

**TRIBHUVAN UNIVERSITY**

Institute Of Engineering

**A PROJECT REPORT ON**

Python Second Lab

Submitted by: Name: Bibek Yadav

Roll no: 081BEL022 Section: AB

Submitted to: Prakash Chandra

1. Write a program to input n numbers and store them in a list. Then perform the following operations: i) Using built in functions ii) without using built-in functions:
2. Find the maximum and minimum number
3. Sort the list in ascending order
4. Remove duplicate elements



nums = []

n = int(input("Enter how many numbers: "))

for i in range(n):

num = int(input(f"Enter number {i + 1}: "))

nums.append(num)

print("\nOriginal List:", nums)

print("\n--- Using Built-in Functions ---")

print("Maximum:", max(nums))

print("Minimum:", min(nums))

sorted\_builtin = sorted(nums)

print("Sorted List:", sorted\_builtin)

unique\_builtin = list(set(nums))

unique\_builtin.sort()

print("List without duplicates:", unique\_builtin)

print("\n--- Without Using Built-in Functions ---")

max\_num = nums[0]

min\_num = nums[0]

for num in nums:

if num > max\_num:

max\_num = num

if num < min\_num:

min\_num = num

print("Maximum:", max\_num)

print("Minimum:", min\_num)

sorted\_custom = nums.copy()

for i in range(len(sorted\_custom)):

for j in range(0, len(sorted\_custom) - i - 1):

if sorted\_custom[j] > sorted\_custom[j + 1]:

sorted\_custom[j], sorted\_custom[j + 1] = sorted\_custom[j + 1], sorted\_custom[j]

print("Sorted List:", sorted\_custom)

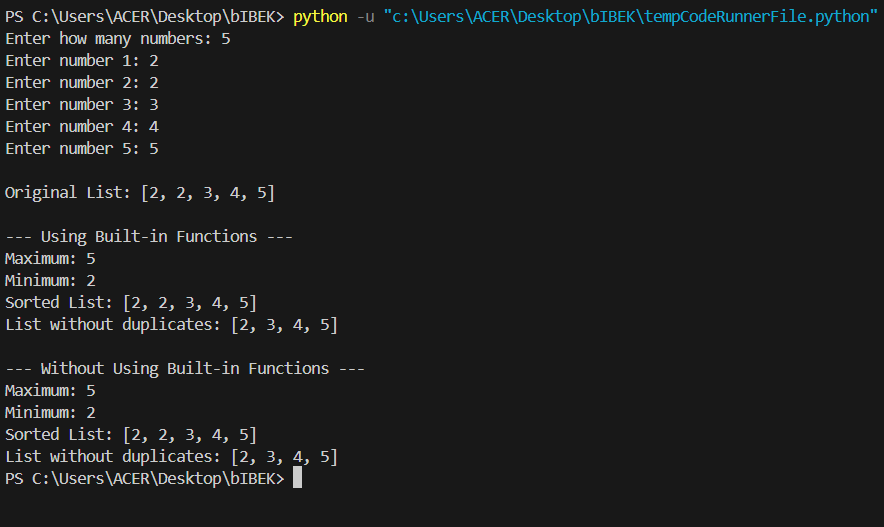
unique\_custom = []

for num in sorted\_custom:

if num not in unique\_custom:

unique\_custom.append(num)

print("List without duplicates:", unique\_custom)

Output: 

1. Given two lists of integers, write a program to merge them into a single list and then remove the elements that are common in both.

a=[]

b=[]

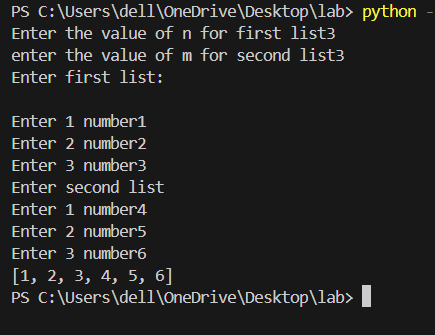
n=int(input("Enter the value of n for first list")) m=int(input("enter the value of m for second list")) print("Enter first list:\n")

for i in range(1,n+1): a.append(int(input(f"Enter {i} number")))

print("Enter second list") for i in range(1,m+1):

