

The screenshot shows the Visual Studio Code editor with a file named `1.py` open. The code is a Python script that defines a function `sum(arr)` to calculate the sum of elements in an array. The array is `[12, 3, 4, 15, 45, 78, 450]`. The script prints the sum of the array, which is 607.

```
1 # PROGRAM 1 A MW
2 # sum of elements in given array
3 def _sum(arr):
4     sum=0
5     for i in arr:
6         sum = sum + i
7     return(sum)
8 arr = [12, 3, 4, 15, 45, 78, 450]
9 n = len(arr)
10 ans = _sum(arr)
11 print("Practical 1A")
12 print('Sum of the array is ', ans)
13
```

The terminal output shows the execution of the script, printing "Practical 1A" and "Sum of the array is 607".

The screenshot shows the Visual Studio Code editor with a file named `1.py` open. The code is a Python script that defines a function `find(x)` to find the index of a character or numerical value in an array. The array is `['p','y','t','h','o','n']`. The script prints the index of the character 'o', which is 4.

```
1 # program 1B finding an element in an array
2 #finding character value
3 x = ['p','y','t','h','o','n']
4 print(x.index('o'))
5
6 # finding numerical value
7 x = [5,1,7,0,3,4]
8 print("Practical 1B")
9 print(x.index(7))
10
```

The terminal output shows the execution of the script, printing "Practical 1B" and "2".

The screenshot shows the Visual Studio Code editor with a file named `1.py` open. The code is a Python script that defines two functions, `getMin(arr, n)` and `getMax(arr, n)`, to find the minimum and maximum elements in an array. The array is `[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`. The script prints the minimum element of the array, which is 1, and the maximum element of the array, which is 10.

```
1 #program 1C:
2 # program to find minimum (or maximum) element in an array # Minimum Function
3 def getMin(arr, n):
4     res = arr[0]
5     for i in range(1,n):
6         res = min(res, arr[i])
7     return res
8
9 # Maximum Function
10 def getMax(arr, n):
11     res = arr[0]
12     for i in range(1,n):
13         res = max(res, arr[i])
14     return res
```

The terminal output shows the execution of the script, printing "Practical 1C", "Minimum element of array: 1", and "Maximum element of array: 10".

The screenshot shows the Visual Studio Code editor with a file named `1.py` open. The code is a Python script that defines a function `countingEvenOdd(arr, arr_size)` to count the number of even and odd elements in an array. The array is `[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`. The script prints the number of even elements, which is 6, and the number of odd elements, which is 6.

```
1
2 def countingEvenOdd(arr, arr_size):
3     even_count = 0
4     odd_count = 0
5
6     # loop to read all the values # in the array
7     for i in range(arr_size):
8         if (arr[i] & 1 == 1):
9             odd_count += 1 #odd_count=odd_count+1
10        else:
11            even_count += 1 #even_count=even_count+1
12
13    print("Number of even elements = ", even_count)
14    print("Number of odd elements = ", odd_count)
```

The terminal output shows the execution of the script, printing "Practical 1D", "Number of even elements = 6", and "Number of odd elements = 6".

The screenshot shows a Visual Studio Code editor window with a Python file named `1.py`. The code prompts the user to enter the number of rows and columns, then enters values into a matrix. The terminal output shows the execution of the script, displaying the entered matrix and the sums of each row and column.

```
college > 1.py > ...
1 print("Practical 2A")
2 n = int(input("Enter the number of rows:"))
3 m = int(input("Enter the number of columns:"))
4 matrix = []
5 print("Enter values in matrix :")
6 # For user input
7 for i in range(n):
8     for j in range(m):
9         matrix.append(int(input()))
10
11 # Sum of row
12 for i in range(n):
13     sum = 0
14     for j in range(m):
15         sum += matrix[i][j]
16     print("Sum of row", i, ":", sum)
17
18 # Sum of column
19 for j in range(m):
20     sum = 0
21     for i in range(n):
22         sum += matrix[i][j]
23     print("Sum of column", j, ":", sum)
```

