Python Basics

In [ ]:

```
'''

Hridoy Ahmed
Daffodil International University
Dept. of Computer Science & Engineering
Web: hridoyahmed.pythonanywhere.com
Email: hridoy15-7981@diu.edu.bd
Facebook Profile: https://web.facebook.com/HridoyAhmedCSE

'''
```

========================= Python Data Types ==========================

In [ ]:

```

```

In [ ]:

```
'''
int
float
double
bool
string
list
tuple
dict
set
etc.
'''
```

In [1]:

```
# Integer
number = 10
```

In [2]:

```
# Float
number2 = 10.5
```

In [3]:

```
# Double
number3 = 100.58
```

In [4]:

```
# Boolean
'''
0 = False
1 = True
'''
isOdd = True
isEven = 0
```

In [5]:

```
# String
'''
Strings is immutable.
Index diye new value Assign kora jay na.
'''

my_name = "Hridoy Ahmed"
```

In [6]:

```
# List
'''
List is mutable.
Index diye new value Assign kora jay.
'''

my_list = [10, 15, "Char", "String", 100.5]
print(my_list)
```

```
[10, 15, 'Char', 'String', 100.5]
```

In [7]:

```
my_list[0] = 50
```

In [8]:

```
print(my_list)
```

```
[50, 15, 'Char', 'String', 100.5]
```

In [9]:

```
# Tuple

'''
Tuple is immutable.
Index diye new value Assign kora jay na.
'''
my_tuple = (10, 15, 20, 50)
print(my_tuple)
```

```
(10, 15, 20, 50)
```

In [10]:

```
my_tuple[0] = 5
```

```
----------------------------------------------------------------------
-----
TypeError                                 Traceback (most recent call
 last)
<ipython-input-10-7202c080a1da> in <module>
----> 1 my_tuple[0] = 5

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

In [11]:

```
# Dictionary

'''
Key and Value Thakbe.
'''

my_dict = {
    # Key : Value
    "name" : "Hridoy",
    "Age" : 25
}

print(my_dict["Age"])
```

```
25
```

In [12]:

```
print(my_dict)
```

```
{'name': 'Hridoy', 'Age': 25}
```

In [13]:

```
# Dictionary inside List
```

In [14]:

```
my_list = [
    # Index 0
    {
        "name" : "Hridoy",
        "Age" : 25
    },

    # Index 1
    {
        # Index           0    1    2

        "Number_List" : [100, 200, 300],
        "Strings" : "psychoCoders",
        "Booleans" : True
    }
]
```

In [15]:

```
# To Print 300
print(my_list[1]["Number_List"][2])
```

```
300
```

In [16]:

```
# Set

'''
Unique value
'''

my_set = {1, 1, 5, 5, 2, 3, 4}
print(my_set)
```

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

In [ ]:

```

```

In [ ]:

```
========================= Strings Advanced ================================
```

In [ ]:

```
# Docstring

'''
Docstring 3 ta dot er por likhte hoy
r 3 ta dot diye shesh korte hoy eirokom vabe
'''
```

In [17]:

```
# Concetanation

'''
Ek string er sathe arek string add kora lagbe.
string ar number ba onno kuno data type concat
hbe na.
'''

firstname = "Hridoy"
space = " "
lastname = "Ahmed"

print(firstname + space + lastname)
```

```
Hridoy Ahmed
```

In [18]:

```
# Type conversion

number = str(100)

print(type(number))
```

```
<class 'str'>
```

In [19]:

```
number = int(number)
print(type(number))
```

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

In [20]:

```
# Escape Sequence

cat_string = 'It's a cat'
```

```
  File "<ipython-input-20-a2001f933070>", line 3
    cat_string = 'It's a cat'
                     ^
SyntaxError: invalid syntax
```

In [21]:

