**# Written by Varad Kinjal Dixit, 23BBT035**

**# Assignment 1**

**# Write a program that asks input of two numbers from user, assigns them to two different variables, swaps the variables, and then prints them.**

**def main() -> None:**

**num1 = int(input("First number: "))**

**num2 = int(input("Second number: "))**

**print("Before swap: a =", num1, "b =", num2)**

**num1, num2 = num2, num1**

**print("After swap: a =", num1, "b =", num2)**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 2**

**# Write a program that checks the maximum of three numbers, input from user.**

**def minimax(p):**

**maximum = minimum = p[0]**

**for x in range(len(p)):**

**if p[x] > maximum:**

**maximum = p[x]**

**if p[x] < minimum:**

**minimum = p[x]**

**return maximum, minimum**

**def main():**

**p = (int(input("Enter three numbers: ")), int(input()), int(input()))**

**r = minimax(p)**

**print('maximum: ', r[0], '\tminimum: ', r[1])**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 3**

**# Write a program to output the times table of a number.**

**def make\_table(number,upto):**

**for x in range(upto):**

**print(number,"\*",x+1,"=\t",number\*(x+1))**

**def main():**

**a = int(input("Number to multiply: "))**

**b = int(input("Times tables up to: "))**

**make\_table(a,b)**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 4, part 1**

**# Write a program to count the number of vowels in a given string**

**ref = "AEIOUaeiou"**

**def counter(string):**

**count = 0**

**for char in string:**

**if char in ref:**

**count += 1**

**return count**

**def main():**

**a = input("Enter your string: ")**

**print(counter(a))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 4, part 2**

**# Write a program to remove vowels from a string**

**reference = "AaEeIiOoUu"**

**def remove\_vowels(string):**

**for x in range(10):**

**string = string.replace(reference[x],'')**

**return string**

**def main():**

**string = input("Enter your string: ")**

**print(remove\_vowels(string))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 5**

**# Write a program to compute the cube root of a number**

**def cube\_root(number):**

**a = 0**

**while a \*\* 3 < number:**

**a += 1**

**if a \*\* 3 == number:**

**ret = f"the cube root of {number} is {a}"**

**else:**

**ret = f"the cube root of {number} is between {a-1} and {a}"**

**return ret**

**def main():**

**a = int(input("Enter number: "))**

**print(cube\_root(a))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 6**

**# Write a program that checks if a number is an armstrong number or not.**

**# An Armstrong number is one whose value is equal to the sum of its digits, each raised to the mth power, where m is the number of digits in the number.**

**def armstrong(number):**

**result = 0**

**# with a string, we can iterate over each digit individually**

**numberstring = str(number)**

**for x in numberstring:**

**result += int(x) \*\* len(numberstring)**

**# is result the same as our initial number?**

**final\_result = (result == number) and (number != 0) # 0 is not considered an armstrong number (by the OEIS), even though 0^1 = 0.**

**return final\_result**

**# optional: write an algorithm that checks numbers up to a desired value**

**def interesting(ran):**

**nums = []**

**i = 0**

**for x in range(ran):**

**if armstrong(x):**

**i += 1**

**nums.append(x)**

**for x in range(len(nums)):**

**if x != len(nums) - 1:**

**print(nums[x], end=', ')**

**else:**

**print(nums[x])**

**return**

**def main():**

**a = int(input("Enter your number: "))**

**print(armstrong(a))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 7**

**# Write a program to check if a number is prime or not**

**def isprime(num):**

**prime = True**

**if (num % 2 == 0 and num != 2) or num == 1:**

**prime = False**

**for x in range(3, 1 + int(num \*\* (1/2)), 2):**

**if num % x == 0:**

**prime = False**

**return prime**

**def primesto(num):**

**result = []**

**for x in range(num+1):**

**if isprime(x):**

**result.append(x)**

**return result**

**def primesto\_(x):**

**return [k for k in range(x+1) if isprime(k)]**

**def main():**

**num = int(input("Enter a number to check primality: "))**

**if isprime(num):**

**print("Number is prime.")**

**else:**

**print("Number is not prime")**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 8**

**# Write a program to implement n C r and n P r functions.**

**# Choose/Combination operator n C r tells you how many subsets of a set with cardinality n have cardinality r i.e. how many "combinations" of cardinality r you can make out of it. the formula for it is (n!/((r!)((n-r)!)))**