Terminal Output:

```
PS C:\Users\acer\Desktop\python> c:\Users\acer\AppData\Local\Programs\Python\Python310\python.exe c:\Users\acer\Desktop\python\college/1.py
Practical 2A
Enter the number of rows:3
Enter the number of columns:3
Enter values in matrix :
1
2
3
4
5
6
7
8
9
1 2 3
4 5 6
7 8 9
Sum of row 1 : 6
Sum of row 2 : 15
Sum of row 3 : 24
Sum of column 0 : 12
Sum of column 1 : 15
Sum of column 2 : 18
PS C:\Users\acer\Desktop\python>
```

The screenshot shows a Visual Studio Code editor window with a Python file named `1.py`. The code defines a function `printDiagonalSums` that takes a matrix and its dimensions as input and prints the sums of the principal and secondary diagonals. The terminal output shows the execution of the script, displaying the matrix and the calculated diagonal sums.

```
college > 1.py > printDiagonalSums
1 MAX = 100
2 def printDiagonalSums(mat, n):
3     principal = 0
4     secondary = 0
5     for i in range(0, n):
6         for j in range(0, n):
7             # Condition for principal diagonal
8             if (i == j):
9                 principal += mat[i][j]
10            # Condition for secondary diagonal
11            if ((i + j) == (n - 1)):
12                secondary += mat[i][j]
13
14    print("Principal Diagonal:", principal)
15    print("Secondary Diagonal:", secondary)
```

Terminal Output:

```
PS C:\Users\acer\Desktop\python> c:\Users\acer\AppData\Local\Programs\Python\Python310\python.exe c:\Users\acer\Desktop\python\college/1.py
Principal Diagonal: 18
Secondary Diagonal: 18
Practical 2B
PS C:\Users\acer\Desktop\python>
```

The screenshot shows a Visual Studio Code editor window with a Python file named `1.py`. The code defines two matrices `X` and `Y`, and calculates their product `result`. The terminal output shows the execution of the script, displaying the matrices and the resulting product matrix.

```
college > 1.py > ...
1 X = [[1,2,3],
2       [4,5,6],
3       [7,8,9]]
4
5 Y = [[9,8,7],
6       [6,5,4],
7       [3,2,1]]
8 result = [[0,0,0],
9           [0,0,0],
10          [0,0,0]]
11
12 # iterate through rows
13 for i in range(len(X)):
14     # iterate through columns
15     for j in range(len(X[0])):
```

Terminal Output:

```
PS C:\Users\acer\Desktop\python> c:\Users\acer\AppData\Local\Programs\Python\Python310\python.exe c:\Users\acer\Desktop\python\college/1.py
[10, 10, 10]
[10, 10, 10]
[10, 10, 10]
Practical 2C
PS C:\Users\acer\Desktop\python>
```

The screenshot shows a Visual Studio Code editor window with a Python file named `1.py`. The code defines two matrices `A` and `B`, and calculates their product `result`. The terminal output shows the execution of the script, displaying the matrices and the resulting product matrix.

```
college > 1.py > ...
1 A = [[12, 7, 3],
2       [4, 5, 6],
3       [7, 8, 9]]
4
5 # take a 3x4 matrix
6 B = [[5, 8, 1, 2],
7       [6, 7, 3, 0],
8       [4, 5, 9, 1]]
9
10 result = [[0, 0, 0, 0],
11           [0, 0, 0, 0],
12           [0, 0, 0, 0]]
13
14 # iterating by row of A
15 for i in range(len(A)):
```

Terminal Output:

```
PS C:\Users\acer\Desktop\python> c:\Users\acer\AppData\Local\Programs\Python\Python310\python.exe c:\Users\acer\Desktop\python\college/1.py
[114, 100, 60, 27]
[141, 97, 71, 34]
[119, 157, 112, 23]
Practical 2D
PS C:\Users\acer\Desktop\python>
```

```
File Edit Selection View Go Run Terminal Help
1.py - python - Visual Studio Code
1.py x photo.jpg
college > 1.py > Stack > mit
30 print('pop')
31 print('quit')
32 do = input('What would you like to do? ').split()
33 operation = do[0].strip().lower()
34 if operation == 'push':
35     a_stack.push(int(do[1]))
36 elif operation == 'pop':
37     popped = a_stack.pop()
38     if popped is None:
39         print('Stack is empty.')
40     else:
41         print('Popped value: ', int(popped))
42 elif operation == 'quit':
43     break
44
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\acer\Desktop\python> & C:\Users\acer\AppData\Local\Programs\Python\Python310/python.exe c:\Users\acer\Desktop\python\college/1.py

Practical 3

push value>

pop

quit

What would you like to do? 15

push value>

pop

quit

What would you like to do? 12

push value>

pop

quit

Ln 7, Col 23 Spaces: 4 UTF-8 CR LF Python 3.10.7 64-bit 4:05 PM 2/8/2023