```

'''
' ba " er age \ dite hoy.
orthat \ er porer symbol count hobe.
\ na dile count hbe na.
\t(tab) and \n(new line) betikrom.
'''



cat_string = 'It\'s a cat'
print(cat_string)
```

```
It's a cat
```

In [22]:

```
weather = "It\'s a \"Kind of\" Sunny"
print(weather)
```

```
It's a "Kind of" Sunny
```

In [23]:

```
# Formatted String

firstname = "Hridoy"
lastname = "Ahmed"
country = "Bangladesh"

print(f'Welcome {firstname} {lastname} from {country}')

# Or

print("Welcome {0} {1} from {2}".format(firstname, lastname, country))
```

```
Welcome Hridoy Ahmed from Bangladesh
Welcome Hridoy Ahmed from Bangladesh
```

In [24]:

```
 # Indexing & Slicing

'''
Starts counting from 0 index.
'''
#         -5-4-3-2-1
#            012345
my_string = "Hridoy"

print(my_string[4])
print(my_string[-1])
```

```
o
y
```

In [25]:

```
'''
start:end:stepover
shuru:ag porjnto:koy ghor por por
'''
#            01234567891011
my_string = "Hridoy Ahmed"

print(my_string[1:9:1])
print(my_string[1:10:2])
```

```
ridoy Ah
rdyAm
```

In [26]:

```
print(my_string[:6])
```

```
Hridoy
```

In [27]:

```
# Reverse
print(my_string[::-1])
```

```
demhA yodirH
```

In [28]:

```
print(my_string[:6:2])
```

```
Hio
```

In [29]:

```
print(my_string[:])
```

```
Hridoy Ahmed
```

In [ ]:

```

```

In [30]:

```
# Built-in functions

# Length     123456
my_string = "Hridoy"

print(len(my_string))
```

```
6
```

In [ ]:

```
# Methods

'''
Original string a poriborton kore na
'''
```

In [31]:

```
new_string = "Dhur baal Sikhar dorkar sikh"

print(new_string.upper())
```

```
DHUR BAAL SIKHAR DORKAR SIKH
```

In [32]:
```
print(new_string.lower())
```

```
dhur baal sikhar dorkar sikh
```

In [33]:
```
print(new_string.capitalize())
```

```
Dhur baal sikhar dorkar sikh
```

In [34]:
```
print(new_string.find("baal"))
```

```
5
```

In [35]:
```
print(new_string.replace("baal", "vai"))
```

```
Dhur vai Sikhar dorkar sikh
```

In [36]:
```
# Anyway, Methods don't change the original strings.
print(new_string)
```

```
Dhur baal Sikhar dorkar sikh
```

In [ ]:
```

```

In [ ]:
```
========================= Lists Advanced ==============================
```

In [37]:
```
# List Slicing

# Index     0   1      2          3(0)      3(1)     3(2)
new_list = [10, 20, "Strings", ["New List", 540, "New_Strings"]]

print(new_list)
```

```
[10, 20, 'Strings', ['New List', 540, 'New_Strings']]
```

In [38]:
```
print(new_list[3][2])
```

```
New_Strings
```

In [39]:
```
# Start : End : Stepover

print(new_list[:2])
```

```
[10, 20]
```

In [40]:
```
print(new_list[::-1])
```

```
[['New List', 540, 'New_Strings'], 'Strings', 20, 10]
```

In [41]:
```
new_list[0] = 500
```

In [42]:

```
print(new_list)
```

```
[500, 20, 'Strings', ['New List', 540, 'New_Strings']]
```

In [ ]:

```

```

In [43]:

```
# Methods
```

In [44]:

```
# Adding
```

In [45]:

```
item_list = []
```

In [46]:

```
item_list.append(10)
```

In [47]:

```
print(item_list)
```

```
[10]
```

In [48]:

```
item_list.append(20)
item_list.append(30)
item_list.append(40)
item_list.append(50)
```

In [49]:

```
print(item_list)
```

```
[10, 20, 30, 40, 50]
```

In [50]:

```
'''
insert can add value in specific location.
append add value to the last index.
'''

item_list.insert(3, [500, 200, 450])
```

In [51]:

```
print(item_list)
```

```
[10, 20, 30, [500, 200, 450], 40, 50]
```

In [52]:

```
'''
extend can add another list to the last index.
But it removes []
'''

item_list.extend([10, 20, 30])
print(item_list)
```

```
[10, 20, 30, [500, 200, 450], 40, 50, 10, 20, 30]
```

In [ ]:

```

```

In [53]:

```
# Removing
```

In [54]:

```
'''
Pop removes the last item from the list.
It can also remove specific item using indexing.
'''

item_list.pop()
print(item_list)
```

```
[10, 20, 30, [500, 200, 450], 40, 50, 10, 20]
```

In [55]:

```
new_list = item_list.pop(3)
print(item_list)
print(new_list)
```

```
[10, 20, 30, 40, 50, 10, 20]
[500, 200, 450]
```

In [56]:

```
'''
Pop return a value
but remove doesn't return value.
It only removes the first existing value.
'''

new_list = item_list.remove(10)
print(new_list)
```

```
None
```

In [57]:

```
print(item_list)
```

```
[20, 30, 40, 50, 10, 20]
```

In [ ]:

```

```

In [58]:

```
# Index
'''
index(value, start, stop)
'''

print(item_list.index(10))
```

```
4
```

In [59]:

```
'''
We get an error because we start from 0 ==> 20
and stop to 3 ==> 50.
between these values 10 doesn't exit.
'''

print(item_list.index(10, 0, 3))
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-59-e7833e4f3ab0> in <module>
      5 '''
      6 
----> 7 print(item_list.index(10, 0, 3))

ValueError: 10 is not in list
```

In [60]:

```
'''
in keyword gives us boolean values
if value exists in the list then it gives True
otherwise False.
'''
print(10 in item_list)
```

```
True
```

In [61]:

```
'''
count keyword counts how many values are in list
'''
print(item_list.count(20))
```

```
2
```

In [62]:

```
'''
sort keyword sort the list.
It modifies the list.
'''

item_list.sort()
print(item_list)
```

```
[10, 20, 20, 30, 40, 50]
```

In [63]:

```
'''
sorted built in function also sort the list
but it doesn\'t modify the original list
'''
new_list = [20, 10, 50, 40, 100, 5]
print(sorted(new_list))
```

```
[5, 10, 20, 40, 50, 100]
```

In [64]:

```
print(new_list)
```

```
[20, 10, 50, 40, 100, 5]
```

In [67]:

```
'''
reverse keyword reverse the list using index
'''
letter_list = ['d' , 'f', 'a', 'x', 'c']
letter_list.reverse()
print(letter_list)
```

```
['c', 'x', 'a', 'f', 'd']
```

In [68]:

```
'''
For reversing alphabatically.
We have to use sort firstly then reverse.
'''
letter_list.sort()
letter_list.reverse()
print(letter_list)
```

```
['x', 'f', 'd', 'c', 'a']
```

In [69]:

```
# Join

sentence = '!'

new_sentence = sentence.join(['Amr', 'nam', 'Hridoy'])
print(new_sentence)
```

```
Amr!nam!Hridoy
```

In [70]:

```
# Separation
first_list = [10, 20, 30]
second_list = [10, 20, 30]
print(my_list,second_list,sep=':')
```

```
[{'name': 'Hridoy', 'Age': 25}, {'Number_List': [100, 200, 300], 'Stri
ngs': 'psychoCoders', 'Booleans': True}]:[10, 20, 30]
```

In [71]:

```
# List Unpacking

a,b,c = [1, 2, 3]

print(a)
print(b)
print(c)
```

```
1
2
3
```

In [72]:

```
a,b,c, *other, d = [1, 2, 3, 4, 5, 6, 7, 8, 9]

print(a)
print(b)
print(c)
print(other)
print(d)
```

```
1
2
3
[4, 5, 6, 7, 8]
9
```

In [ ]:

In [ ]:

```
========================= Dictionary Advanced =========================
```

In [73]:

```
# Keys

'''
Keys have to be unique.
Otherwise it holds last value
'''
my_dict = {
    'name' : 'hridoy',
    'name': 'shovan'
}

print(my_dict['name'])
```

```
shovan
```

In [74]:

```
# Methods
```

In [75]:

```
'''
get method can insert keys and values
if it doesn't exist yet.
but doesn't modify original dictionary.
'''
user_dict = {
    'name' : 'hridoy',
    'country' : 'BD'
}
print(user_dict.get('Age', 25))
```

```
25
```

In [76]:

```
print(user_dict)
```

```
{'name': 'hridoy', 'country': 'BD'}
```

In [77]:

```
# Dictinary creation alternatively

user2 = dict(name = "Asif", age = 24)
print(user2)
```

```
{'name': 'Asif', 'age': 24}
```

In [78]:

```
print('name' in user2)
```

```
True
```

In [79]:

```
print('country' in user2)
```

```
False
```

In [80]:

```
print('name' in user2.keys())
```

```
True
```

In [81]:

```
print('Asif' in user2.values())
```

```
True
```

In [82]:

```
'''
items gives tuple as output
'''
print(user2.items())
```

```
dict_items([('name', 'Asif'), ('age', 24)])
```

In [83]:

```
'''
clear removes the keys and values from dictionary
'''
user2.clear()
print(user2)
```

```
{}
```

In [84]:

```
'''
copy keyword copies the dictionary
'''
user_dict = {
    'name' : 'hridoy',
    'country' : 'BD'
}
user2 = user_dict.copy()
print(user2)
```

```
{'name': 'hridoy', 'country': 'BD'}
```

In [85]:

```
'''
pop returns the value from the dictionary
and removes the last key value pair
'''
print(user2.pop('country'))
print(user2)
```

```
BD
{'name': 'hridoy'}
```

In [86]:

```
'''
update updates the values of keys if exists or
update new keys and values
'''
user_dict.update({'Age': 25})
print(user_dict)
```

```
{'name': 'hridoy', 'country': 'BD', 'Age': 25}
```

In [ ]:

```

```

In [ ]:

```
=========================== Tuples Advanced ============================
```

In [87]:

```
a,b,c, *other, d = (1, 2, 3, 4, 5, 6, 7, 8, 9)

print(a)
print(b)
print(c)
print(other)
print(d)
```

```
1
2
3
[4, 5, 6, 7, 8]
9
```

In [88]:

```
# Methods
```

In [89]:

```
'''
count keyword dekhbe kotobar value ta tuple a ache
'''

my_tuple = (1, 2, 3, 4, 4, 4, 5, 6, 7, 8, 9)
print(my_tuple.count(4))
```

```
3
```

In [90]:
```
'''
index dekhbe koto number index a prothom value ta ache
'''
print(my_tuple.index(7))
```
```
8
```

In [ ]:
```

```

In [ ]:
```
=========================== Tuples Advanced ===========================
```

In [91]:
```
# Methods
```

In [92]:
```
my_set = {10, 20, 30, 40}
```

In [93]:
```
my_set.add(50)
```

In [94]:
```
print(my_set)
```
```
{40, 10, 50, 20, 30}
```

In [95]:
```
sample_list = [1, 1, 2, 3, 4, 5, 3]

new_set = set(sample_list)
print(new_set)
```
```
{1, 2, 3, 4, 5}
```

In [96]:
```
print(len(new_set))
```
```
5
```

In [97]:
```
print(list(new_set))
```
```
[1, 2, 3, 4, 5]
```

In [98]:
```
another_set = new_set.copy()
print(another_set)
```
```
{1, 2, 3, 4, 5}
```

In [99]:
```
another_set.clear()
print(another_set)
```
```
set()
```

In [100]:
```
my_set = {1, 2, 3, 4, 5}
your_set = {4, 5, 6, 7, 8, 9, 10}

print(my_set.difference(your_set))
```
```
{1, 2, 3}
```

In [101]:

```
my_set.discard(5)
print(my_set)
```

```
{1, 2, 3, 4}
```

In [102]:

```
'''
differences are removed from the set
'''
my_set.difference_update(your_set)
```

In [103]:

```
print(my_set)
```

```
{1, 2, 3}
```

In [104]:

```
'''
intersection gives us common values from two sets.
'''
my_set = {1, 2, 3, 4, 5}
your_set = {4, 5, 6, 7, 8, 9, 10}

print(my_set.intersection(your_set))
# or
print(my_set & your_set)
```

```
{4, 5}
{4, 5}
```

In [105]:

```
'''
Union gives us all values from two sets.
'''
my_set = {1, 2, 3, 4, 5}
your_set = {4, 5, 6, 7, 8, 9, 10}

print(my_set.union(your_set))

# or
print(my_set | your_set)
```

```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

In [106]:

```
'''
isdisjoint return korbe true and false.
common thakle false return korbe na thakle true.
mane hocche check korbe set duitar value unique kina.
'''
my_set = {1, 2, 3, 4, 5}
your_set = {4, 5, 6, 7, 8, 9, 10}

print(my_set.isdisjoint(your_set))
```

```
False
```

In [107]:
```
'''
issubset chcek korbe prothom set ta onno set er ekta ongsho kina.
'''
my_set = {4, 5}
your_set = {4, 5, 6, 7, 8, 9, 10}

print(my_set.issubset(your_set))
```

```
True
```

In [108]:
```
'''
issuperset chcek korbe prothom set tar ongsho onno set kina.
'''
my_set = {4, 5}
your_set = {4, 5, 6, 7, 8, 9, 10}

print(my_set.issuperset(your_set))
```

```
False
```

In [ ]:
```

```

In [ ]:
```
========================== Conditional Operation =======================
```

In [ ]:
```
'''
it (condition):
    execute
elif (condition):
    execute
else:
    execute
'''
```

In [110]:
```
number = 10

if number == 0:
    print('It\'s zero')

elif number > 0:
    print('Positive')

else:
    print('Negative')
```

```
Positive
```

In [ ]:
```

```

In [111]:

```
# Take input from user

'''
By default input() returns a string type.
we have to convert if needed
'''

problem_name = input()
number = int(input())

if (number %2 == 0):
    print('Even Number')
else:
    print('Odd number')
```

```
Even Odd Check
15
Odd number
```

In [ ]:

```

```

In [ ]:

```
========================== Logical Operation ==========================
```

In [ ]:

```
'''
and
or
not
'''
```

In [112]:

```
username = 'hridoy'
password = '1234'

if (username == 'hridoy' and password == '1234'):
    print('Login Successfully')
else:
    print('Login failed')
```

```
Login Successfully
```

```
In [119]: short_range_gun = True
          sniper = False
          expert = False

          if short_range_gun and sniper and expert:
              print("100% Possibility to get chicken dinner")

          elif short_range_gun and expert and not sniper:
              print("70% Possibility to get chicken dinner")

          elif (short_range_gun or sniper) and not expert:
              print("Less Possibility to get chicken dinner")

          else:
              print("You are a noob")
```

```
Less Possibility to get chicken dinner
```

```
In [ ]:
```

```
In [ ]: ============================== Loops ===================================
```

```
In [120]: # For loops
```

```
In [123]: item_list = ['a', 'b', 'c', 'd']

          '''
          item er jaygay je kuno var neya jabe
          like i, j, k, l etc.
          '''
          for item in item_list:
              print(item)
```

```
a
b
c
d
```