**# n P r refers to permutations. For instance, {1,3,2} == {1,2,3}, but (1,3,2) != (1,2,3). There are thus r! more permutations of something than combinations (as there are r! unique ways to arrange r elements)**

**def factorial(x):**

**result = 1**

**for i in range(x):**

**result \*= (i+1)**

**return result**

**def c(n,r):**

**# by formula**

**result = factorial(n) // (factorial(r) \* factorial(n-r))**

**return result**

**def p(n,r):**

**# also by formula**

**result = factorial(n) // factorial(n-r)**

**return result**

**def main():**

**num1, num2 = int(input("Enter first number: ")), int(input("Enter second number: "))**

**print("n C r = ", c(num1, num2), "n P r = ", p(num1, num2))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 9, part 1**

**# Write a program that checks if a number is a palindrome or not**

**# A palindrome is a number whose value is identical whether its digits are read from left to right or right to left (omitting trailing zeroes)**

**def ispal(number):**

**temp = number**

**result = 0**

**while temp > 0:**

**#print(temp, number, result)**

**result \*= 10**

**result += temp % 10**

**temp //= 10**

**#print(temp, number, result)**

**return result == number**

**def main():**

**n = int(input("Enter a number: "))**

**print(ispal(n))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 9, part 2**

**# Write a program that checks if a given text input is a palindrome or not.**

**# A palindrome (like racecar) is spelled identically forwards as well as backwards.**

**#string = ["li", "ma", "di", "ga", "ma", "ma", "ga", "di", "ma", "li"]**

**def main():**

**string = input("enter your string: ")**

**if string == string[::-1]:**

**print(string, "is a palindrome")**

**else:**

**print(string, "is not a palindrome; its reverse is", string[::-1])**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 10**

**# Write a program to understand the scope of a function**

**def g(x):**

**def h():**

**#print('x in h =', x) #-> NameError**

**x = 'abc'**

**#print('x after h =', x) #-> 'abc'**

**#print('before g, x =', x) #-> input value**

**x = x + 1**

**print('in g(x): x =', x) #-> input value + 1**

**h() # doesn't do anything because the x in h is treated differently from the x in g**

**# this is called scope**

**return x**

**def main():**

**x = int(input('what is x '))**

**z = g(x)**

**print(z)**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 11**

**# copy the Towers of Hanoi algorithm given below**

**def print\_move(fr, to):**

**fr, to = str(fr), str(to)**

**print('move from ' + fr + ' to ' + to)**

**return**

**def towers(n, fr, to, spare):**

**if n == 1:**

**print\_move(fr,to)**

**else:**

**towers(n-1,fr,spare,to)**

**towers(1,fr,to,spare)**

**towers(n-1,spare,to,fr)**

**return # note that the value returned here is None**

**# Copy the improved towers of Hanoi algorithm**

**def Towers(n,fr,to,spare):**

**if n==1:**

**print\_move(fr,to)**

**return**

**Towers(n-1,fr,spare,to)**

**print\_move(fr,to)**

**Towers(n-1,spare,to,fr)**

**def main():**

**n = int(input("Enter your number here: "))**

**print("Assume the pole the stack currently is called 'start', and the other two are called 'end' and 'spare'. To move the stack from 'start' to 'end', do the following:")**

**Towers(n, "'start'", "'end'", "'spare'")**

**print("The stack is moved from 'start' to 'end.'")**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 12**

**# Write a program to compute whether a given point is on a given circle or not**

**def oncircle(x,y,r):**

**dsqr = x \*\* 2 + y \*\* 2**

**if dsqr > r \*\* 2:**

**l = "outside"**

**elif dsqr < r \*\* 2:**

**l = "inside"**

**else:**

**l = "on"**

**print("The point lies", l, "the circle")**

**from math import sqrt**

**def newoncircle(x,y,r):**

**dsqr = x \*\* 2 + y \*\* 2**

**d = sqrt(dsqr)**

**if d > r:**

**l = "outside"**

**elif d < r:**

**l = "inside"**

**else:**

**l = "on"**

**print("The point lies", l, "the circle")**

**# optional: in arbitrary dimensions**

**def On\_Circle(p: list[int], c: list[int], r: int):**

**#p = [point\_x, point\_y, ...]**

**#c = [center\_x, center\_y, ...]**

**#r = radius**

**p\_1 = p.copy()**

**dimensions = len(p\_1)**

**sumofsqrs = 0**

**# compute each delta(x) and square it**

**for d in range(dimensions):**

**p\_1[d] -= c[d]**

**sumofsqrs += p\_1[d] \*\* 2**

**distance = sqrt(sumofsqrs)**

**if distance > r:**

**l = 'outside'**

**elif distance < r:**

**l = 'inside'**

**else:**

**l = 'on'**

**print("The point lies", l, "the circle")**

**return**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\nUse the function On\_Circle() for an arbitrary-dimensional circle and newoncircle() for a 2 dimensional one.") # Assignment 13**

