**LAB 4: PYTHON BASICS (HOMEWORK)**

Disha Jain – 220905554 – 61

1. Write a python program to select smallest element from a list in an expected linear time.

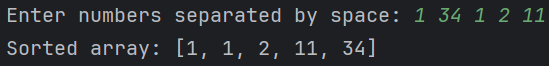
arr = list(map(int, input("Enter numbers separated by space: ").split()))  
smallest = arr[0]  
  
for val in arr:  
 if val < smallest:  
 smallest = val  
  
print(f"{smallest} is smallest")

A black background with white text

Description automatically generated

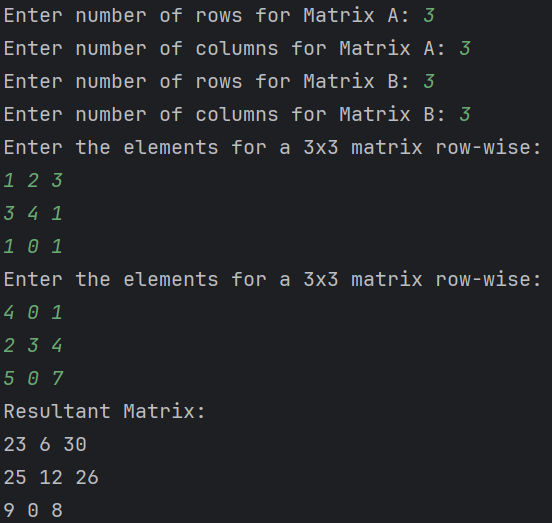
1. Write a python program to implement bubble sort.

def bubble\_sort(arr):  
 n = len(arr)  
 for i in range(n - 1):  
 swapped = False  
 for j in range(n - i - 1):  
 if arr[j] > arr[j + 1]:  
 arr[j], arr[j + 1] = arr[j + 1], arr[j] # Swap elements  
 swapped = True  
 if not swapped:  
 break # Optimization: Stop if no swaps were made in a pass  
  
# Example usage  
arr = list(map(int, input("Enter numbers separated by space: ").split()))  
bubble\_sort(arr)  
print("Sorted array:", arr)



1. Write a python program to multiply two matrices.

def read\_matrix(rows, cols):  
 print(f"Enter the elements for a {rows}x{cols} matrix row-wise:")  
 matrix = []  
 for i in range(rows):  
 row = list(map(int, input().split()))  
 if len(row) != cols:  
 print(f"Error: Please enter exactly {cols} values per row.")  
 return read\_matrix(rows, cols) # Ask for input again if incorrect  
 matrix.append(row)  
 return matrix  
  
  
def matrix\_multiplication(A, B):  
 rows\_A, cols\_A = len(A), len(A[0])  
 rows\_B, cols\_B = len(B), len(B[0])  
  
 result = [[0 for \_ in range(cols\_B)] for \_ in range(rows\_A)]  
  
 for i in range(rows\_A):  
 for j in range(cols\_B):  
 for k in range(cols\_A): # or rows\_B (since cols\_A == rows\_B)  
 result[i][j] += A[i][k] \* B[k][j]  
  
 return result  
  
  
# Read matrix dimensions from user  
rows\_A = int(input("Enter number of rows for Matrix A: "))  
cols\_A = int(input("Enter number of columns for Matrix A: "))  
  
rows\_B = int(input("Enter number of rows for Matrix B: "))  
cols\_B = int(input("Enter number of columns for Matrix B: "))  
  
# Ensure matrices can be multiplied  
if cols\_A != rows\_B:  
 print("Error: Number of columns in Matrix A must be equal to number of rows in Matrix B.")  
else:  
 # Read matrices from user  
 A = read\_matrix(rows\_A, cols\_A)  
 B = read\_matrix(rows\_B, cols\_B)  
  
 # Perform matrix multiplication  
 result = matrix\_multiplication(A, B)  
  
 # Print result  
 print("Resultant Matrix:")  
 for row in result:  
 print(" ".join(map(str, row)))



1. Write a Python class to find validity of a string of parentheses, '(', ')', '{', '}', '[' and ']. These brackets must be close in the correct order, for example "()" and "()[]{}" are valid but "[)", "({[)]" and "{{{" are invalid.

class ParenthesesValidator:  
 def is\_valid(self, s: str) -> bool:  
 stack = []  
 bracket\_map = {')': '(', '}': '{', ']': '['}  
  
 for char in s:  
 if char in bracket\_map.values(): # If it's an opening bracket  
 stack.append(char)  
 elif char in bracket\_map.keys(): # If it's a closing bracket  
 if not stack or stack.pop() != bracket\_map[char]:  
 return False # Mismatched or unbalanced bracket  
 else:  
 return False # Invalid character  
  
 return not stack # Stack should be empty if valid  
  
  
# Example usage  
validator = ParenthesesValidator()  
test\_strings = ["()", "()[]{}", "(]", "({[)]", "{{{"]  
for s in test\_strings:  
 print(f"{s}: {validator.is\_valid(s)}")

A screenshot of a computer screen

Description automatically generated

1. Write a Python class to reverse a string word by word.

class StringReverser:  
 def reverse\_words(self, s: str) -> str:  
 return " ".join(s.split()[::-1]) # Split, reverse, and join back  
  
# Example usage  
reverser = StringReverser()  
str = input("Enter a sentence: ")  
print(f"Reversed: {reverser.reverse\_words(str)}")

A black background with white text

Description automatically generated

1. Write a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle.

import math  
  
class Circle:  
 def \_\_init\_\_(self, radius):  
 self.radius = radius  
  
 def area(self):  
 return math.pi \* self.radius \*\* 2  
  
 def perimeter(self):  
 return 2 \* math.pi \* self.radius  
  
# Example usage  
radius = int(input("Enter radius: "))  
circle = Circle(radius)  
print(f"Radius: {circle.radius}")  
print(f"Area: {circle.area():.2f}")  
print(f"Perimeter: {circle.perimeter():.2f}")

A screenshot of a computer

Description automatically generated