In [124]:

```
'''
range(3)
By default range starts from 0.
It iterates from 0 to 3 here. // 0 1 2
Amra jodi chai j 5 theke start hoye 15 er ag
porjnto cholbe taile range(5, 15) likhte hbe.

range(start, end, stepover) ==> range(0, 10, 2)
'''

for item in range(10):
    print(item)
```

```
0
1
2
3
4
5
6
7
8
9
```

In [125]:

```
for item in range(5, 15):
    print(item)
```

```
5
6
7
8
9
10
11
12
13
14
```

In [134]:

```
for item in range(0, 10, 2):
    print(item)
```

```
0
2
4
6
8
```

In [135]:

```
# For loops in dictionary

'''
For loop list, dictionary, tuple, set, string
sobgulotei bebohar kora jabe
'''
user = {
    'name' : 'Hridoy',
    'Country' : 'BD',
    'Age' : 25
}

for i in user.items():
    print(i)
```

```
('name', 'Hridoy')
('Country', 'BD')
('Age', 25)
```

In [136]:

```
for j in user.keys():
    print(j)
```

```
name
Country
Age
```

In [137]:

```
for k in user.values():
    print(k)
```

```
Hridoy
BD
25
```

In [138]:

```
# for loop in List
```

In [185]:

```
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9]

add = 0

for item in my_list:
    add = add + item

print(add)
```

```
45
```

In [140]:

```
# Enumerate
'''
enumerate by default 0 theke count kora shuru korbe.
value assign korle value theke shuru korbe.
'''
for i, char in enumerate('Hridoy'):
    print(i, char)
```

```
0 H
1 r
2 i
3 d
4 o
5 y
```

In [141]:

```
for i, char in enumerate('Hridoy', 5):
    print(i, char)
```

```
5 H
6 r
7 i
8 d
9 o
10 y
```

In [ ]:

```

```

In [142]:

```
# While Loops
```

In [146]:

```
'''
while (condition):
    execute
'''
number = 10
while (number <= 15):
    print(number)
    number += 1
```

```
10
11
12
13
14
15
```

In [149]:

```
'''
while loop kaj shesh kore else a jabe
but while loop a break thakle else a jabe na.
'''

number = 10
while (number <= 15):
    print(number)
    number += 1
else:
    print('Done')
```

```
10
11
12
13
14
15
Done
```

In [150]:

```
number = 10
while (number <= 15):
    print(number)
    number += 1
    break
else:
    print('Done')
```

```
10
```

In [152]:

```
# Continuous take input from user using while loops

while True:
    number = int(input('Enter your number: '))

    if (number == 0):
        break
```

```
Enter your number: 5
Enter your number: 2
Enter your number: 1
Enter your number: 0
```

In [154]:

```
# Break, Continue, Pass

'''
Break use hoy loop loop thamanor jonno
Continue loop chalanor jonno
pass use kora hoy error handle korar jnno
suppose, ami chacchi j ekta for loop niye pore
kaj korbo tokhon pass kaje dey.
'''


for i in range(10):
```

```
  File "<ipython-input-154-8211ff77218d>", line 13

    ^
SyntaxError: unexpected EOF while parsing
```

In [155]:

```
for i in range(10):
    pass
```

In [ ]:

In [ ]:

```
=============================== Functions ================================
```

In [159]:

```
'''
we use functions to do the samethings over and over.

def  function_name(parameters):
    code

for calling a function:
function_name(parameters value or, arguments)
'''

def even_odd_check(number):
    if number % 2 == 0:
        print('Even Number')
    else:
        print("Odd Number")
```