**# Write a program to import pi from math, and then use it to define area and circumference calculating functions**

**from math import pi**

**def area(r):**

**a = pi \* r \*\* 2**

**return a**

**def circum(r):**

**c = pi \* r \* 2**

**return c**

**def main():**

**r = int(input("Enter the radius of the circle: "))**

**print("The area of the circle is:", area(r), "\nThe circumference is:", circum(r))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 14**

**# collinear = lying on the plane such that there exists a line that passes through all points**

**# eqn for a line is y = mx+c**

**# or more relevantly, that dy/dx is constant over the graph**

**# define a function that tells us the slope between two points**

**def riorun(x\_1, y\_1, x\_2, y\_2): # "RIse Over RUN"**

**numerator = y\_2 - y\_1**

**denominator = x\_2 - x\_1**

**numerator, denominator = abs(numerator), abs(denominator)**

**try:**

**result = numerator / denominator**

**except ZeroDivisionError:**

**result = 'vert' # vertical, since y changes but x remains constant**

**return result**

**# define a function that tells us if three points are colinear**

**def iscollinear(x\_1,y\_1,x\_2,y\_2,x\_3,y\_3):**

**x1tox2 = riorun(x\_1,y\_1,x\_2,y\_2)**

**x2tox3 = riorun(x\_2,y\_2,x\_3,y\_3)**

**collin\_flag = False**

**if x1tox2 == x2tox3:**

**collin\_flag = True**

**return collin\_flag**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\nUse the function iscollinear() to find if 3 points are collinear.") # Assignment 15**

**# Write a program to check the area of a triangle given the cartesian coordinates of its vertices**

**def triangle\_area(x\_1,y\_1,x\_2,y\_2,x\_3,y\_3):**

**# imagine copying the triangle you have and rotating it by 180 degrees**

**# now put like sides of the triangles over top of each other and deleting that side**

**# this resultant shape is a parallelogram**

**# the formula for the area of a paralellogram given three of its points is:**

**area = (x\_1-x\_3)\*(y\_2-y\_1) - (x\_1-x\_2)\*(y\_3-y\_1)**

**# but this value may be negative for certain coordinates, so we take modulus:**

**area = abs(area)**

**# and the triangle is, by definition, half of the parallelogram**

**area /= 2**

**return area**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\nUse the function triangle\_area to find the area of a given triangle.") # Assignment 16**

**# Write a program to print a triangle that has alternating lettter and number lines and the letter lines reset and the number ones don't**

**# eg**

**# 1**

**# A B**

**# 2 3 4**

**# A B C D**

**# 5 6 7 8 9**

**#def numbline(x):**

**# for y in range((x-1)\*\*2, x \*\* 2):**

**# print(y+1,end=' ')**

**# print('')**

**def letline(x):**

**letters = x \* 2**

**if letters >= 26:**

**print("ABCDEFGHIJKLMNOPQRSTUVWXYZ " \* (letters // 26), end='')**

**print("ABCDEFGHIJKLMNOPQRSTUVWXYZ"[:letters%26])**

**def my\_tri(lines):**

**#string = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"**

**i = 1**

**for x in range(None, lines, 2):**

**#numbline(i)**

**print(list(range((i-1)\*\*2, i\*\*2)))**

**letline(i)**

**i += 1**

**return**

**def main():**

**b = int(input("How many lines "))**

**my\_tri(b)**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 17**

**# Write a program to read two text files and merge their contents**

**def initialize\_files():**

**with open('17file1.txt', 'w') as file:**

**file.write("Hello, ")**

**with open('17file2.txt', 'w') as file:**

**file.write("World!")**

**return True**

**def merge\_files():**

**with open('17file1.txt', 'r') as file1:**

**with open('17file2.txt', 'r') as file2:**

**result = file1.read() + file2.read()**

**with open('17cat.txt', 'w') as file:**

**file.write(result)**

**return True**

**def main():**

**initialize\_files()**

**merge\_files()**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 18**

**# Write a program to create a CSV file and write a table of USER-IDS and PASSWORDS to it.**

**# The program should then read a password from the user and print the corresponding USER-ID if applicable, else throw an appropriate error message.**