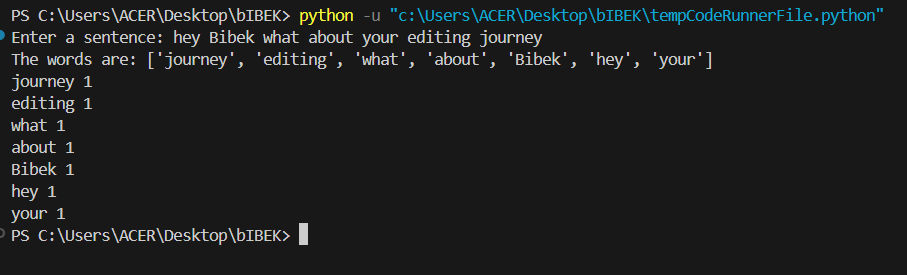
b.append(int(input(f"Enter {i} number"))) merged=a + b

print(list(set(merged)))

Output:

1. Create a program that reads a sentence from the user and stores each word as an element of a list. Then count the frequency of each word using only lists



Output: 

1. Write a program to simulate a basic stack and queue using a list. Provide options to push, pop (stack), enqueue, and dequeue (queue).

l=[]

while True:

choice=input("Enter 1 for push and 2 for pop: ") if choice=="1":

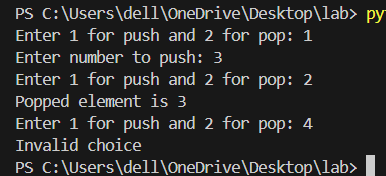
num=int(input("Enter number to push: ")) l.append(num)

elif choice=="2": if len(l)==0: continue

print(f"Popped element is {l.pop()}") else:

print("Invalid choice") break

Output:



1. Write a Python function that accepts a list and returns a new list containing only the elements at even indexes and those that are prime numbers.

a = [1, 3, 2, 4, 5, 6, 7, 8, 9, 10]

new\_list = []

for i in range(len(a)): if i % 2 == 0:

new\_list.append(a[i]) for num in a:

if num < 2: continue

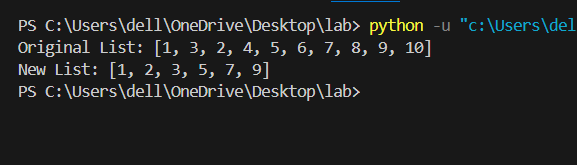
prime = True

for j in range(2, int(num \*\* 0.5) + 1): if num % j == 0:

prime = False break

if prime and num not in new\_list: new\_list.append(num)

new\_list.sort() print("Original List:", a) print("New List:", new\_list)

Output:

1. Write a program to create a tuple of n numbers, then find:
2. The average of the numbers
3. The median
4. The mode (without using libraries)

a=[]

a\_t=() sum=0 d={}

f=0 max\_count=0 mode=[]

n=int(input("Enter the value of n: ")) for i in range(1,n+1):

a.append(int(input(f"Enter {i} number: "))) a\_t=tuple(a)

for num in a\_t: sum+=num

avg=sum/n print(avg)

for i in range(len(a)):

for j in range(0, len(a) - i - 1): if a[j] > a[j + 1]:

temp = a[j] a[j] = a[j + 1] a[j + 1] = temp

if(n%2==1):

median=a[n//2] else:

mid1=a[n//2 -1]

mid2=a[n//2] median=(mid1+mid2)/2

print(median) for i in a\_t:

if i in d:

d[i]+=1

else:

d[i]=1 for key in d:

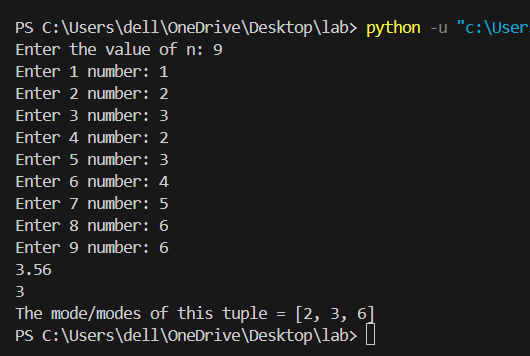
if(d[key]>max\_count): max\_count=d[key]

for key in d: if(d[key]==max\_count):

mode.append(key)

print(f"The mode/modes of this tuple = {mode}")

Output:



1. Write a program that receives a list of tuples representing (x, y) coordinates. Determine whether the points form a straight line.

b = []

t = ()

n = int(input("Enter the value of n: ")) for i in range(0, n):

a = []

a.append(int(input(f"x{i}: ")))

a.append(int(input(f"y{i}: "))) t = tuple(a)

b.append(t) print("Points entered:", b) if n < 2:

print("All points lie on a straight line.") else:

x0, y0 = b[0] x1, y1 = b[1]

if x1 - x0 == 0: slope = None

else:

slope = (y1 - y0) / (x1 - x0) flag = True

for i in range(2, n): x, y = b[i]

if slope is None: if x != x0:

flag = False break

else:

if (y - y0) != slope \* (x - x0): flag = False

break

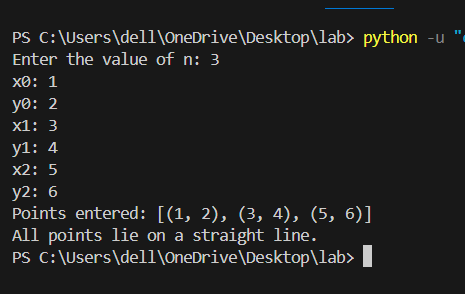
if flag:

print("All points lie on a straight line.")

else:

print("Points do NOT lie on a straight line.")

Output:



1. Write a program to input two sets of student roll numbers: one who play cricket and another who play football. Find:
   1. Students who play both sports
   2. Students who play only one sport
   3. Students who play neither (given a master list of all students)

a=[]

c=[]

f=[]

for i in range(1,49): a.append(i)

u=set(a)

print("Enter the roll numbers who play cricket: ")

i=0

while True:

x=int(input(f"Enter {i+1} roll number: ")) c.append(x)

i+=1

ch=input("Any one else?(y/n)") ch=ch.lower()

if(ch=='y'):

True else:

break

print("Enter the roll numbers who play football: ") i=0

while True:

x=int(input(f"Enter {i+1} roll number: ")) f.append(x)

i+=1

ch=input("Any one else?(y/n)") ch=ch.lower()

if(ch=='y'):

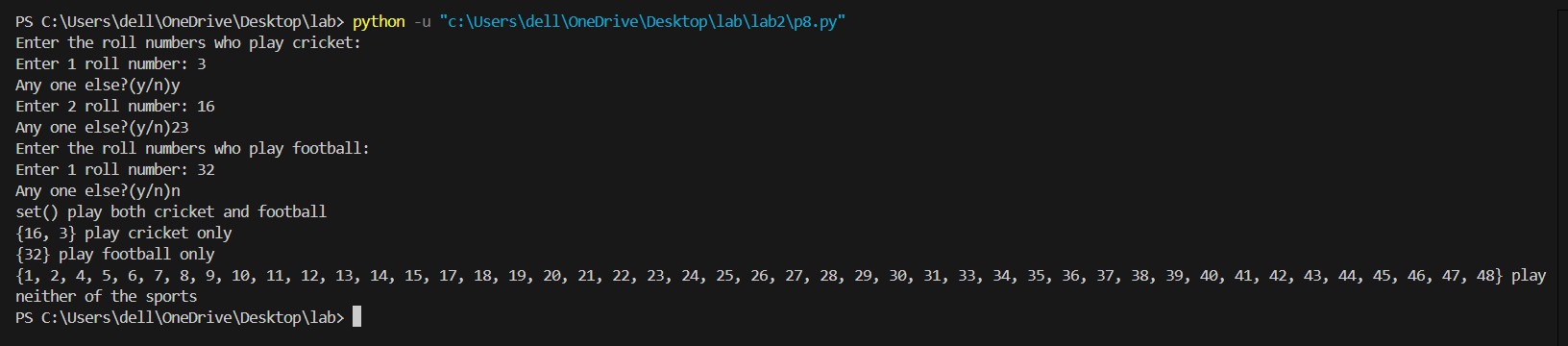
True else:

break c\_set=set(c) f\_set=set(f) int\_set=c\_set & f\_set union\_set=c\_set | f\_set

set1=c\_set.difference(int\_set) set2=f\_set.difference(int\_set) neither=u.difference(union\_set) print(f"{int\_set} play both cricket and football") print(f"{set1} play cricket only")

print(f"{set2} play football only") print(f"{neither} play neither of the sports")

Output:



1. Create a set of random numbers. Add more numbers until the set has 10 unique elements. Also, remove the smallest and largest element.

import random

c = set(random.sample(range(1, 100), 5)) print(f"The original set of random numbers is: {c}")

c\_list = list(c)

while len(c\_list) < 10:

x = int(input(f"Enter number #{len(c\_list)+1}: ")) if x not in c\_list:

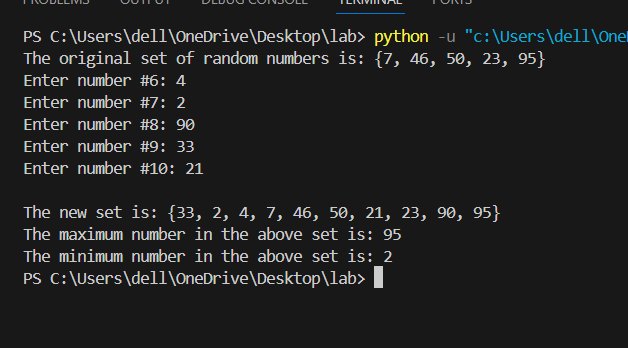
c\_list.append(x) else:

print("Duplicate! Enter a unique number.")

final\_set = set(c\_list)

print(f"\nThe new set is: {final\_set}")

print(f"The maximum number in the above set is: {max(final\_set)}") print(f"The minimum number in the above set is: {min(final\_set)}")

Output:

1. Write a Python function that accepts a sentence and returns a set of all unique vowels used.

 sentence = input("Enter a sentence: ")

vowels = {'a', 'e', 'i', 'o', 'u'}

used\_vowels = set()

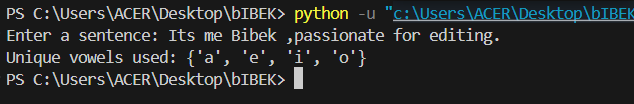
for char in sentence.lower():

if char in vowels:

used\_vowels.add(char)

print("Unique vowels used:", used\_vowels)

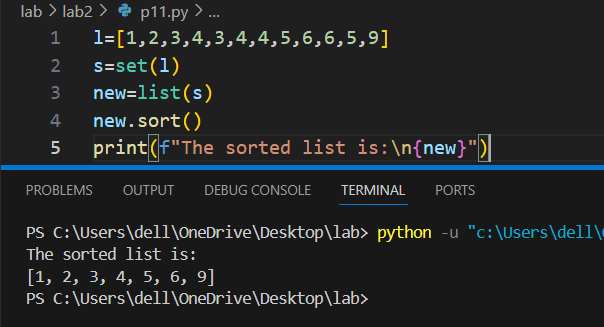
Output:



1. Given a list of numbers with duplicates, use a set to remove the duplicates. Then, convert it back to a sorted list and display the result.

l=[1,2,3,4,3,4,4,5,6,6,5,9]

s=set(l) new=list(s) new.sort()

print(f"The sorted list is:\n{new}") Output:

1. Create a dictionary to store student names as keys and their scores in three subjects as values (in a list). Write functions to:
   1. Display the average marks of each student
   2. Find the topper
   3. Update the marks of a student

 def input\_student\_data():

d = {}

n = int(input("How many students? "))

for i in range(n):

name = input(f"\nEnter the name of student {i+1}: ")

marks = []

print(f"Enter the marks in three subjects of {name}:")

for j in range(1, 4):

mark = int(input(f"Mark {j}: "))

marks.append(mark)

d[name] = marks

return d

def calculate\_averages(d):

new = {}

for key in d:

avg = round(sum(d[key]) / 3, 2)

new[key] = avg

return new

def display\_averages(new):

print("\nAverage marks of each student:")

for student, avg in new.items():

print(f"{student}: {avg}")

def find\_topper(new):

topper = max(new, key=lambda k: new[k])

print(f"\nStudent with highest average is: {topper}")

print(f"Their average is: {new[topper]:.2f}")

def update\_marks(d, new):

update = input("\nDo you want to update marks of any student? (y/n): ").lower()

while update == 'y':

student = input("Enter the name of the student whose marks you want to update: ")

if student in d:

print(f"Current marks for {student}: {d[student]}")

new\_marks = []

print("Enter the new marks for three subjects:")

for i in range(1, 4):

mark = int(input(f"Mark {i}: "))

new\_marks.append(mark)

d[student] = new\_marks

avg = round(sum(new\_marks) / 3, 2)

new[student] = avg

print(f"Marks and average updated for {student}!")

else:

print("Student not found.")

update = input("\nDo you want to update marks of another student? (y/n): ").lower()

def main():

d = input\_student\_data()

new = calculate\_averages(d)

display\_averages(new)

update\_marks(d, new)

print("\nUpdated marks dictionary:")

print(d)

print("\nUpdated averages dictionary:")

print(new)

sorted\_new = dict(sorted(new.items(), key=lambda item: item[1]))

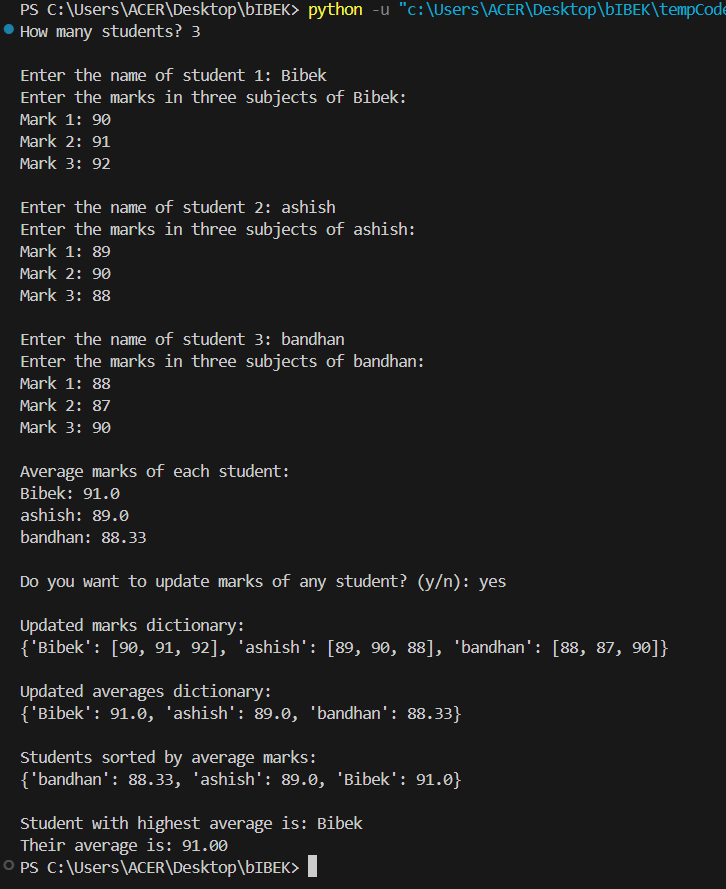
print("\nStudents sorted by average marks:")

print(sorted\_new)

find\_topper(new)

main()

