```
n = input("enter a number : ")
n = int(n)
total = 0
i = 1
while i <= n:
    total += i
    i += 1
print(total)
```

```
enter a number : 25
325
```

```
# LOOPING in list
fruits = ['orange','apple', 'banana', 'kiwi','pear']
#for Loop
for fruits in fruits:
  print(fruits)
```

```
orange
apple
banana
kiwi
pear
```

```
# List inside List
matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
print(matrix[2])
```

```
[7, 8, 9]
```

```
matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
for subList in matrix:
  for i in subList:
    print(i)
```

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

```
matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
print(matrix[2][0])
```

```
7
```

```
# generate lists with range functions

numbers = list(range(1,11))
print(numbers)

# pop method

numbers.pop()
print(numbers)

popped_item = numbers.pop()
print(numbers)
```

```
# index method

print(numbers.index(5))

# pass list to a function

def negative_list(l):
  negative = []
  for i in l:
    negative.append(-i)
  return negative

print(negative_list(numbers))
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
[1, 2, 3, 4, 5, 6, 7, 8, 9]
[1, 2, 3, 4, 5, 6, 7, 8]
4
[-1, -2, -3, -4, -5, -6, -7, -8]
```

```
# define a function which will take list containing numbers as input
# and return list containing square of every elements

# example
# numbers = [1,2,3,4]
# square_list(numbers) ----> return ----> [1,4,9,16]

# SOLUTION

def square_list(l):
  square = []
  for i in l:
    square.append(i**2)
    return square

numbers = list(range(1,11))
print(square_list(numbers))
```

```
[1]
```

```
# define a function which will take list as a argument and this function
# will return a reversed list

# examples
#[1,2,3,4] --- > [4,3,2,1]
# ['word1', 'word2'] ---> ['word2', 'word1']

# Note you simply do this with reverse method or [::-1]

# but  try to do this with the help of append and pop method

# SOLUTION

def reverse_list(l):
   l.reverse()
   return l

numbers = [1,2,3,4]
print(reverse_list(numbers))
```

```
[4, 3, 2, 1]
```

```
# define a function which will take list as a argument and this function
# will return a reversed list

# examples
#[1,2,3,4] --- > [4,3,2,1]
# ['word1', 'word2'] ---> ['word2', 'word1']
# Note you simply do this with reverse method or [::-1]

# but  try to do this with the help of append and pop method

# SOLUTION

def reverse_list(l):
  return l[::-1]

numbers = [1,2,3,4]
print(reverse_list(numbers))



def reverse_list(l):
  r_list = []
  for i in range(len(l)):
    popped_item = l.pop()
    r_list.append(popped_item)
  return r_list



numbers = [1,2,3,4]

print(reverse_list(numbers))
```

```
[4, 3, 2, 1]
[4, 3, 2, 1]
```

```
# define a function that take list of words as argument and
# return list with reverse of every element in that list

# example
# ['abc', 'tuv', 'xyz'] ---> ['cba', 'vut', 'zyx']

# SOLUTION

def reverse_element(l):
  elements = []
  for i in l:
    elements.append(i[::-1])
  return elements

words = ['abc', 'tuv', 'xyz']
print(reverse_element(words))
```

```
['cba', 'vut', 'zyx']
```

```
# filter odd even

# define a function
#input
#list ---. [1,2,3,4,5,6,7]

# output ----> [[1,3,5,7], [2,4,6,]]

# SOLUTION

def filter_odd_even(l):
  odd_nums = []
  even_nums = []
  for i in l:
    if i % 2 == 0:
      even_nums.append(i)

    else:
      odd_nums.append(i)
  output = [odd_nums, even_nums]
  return output
nums = [1,2,3,4,5,6,7]
```

```
print(filter_odd_even(nums))
```