**def initialize\_csv():**

**# csv is a surprisingly simple format so we don't need to import csv because we're not lazy and write code ourselves.**

**with open('18csv.csv', 'w', newline='') as file:**

**file.write("User-ID,Password\n")**

**file.write("user1,pass1\n")**

**file.write("user2,pass2\n")**

**file.write("user3,pass3\n")**

**return True**

**def read\_csv(p):**

**# p is the password**

**with open('18csv.csv', 'r') as file:**

**# iterating over a file goes line by line**

**for line in file:**

**if p + '\n' in line:**

**result = line.split(',')[0]**

**break**

**else:**

**result = "No user with that password"**

**return result**

**def main():**

**initialize\_csv()**

**password = input('Enter the password: ')**

**print(read\_csv(password))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 19**

**# Write a program that counts the number of duplicate elements in a list.**

**# adds one to duplicate count for every occurence of an element that is not the first**

**def duplicount(l):**

**duplicates = 0**

**for i in range(len(l)):**

**if l[i] in l[:i]:**

**duplicates += 1**

**print ("there are", duplicates, "duplicate items")**

**# adds one to duplicate count only if it is the second occurrence of that element**

**def Duplicount(l):**

**duplicates = []**

**for i in range(len(l)):**

**if l[i] in l[:i] and l[i] not in duplicates:**

**duplicates.append(l[i])**

**print ("there are", len(duplicates), "duplicate items (", duplicates, ")")**

**# simply counts every non-unique element**

**def dupli\_count(l):**

**duplicates = 0**

**for i in range(len(l)):**

**l2 = l[:i] + l[i+1:]**

**if l[i] in l2:**

**duplicates += 1**

**print ("there are", duplicates, "duplicate items")**

**def main():**

**mylist = input("Enter a list of space separated integers: ").split()**

**mylist = [int(i) for i in mylist]**

**print(duplicount(mylist), Duplicount(mylist), dupli\_count(mylist))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\nUses three functions which each have subtly different functionality, hence the three outputs.")**