Output:



1. Write a program that reads a text and counts the frequency of each character (excluding spaces and special characters) using a dictionary.

text = input("Enter a text: ") freq = {}

for char in text.lower(): if char.isalpha():

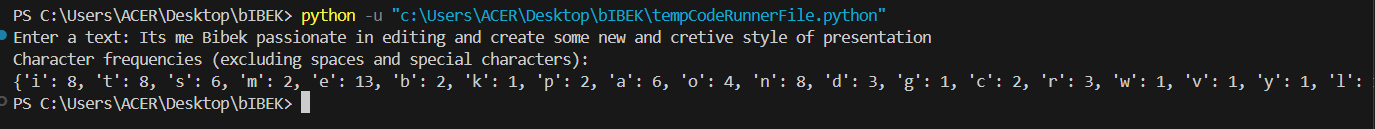
if char in freq: freq[char] += 1

else:

freq[char] = 1

print("Character frequencies (excluding spaces and special characters):") print(freq)

Output:



1. Build a dictionary where the keys are product names and the values are their prices. Implement options to:
   1. Add a new product
   2. Update price of an existing product
   3. Find products within a given price range

d={}

for i in range(5):

key=input(f"Enter the {i+1} product name: ") d[key]=int(input(f"Enter the price of {key}: "))

print(d) while True:

print("a -> Adding new products\nb -> Update the price of product\nc -> Find the product in given price range")

choice=input("Enter your choice: ") choice=choice.lower() if(choice=='a'):

key=input("Enter the new product: ") if(key not in d):

d[key]=int(input(f"Enter the price of {key}: ")) else:

print("Product already exixted.") elif(choice=='b'):

key=input("Enter the product to be updated: ") if(key in d):

d[key]=int(input(f"Enter the updated price of {key}:")) else:

print("Product doesnot exixt.") elif(choice=='c'):

r1=int(input("Enter the initial value: ")) r2=int(input("Enter the final value: ")) rangee={}

for key in d:

if(d[key] in range(r1,r2+1)): rangee[key]=d[key]

print(f"The products for your price range is/are:\n{rangee}") else:

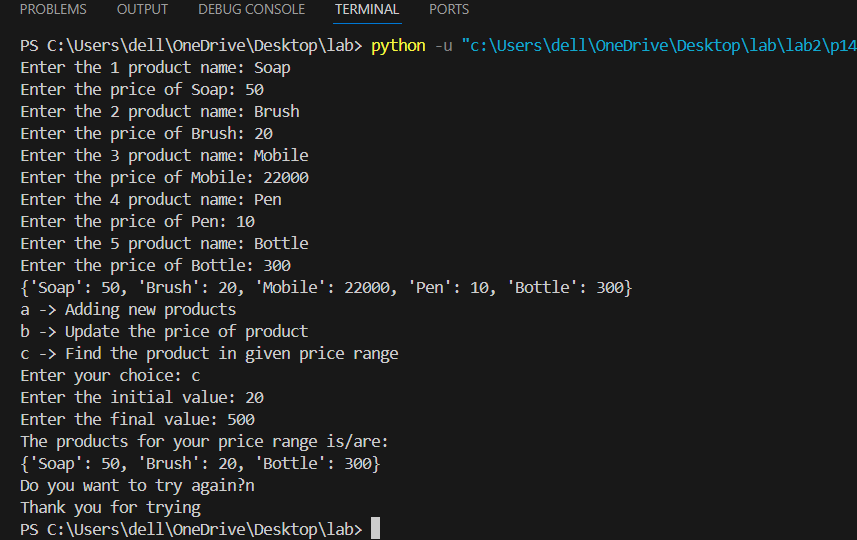
print("Invalid choice!!!") ch=input("Do you want to try again?")

if(ch.lower()=='y'): continue

else:

break

print("Thank you for trying")

Output:

My github link for this report is