(s)
print(state)
```

```
['Karnataka', 'TamilNadu', 'Maharashtra']
```

```
[   s    for s in state_name     if len(s) > 7 ]
```

```
['Karnataka', 'TamilNadu', 'Maharashtra']
```

```
# dict comprehension


{ i : i**2         for i in range(1,10)   }
```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81}
```

```
{ i : "even" if i%2 ==0 else "odd"         for i in range(1,10)   }
```

```
{1: 'odd',
 2: 'even',
 3: 'odd',
 4: 'even',
 5: 'odd',
 6: 'even',
 7: 'odd',
 8: 'even',
 9: 'odd'}
```

```


# functions - helps to write non repetitive codes

num1 = 10
num2 = 20
result = num1+num2
print(result)
```

```
30
```

```
num3 = 50
num4 = 5
result2 = num3 + num4
print(result2)
```

```
55
```

```
# creating a function
# def <name>(arguments):
#     statements
#  return

# function defination

def addition(num1, num2):
    result = num1 + num2
    return result


# function call
addition(10, 20)
```

```
30
```

```
addition(100, 200)
```

```
300
```

```
# write a function for greeting a person with Good Morning, Name

# greet('John') -> Good Morning, John

def greet(name):
    return f"Good Morning, {name}"
```

```
greet("Manvendra")
```

```
'Good Morning, Manvendra'
```

```
greet("John")
```

```
'Good Morning, John'
```

```
# write a function to check if a number is even or odd - evenorodd

# evenorodd(20) -> even

def evenorodd(num):
    if num%2 == 0:
        return "even"
    else:
        return "odd"
```

```
evenorodd(10)
```

```
'even'
```

```
evenorodd(23)
```

```
'odd'
```

```
# write a function to check if a student has passed or failed
# passorfail(45)  -> pass

def passorfail(marks):
    if marks >= 33:
        return "pass"
    else:
        return "fail"
```

```
passorfail(80)
```

```
'pass'
```

```
# min, max, sum - inbuilt functions

# min(10,20,9,23)

# write a function which will have 2 arguments - price and
discount_percent
# the output of the function will be -> discounted price (price after
apply the discount)

# discountedprice(500, 0.1)  -> 450

def discountedprice(price, discount_per):
    new_price = price - (price * discount_per)
    return new_price

discountedprice(500, 0.1)
```

```
450.0
```

```
discountedprice(500, 0.15)
```

```
425.0
```

```
discountedprice(5000, 0.15)
```

```
4250.0
```

```
discountedprice(0.15, 500)   #positional arguments
```

```
-74.85
```

```
# keyword arguments
discountedprice(discount_per = 0.15, price = 500)
```

```
425.0
```

```
# default arguments

def discountedprice(price, discount_per = 0.1):
    new_price = price - (price * discount_per)
    return new_price

discountedprice(500)
```

```
450.0
```

```
discountedprice(5000)
```

```
4500.0
```

```
discountedprice(5000, 0.15)
```

```
4250.0
```

```
discountedprice(5000)
```

```
4500.0
```

```
addition(10,20)
```

```
30
```

```
min(10,2,7,8)
```

```
2
```

```
addition(10,20,30)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_14828/2121680615.py in <module>
----> 1 addition(10,20,30)

TypeError: addition() takes 2 positional arguments but 3 were given
```

```
min(10,20, 60, 2)  # variable length arguments
```

```
2
```

```
def varlenfunc(*a): # variable length arguments
    return a
```

```
varlenfunc(10,2,5,6,7,9, 12, 15,18)
```

```
(10, 2, 5, 6, 7, 9, 12, 15, 18)
```

```
addition(10,20, 40)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_14828/818344951.py in <module>
----> 1 addition(10,20, 40)

TypeError: addition() takes 2 positional arguments but 3 were given
```

```
#keyword variable length arguments
def keyvarfunc(**a):
    return a
```

```
keyvarfunc(s1 = 30, s2 = 40, s3 = 60, s4 = 44)
```

```
{'s1': 30, 's2': 40, 's3': 60, 's4': 44}
```

```
# lambda function -
# lambda is onliner function - ananomyous function

add = lambda x, y  :   x+y
add(10,20)
```

```
30
```

```
# using lambda function find the square of a number
# square(2)  -> 4

square = lambda x :  x**2
square(3)
```

```
9
```

```
# lambda with conditional statements
# check is a num is even or odd using lambda function
eveodd = lambda x : "even"  if x%2 == 0 else "odd"
eveodd(12)
```

```
'even'
```

```
# check if a student pass or fail >= 33 pass else fail using lambda
function
passorfail = lambda x : "pass"  if x>=33 else "fail"
passorfail(89)
```

```
'pass'
```

```
# lambda functions are mostly used with map and filter

# filter
scores
```

```
[82, 30, 56, 23, 87]
```

```
list(filter(  lambda x : x < 33  ,  scores ))
```

```
[30, 23]
```

```
# using filter function - from scores filter the even numbers
list(filter( lambda x : x%2 == 0 , scores ))
```

```
[82, 30, 56]
```

```
state_name  # filter the states which starts with G
```

```
['Karnataka', 'TamilNadu', 'Maharashtra', 'Gujarat', 'Goa', 'Kerala']
```

```
list(filter(lambda x : x.startswith('G') ,  state_name))
```

```
['Gujarat', 'Goa']
```

```
# filter the states which len is >7
list(filter(lambda x : len(x) > 7 ,  state_name))
```

```
['Karnataka', 'TamilNadu', 'Maharashtra']
```

```
state_name
```

```
['Karnataka', 'TamilNadu', 'Maharashtra', 'Gujarat', 'Goa', 'Kerala']
```

```
# map
list(map(  lambda x : len(x)  ,  state_name))
```

```
[9, 9, 11, 7, 3, 6]
```

```
scores = [82, 30, 56, 23, 87]
scores
```

```
[82, 30, 56, 23, 87]
```

```
# using the map function, map the values in score as pass or fail

list(map(lambda x : 'pass'  if x>33 else 'fail' ,  scores))
```

```
['pass', 'fail', 'pass', 'fail', 'pass']
```

```
# from the given list filter the even numbers using filter function
# and then map them into it's squares using map

num_list = [10,12,6,9,15,8]

# [10,12,6,8]
# [100, 144, 36, 64]

a = list(filter(lambda x : x%2==0 , num_list))
list(map(lambda x : x**2, a))
```

```
[100, 144, 36, 64]
```

```
student_name = ['Ram', 'Rahul' ,'johnson', 'john']
student_name
```

```
['Ram', 'Rahul', 'johnson', 'john']
```

```
list(filter(lambda x : x.endswith('n'), student_name))
```

```
['johnson', 'john']
```

```
# hackerrank
```