**main() # Assignment 20**

**# Write a program to sum over the elements in a list**

**def Sigma(l):**

**mysum = 0**

**for x in l:**

**mysum += x**

**return mysum**

**def main():**

**mylist = input("Enter a list of space separated integers: ").split()**

**mylist = [int(i) for i in mylist]**

**print(Sigma(mylist))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 21**

**# Write a program to replace vowels in a string with special symbols of your choice**

**# (I chose 4, 3, 1, 0, 9 for a, e, i, o, u respectively)**

**def vowelreplace(inst):**

**myst = ""**

**for char in inst:**

**if char == 'a' or char == 'A':**

**myst += '4'**

**elif char == 'e' or char == 'E':**

**myst += '3'**

**elif char == 'i' or char == 'I':**

**myst += '1'**

**elif char == 'o' or char == 'O':**

**myst += '0'**

**elif char == 'u' or char == 'U':**

**myst += '9'**

**else:**

**myst += char**

**print(myst)**

**def vowelreplace\_(inst):**

**ref1 = "AEIOUaeiou"**

**ref2 = "4310943109"**

**myst = ""**

**for char in inst:**

**if char in ref1:**

**myst += ref2[ref1.index(char)]**

**else:**

**myst += char**

**print(myst)**

**# better approach using map() (not yet taught(?))**

**def vowelmap(inst):**

**def repl(char):**

**result = char**

**if char == 'a' or char == 'A':**

**result = '4'**

**elif char == 'e' or char == 'E':**

**result = '3'**

**elif char == 'i' or char == 'I':**

**result = '1'**

**elif char == 'o' or char == 'O':**

**result = '0'**

**elif char == 'u' or char == 'U':**

**result = '9'**

**return result**

**myst = map(repl, inst)**

**myst = list(myst)**

**myst = ''.join(myst)**

**return myst**

**def vowelmap\_(inst):**

**ref1 = "AEIOUaeiou"**

**ref2 = "4310943109"**

**def repl(char):**

**if char in ref1:**

**return ref2[ref1.index(char)]**

**else:**

**return char**

**myst = map(repl, inst)**

**myst = list(myst)**

**myst = ''.join(myst)**

**return myst**

**def main():**

**s = input("Enter a string to replace the vowels: ")**

**print(vowelmap\_(s))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 22**

**# Write a program to count the number of odd and even numbers in a list**

**def oddeven(l):**

**oddnumbers = evennumbers = 0**

**for x in l:**

**isodd = x % 2**

**if isodd == 1:**

**oddnumbers += 1**

**if isodd == 0:**

**evennumbers += 1**

**return (oddnumbers, evennumbers)**

**def main():**

**mylist = input("Enter a list of space separated integers: ").split()**

**mylist = [int(i) for i in mylist]**

**print(oddeven(mylist))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 23**

**# Write a function that takes a list of numbers and returns a new list with the square of each number. Write a list comprehension that does the same thing.**

**def squareeach(l):**

**result = []**

**for x in range(len(l)):**

**result.append(l[x] \*\* 2)**

**return result**

**def squarecomp(l):**

**return [x \*\* 2 for x in l]**

**def squareeven(l):**

**return [x \*\* 2 for x in l if x % 2 == 0]**

**def main():**

**mylist = input("Enter a list of space separated integers: ").split()**

**mylist = [int(i) for i in mylist]**

**print(squareeven(mylist))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 24**

**# Write a program to remove duplicate elements from a list**

**def removeduplicates(l):**

**for x in reversed(range(len(l))):**

**if l[x] in l[:x]:**

**del l[x]**

**return l**

**def main():**

**mylist = input("Enter a list of space separated values: ").split()**

**print(removeduplicates(mylist))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 25**

**# Write a program that removes all occurrences of any of a given list of values from a list.**

**def rem\_all(l, torem):**

**return [i for i in l if i not in torem]**

**def main():**

**mylist = input("Enter a list of space separated values: ").split()**

**rem = input("What to remove from your list?").split()**

**print(rem\_all(mylist, rem))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 26**

**# Write a program that takes a list as input and returns a dictionary containing key-value pairs of {key\_a: [a1, a2 ,a3, ... an], key\_b: [b1, b2, b3 ... bm]} for n instances of the value key\_a in original list, and m instances of key\_b.**

**def dictvals(l):**

**d = {}**

**# iterate over L**

**for i in l:**

**# add the value to the dictionary if it doesn't exist and give it a value of [i]**

**if i not in d:**

**d[i] = [i]**

**# if the value exists, append the value to the list**

**else:**

**d[i].append(i)**

**return d**

**# alternate implementation**

**def dictvals\_(l):**

**d = {}**

**for ele in l:**

**d[ele] = l.count(ele)**

**return d**

**def main():**

**mylist = input("Enter a list of space separated integers: ").split()**

**mylist = [int(i) for i in mylist]**

**print(dictvals(mylist))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 27**

**# Write a program that takes two lists as inputs and returns True if they have at least one common element and False otherwise.**