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

```
# common elements finder function
# define a functions which take 2 lists as input and return a list
# which contains common elements of both lists

# example
# input ----> [1,2,5,8], [1,2,7,6]
# output ---> [1,2]

# SOLUTION

def common_finder(l1, l2):
  output = []
  for i in l1:
    if i in l2:
      output.append(i)
  return output

print(common_finder([1,2,5,8], [1,2,7,6]))
```

```
[1, 2]
```

```
# min and mix functions

numbers = [6,60,2]
print(min(numbers))
print(max(numbers))

def greatest_diff(l):
  return max(l) - min(l)

print(greatest_diff(numbers))
```

```
2
60
58
```

```
# function
# [1,2,3, [1,2], [3,4]] , input

# SOLUTION

def sublist_counter(l):
  count = 0
  for i in l:
    if type(i) == list:
      count += 1
  return count

mixed = [1,2,3, [1,2], [3,4]]
print(sublist_counter(mixed))
```

```
2
```

```
# Tuple data structure it is used data is not change
# Tuple can store any data type
# most important tuples are immutable, once tuple is created you can't update
# data inside tuple

example = ('one', 'two', 'three')
# no append , no insert , no pop , no remove
# days = ('monday', 'tuesday')
# tuples are faster than lists

# method are used in tuple
# conuts; index
# len function
# slicing
print(example[:2])
```

```
     ('one', 'two')
```

```
# more about toples

# looping in tuples
# tuple with one elements
# tuple without parenthesis
# tuple unpacking
# list inside tuple
# some functions that you can use with tuples

mixed = (1,2,3,4.0)

# for loop and tuple

for i in mixed:
  print(i)
# NOTE - you can use while loop too

# tuple with one elements
nums = (1,)
words = ('word1',)
print(type(nums))
print(type(words))

# tuple without parenthesis
guitars = 'yamaha', 'baton rouge', 'taylor'
print(type(guitars))

# TUPLE unpacking
guitarists = ('Maneli jamal', 'Eddie Van Der Meer', 'Andrew Foy')
guitarist1, guitarist2, guitarists3 = (guitarists)
print(guitarist1)

# list inside tuples
favorites = ('southern magnolia', ['Tokyo Ghoul theme', 'landscape'])
favorites[1].pop()
favorites[1].append("we made it")
print(favorites)

# min(), max ,sum
print(min(mixed))

print(max(mixed))

print(sum(mixed))
```

```
     1
     2
     3
     4.0
     <class 'tuple'>
     <class 'tuple'>
     <class 'tuple'>
     Maneli jamal
     ('southern magnolia', ['Tokyo Ghoul theme', 'we made it'])
     1
     4.0
     10.0
```

```
# Function returning two values
def func(int1, int2):
  add = int1 + int2
  multiply = int1*int2
  return add, multiply

print(func(2,3))
add, multiply = func(2,3)
print(add)
print(multiply)
```

```
(5, 6)
5
6
```

```
# something more about tuple , list , str

# num = tuple(range(1,11))

nums = list((1, 2, 3, 4, 5, 6, 7, 8, 9, 10))
print(nums)

nums= str((1, 2, 3, 4, 5, 6, 7, 8, 9, 10))
print(nums)

num_list = str((1, 2, 3, 4, 5, 6, 7, 8, 9, 10))
print(num_list)
print(type(num_list))
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
<class 'str'>
```

```
# dictionaries intro
# Q - why we use dictionaries?
# A -Because of limitations of lists , lists are not enough to represent
# real data

# Example
user = ['Akshay', 18, ['coco', 'kimi no na wa'], ['awakening', 'fairy tale']]
# this list contains user name , age , favi movies , fav tunes
# you can do this but this is not a good way to do this.

# Q - what are dictionaries
# A - unordered collections of data in key : values pair.

# how to create dictionaries
user = {'name' : 'Akshay', 'age' : 18}
print(user)
print(type(user))

# second method to create dictionary
user1 = dict(name =  'Akshay', age = 18)
print(user1)


# how  to access data from dictionary
# NOTE - There is no indexing because of unordered collections of data.
print(user['name'])
print(user['age'])

# Which type of data a dictionary can store ?
# anythings
# numbers, strings, list , dictionary


user_info = {
    'name' : 'Akshay',
    'age' : 18,
    'fav_movies' : ['coco', 'kimi no na wa'],
    'fav_tunes' : ['awakening', 'fairy tale'],
}
print(user_info['fav_movies'])

# How to add data to empty dictionary
user_info2 = {}
user_info2['name'] = 'Mohit'

print(user_info2)
```

```
{'name': 'Akshay', 'age': 18}
<class 'dict'>
{'name': 'Akshay', 'age': 18}
Akshay
18
['coco', 'kimi no na wa']
{'name': 'Mohit'}
```

```
 # in keyword and iterations in dictionary

 user_info = {
    'name' : 'Akshay',
    'age' : 18,
    'fav_movies' : ['coco', 'kimi no na wa'],
    'fav_tunes' : ['awakening', 'fairy tale'],
}

# check if key exist in dictionary
 if 'name' in user_info:
   print('present')
 else:
    print('not present')

# check if value exist in dictionary
 if 'Akshay' in user_info.values():
   print('present')
 else:
    print('not present')

# loops in dictionaries
 for i in user_info:
    print(i)

 for i in user_info.values():
    print(i)

# values method
user_info_values = user_info.values()
print(user_info_values)
print(type(user_info_values))

# items method
user_items = user_info.items()
print(user_items)
print(type(user_items))
```

```
present
present
name
age
fav_movies
fav_tunes
Akshay
18
['coco', 'kimi no na wa']
['awakening', 'fairy tale']
dict_values(['Akshay', 18, ['coco', 'kimi no na wa'], ['awakening', 'fairy tale']])
<class 'dict_values'>
dict_items([('name', 'Akshay'), ('age', 18), ('fav_movies', ['coco', 'kimi no na wa']), ('fav_tunes', ['awakening', 'fairy tale'])])
<class 'dict_items'>
```

```
# add and delete data
user_info = {
    'name' : 'Akshay',
    'age' : 18,
    'fav_movies' : ['coco', 'kimi no na wa'],
    'fav_tunes' : ['awakening', 'fairy tale'],
}

# how to add data
user_info['fav_songs'] = ['song1', 'song2']
print(user_info)
```

```
# pop method
popped_item = user_info.pop('fav_tunes')
print(f"popped item is {popped_item}")
print(user_info)

# popitem method
popped_item = user_info.popitem()
print(user_info)
print(type(popped_item))
```

```
{'name': 'Akshay', 'age': 18, 'fav_movies': ['coco', 'kimi no na wa'], 'fav_tunes': ['awakening', 'fairy tale'], 'fav_songs': ['song1',
popped item is ['awakening', 'fairy tale']
{'name': 'Akshay', 'age': 18, 'fav_movies': ['coco', 'kimi no na wa'], 'fav_songs': ['song1', 'song2']}
{'name': 'Akshay', 'age': 18, 'fav_movies': ['coco', 'kimi no na wa']}
<class 'tuple'>
```

```
# Update Dictionary
user_info = {
    'name' : 'Akshay',
    'age' : 18,
    'fav_movies' : ['coco', 'kimi no na wa'],
    'fav_tunes' : ['awakening', 'fairy tale'],
}

more_info = {'state' : 'Haryana', 'hobbies' : ['coding','reading', 'guitar']}
user_info.update(more_info)
print(user_info)
```

```
{'name': 'Akshay', 'age': 18, 'fav_movies': ['coco', 'kimi no na wa'], 'fav_tunes': ['awakening', 'fairy tale'], 'state': 'Haryana', 'hc
```

✓ 0s completed at 9:56 PM

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.