In [160]:

```
even_odd_check(5)
```

```
Odd Number
```

In [161]:

```
even_odd_check(10)
```

```
Even Number
```

In [162]:

```
# Default parameter
```

In [163]:

```
'''
parameter dichi kintu call korar somoy
value dei nai se khetre error dekhabe,
eita fix korar jnno default value set kore dite hoy function a.
'''

even_odd_check()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call
 last)
<ipython-input-163-86b163e16292> in <module>
      4 '''
      5
----> 6 even_odd_check()

TypeError: even_odd_check() missing 1 required positional argument: 'n
umber'
```

In [164]:

```
def even_odd_check(number=5):
    if number % 2 == 0:
        print('Even Number')
    else:
        print("Odd Number")
```

In [165]:

```
even_odd_check()
```

```
Odd Number
```

In [179]:

```
'''
return keyword grab the value of the function.
'''

def add(number1, number2):
    return number1 + number2

add(5, 10)
```

Out[179]:

```
15
```

In [180]:

```python
# Docstring in function

def test(a):
    '''
    Info: This is a docstring
    '''
    print(a)
print(test.__doc__)
```

```
    Info: This is a docstring
```

In [181]:

```python
# Arguments and Keyword agruments
```

In [182]:

```python
'''
Parameter hisebe *args dile eker odik arguments ba parameter value deya
'''

def arg_check(*args):
    print(args)
arg_check(1, 2, 3, 4, 5)
```

```
(1, 2, 3, 4, 5)
```

In [189]:

```python
'''
parameter hisebe **kwargs dile parameter value hisebe
eker odhik keyword deya jay.

Rule: params, *args, default params, **kwargs
'''
def print_values(**kwargs):
    for key, value in kwargs.items():
        print("The value of {} is {}".format(key, value))

print_values(
            name_1="Hridoy",
            name_2="Asif",
            name_3="Shovan"
        )
```

```
The value of name_1 is Hridoy
The value of name_2 is Asif
The value of name_3 is Shovan
```

In [192]:

```
# Scope
'''
global scope and local scope ba variable.
'''

#global
a = 1

def scope_example():
    #Local Scope
    a = 5
    return a

print(scope_example())
print(a)
```

```
5
1
```

In [193]:

```
'''
function er moddhe global variable use korte hole
global keyword dite hobe ta chara error asbe.

1) Local
2) Parent local?
3) Global
'''

count = 0
def counter():
    count += 1
    return count

print(counter())
```

```
---------------------------------------------------------------------------
-----
UnboundLocalError                         Traceback (most recent call
 last)
<ipython-input-193-1dddb2c61bde> in <module>
      9     return count
     10
---> 11 print(counter())

<ipython-input-193-1dddb2c61bde> in counter()
      6 count = 0
      7 def counter():
----> 8     count += 1
      9     return count
     10

UnboundLocalError: local variable 'count' referenced before assignment
```

In [195]:

```
count = 10
def counter():
    global count
    count += 1
    return count

print(counter())
```

```
11
```

In [196]:

```
# Nonlocal Keyword
```

In [204]:

```
'''
nonlocal keyword function er vitore
local variable er value modify kore.
'''
def outer():
    x = "local"
    def inner():
        nonlocal x
        x = "Nonlocal"
        print("Inner : ", x)
    inner()
    print("Outer : ", x)
outer()
```

```
Inner :  Nonlocal
Outer :  Nonlocal
```

In [ ]:

```

```

In [ ]:

```
============================== Extra ==================================
```

In [205]:

```
number = float(input("Enter Your Number: "))

print("Your Number = ", number)
```

```
Enter Your Number: 10.2555651
Your Number =  10.2555651
```

In [208]:

```
'''
kintu ami chai doshomiker pore 2 ghor ba 3 ghor dekhak.
sekhetre amader % dite hobe r . ere por
bole dite hobe ami koy ghor dekhte chai.
r ""er por , thakbe na
'''
number = float(input("Enter Your Number: "))

print("Your Number = %.2f"%number)
```

```
Enter Your Number: 45.2544545
Your Number = 45.25
```

```
In [ ]: ============================ Basics Done ===============================
```