**def common(l1, l2):**

**result = False**

**for i in l1:**

**if i in l2:**

**result = True**

**return result**

**def main():**

**mylist1 = input("Enter a list of space separated values: ").split()**

**mylist2 = input("Enter a list of space separated values: ").split()**

**print(common(mylist1, mylist2))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 28**

**# Write a program that prints the input list after removing the elements at indices 0, 2, 4, and 5**

**def remove\_specific(l, indices = [0, 2, 4, 5]):**

**return [l[i] for i in range(len(l)) if i not in indices]**

**def main():**

**mylist = input("Enter a list of space separated values: ").split()**

**indices = input("Enter the indices to remove: ").split()**

**indices = [int(indx) for indx in indices]**

**print(remove\_specific(mylist, indices))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 29**

**# Write a program that concatenates the dictionaries {1:10,2:20} {3:30,4:40} {5:50,6:60} to create a new dictionary**

**def concat\_dicts(dict1, dict2):**

**return {\*\*dict1, \*\*dict2}**

**def main():**

**dict1 = {1:10, 2:20}**

**dict2 = {3:30, 4:40}**

**dict3 = {5:50, 6:60}**

**result = concat\_dicts(dict1, dict2)**

**result = concat\_dicts(result, dict3)**

**print(result) # {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 30**

**# Write a program to sort a tuple of tuples by the \*second\* item of those tuples.**

**# Example input: (('a', 34), ('b', 23), ('c', 67), ('d', 100), ('e', 0))**

**#def sort\_tuples(t):**

**# return sorted(t, key = lambda x: x[1])**

**def sort\_tuples(t):**

**l = []**

**for i in t:**

**l.append(i)**

**for i in range(len(l)):**

**for j in range(len(l)):**

**if l[i][1] < l[j][1]:**

**l[i], l[j] = l[j], l[i]**

**return tuple(l)**

**def main():**

**mylist = [('a', 3),('b', 5),('c', 2),('d', 4),('e', 1)]**

**print(mylist)**

**print(sort\_tuples(mylist))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 31**

**# Write a program to implement matrix multiplication.**

**def matmul(a,b):**

**r1 = len(a)**

**c2 = len(b[0])**

**if len(a[0]) != len(b):**

**raise ValueError("bad value (matrix 1 columns should equal matrix 2 rows)")**

**else:**

**r2c1 = len(b)**

**result = [[0 for x in range(c2)] for x in range(r1)]**

**# print(result)**

**for i in range(r1):**

**for j in range(c2):**

**for k in range(r2c1): # can just as well be range(len(a[0]))**

**result[i][j] += a[i][k] \* b[k][j]**

**return result**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\nUse the function matmul() to multiply two matrices.") # Assignment 32**

**# Write a program to implement a pangram checker**

**def pangramchecker(a):**

**flag = True**

**for char in "abcdefghijklmnopqrstuvwxyz":**

**if char not in a:**

**flag = False**

**break**

**return flag**

**def main():**

**string = input("Enter a word: ")**

**print(pangramchecker(string))**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 33**

**# Write a program to read only columns with specific headers from a csv files**

**import csv**

**def readspecific(colslist):**

**with open("myfile.csv", newline = '') as mycsv:**

**robj = csv.DictReader(mycsv)**

**for r in robj:**

**truncrow = []**

**for col in colslist:**

**truncrow.append(r[col])**

**for x in truncrow:**

**print(f"{x}", end = ';\t')**

**print("\n")**

**return**

**def main():**

**columns\_to\_read = input("What are the headings of the columns you want to read? ").split()**

**readspecific(columns\_to\_read)**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 34**

**# Write a program to copy one file into another using file handling methods**

**def copy\_file(a, b = "copy.txt"):**

**# a is a string representing the path to the source file**

**with open(a, 'r') as src:**

**with open(b, 'w') as tgt:**

**tgt.write(src.read())**

**return True**

**def main():**

**a = input("Write complete path to the file you want to copy ")**

**copy\_file(a)**

**with open('copy.txt', 'r') as file:**

**print(file.read())**

**print("done")**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main() # Assignment 35**

**# Write a program to generate a CSV file that has one column containing each alphabet, and one column containing a random integer. The headings of these columns should be alphabet and count respectively. Also plot a histogram over the alphabets that represents the count of each.**

**import csv**

**def tsri(seed):**

**return [((x \*\* 5) + ((seed % 631) \* x \*\* 4) + (seed \* x \*\* 3) + (18 \* x \*\* 2) + ((seed + 45) \* x) + (seed)) % ((seed // 5) + 107) for x in range(26)]**

**def makecsv(seed):**

**with open("alphabet.csv", 'w', newline='') as file:**

**w = csv.writer(file)**

**w.writerow(['alphabet','count'])**

**nums = tsri(seed)**

**chars = "abcdefghijklmnopqrstuvwxyz"**

**for i in range(len(chars)):**

**w.writerow([chars[i], nums[i]])**

**return**

**import matplotlib.pyplot as plt**

**def plotgraph():**

**letters = []**

**counts = []**

**with open('alphabet.csv', "r") as file:**

**r = csv.reader(file)**

**next(r)**

**for row in r:**

**letters.append(row[0])**

**counts.append(int(row[1]))**

**plt.bar(letters, counts)**

**plt.title('Histogram of letters and their counts')**

**plt.xlabel('Letter')**

**plt.ylabel('Count')**

**plt.grid(axis='y')**

**plt.show()**

**def main():**

**seed = input("Enter the seed for the random number generator: ")**

**makecsv(seed)**

**plotgraph()**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print("Written by Varad Kinjal Dixit, 23BBT035\n")**

**main()**