```
File Edit Selection View Go Run Terminal Help
1.py - python - Visual Studio Code
1.py x photo.jpg
college > 1.py > ...
1 def LinearSearch(array, n, k):
2     for j in range(0, n):
3         if array[j] == k:
4             return j
5     return -1
6
7 print("Practical 4A")
8 array = [1, 3, 5, 7, 9]
9
10 k = 7
11 n = len(array)
12
13 result = LinearSearch(array, n, k)
14 if result == -1:
15     print("Element not found")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\acer\Desktop\python> & C:\Users\acer\AppData\Local\Programs\Python\Python310/python.exe c:\Users\acer\Desktop\python\college/1.py

Practical 4A

Element found at index: 3

PS C:\Users\acer\Desktop\python>

Ln 6, Col 1 Spaces: 4 UTF-8 CR LF Python 3.10.7 64-bit 4:07 PM 2/8/2023

```
File Edit Selection View Go Run Terminal Help
1.py - python - Visual Studio Code
1.py x photo.jpg
college > 1.py > ...
1 def BinarySearch(arr, k, low, high):
2     if high >= low:
3         mid = low + (high - low)//2
4         if arr[mid] == k:
5             return mid
6         elif arr[mid] > k:
7             return BinarySearch(arr, k, low, mid-1)
8         else:
9             return BinarySearch(arr, k, mid + 1, high)
10     else:
11         return -1
12
13 print("Practical 4B")
14 arr = [1, 3, 5, 7, 9, 15, 16, 14, 45]
15 k = 15
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\acer\Desktop\python> & C:\Users\acer\AppData\Local\Programs\Python\Python310/python.exe c:\Users\acer\Desktop\python\college/1.py

Practical 4B

Element is present at index 5

PS C:\Users\acer\Desktop\python>

Ln 16, Col 46 Spaces: 4 UTF-8 CR LF Python 3.10.7 64-bit 4:09 PM 2/8/2023

```
File Edit Selection View Go Run Terminal Help
1.py - python - Visual Studio Code
1.py x photo.jpg
college > 1.py > ...
1 def bubble_sort(list1):
2     # Outer loop for traverse the entire list
3     for i in range(0, len(list1)-1):
4         for j in range(len(list1)-1):
5             if(list1[j]>list1[j+1]):
6                 temp = list1[j]
7                 list1[j] = list1[j+1]
8                 list1[j+1] = temp
9     return list1
10
11 print("Practical 5A")
12 list1 = [5, 3, 8, 6, 7, 2]
13 print("The unsorted list is: ", list1) # Calling the bubble sort function
14 print("The sorted list is: ", bubble_sort(list1))
15
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\acer\Desktop\python> & C:\Users\acer\AppData\Local\Programs\Python\Python310/python.exe c:\Users\acer\Desktop\python\college/1.py

Practical 5A

The unsorted list is: [5, 3, 8, 6, 7, 2]

The sorted list is: [3, 5, 6, 7, 2, 8]

PS C:\Users\acer\Desktop\python>

Ln 11, Col 20 Spaces: 4 UTF-8 CR LF Python 3.10.7 64-bit 4:11 PM 2/8/2023

```
File Edit Selection View Go Run Terminal Help
1.py - python - Visual Studio Code

college > 1.py > bubble_sort
1 def bubble_sort(list1):
2     total_iteration = 0
3     while(has_swapped):
4         has_swapped = False
5         for i in range(len(list1) - total_iteration - 1):
6             if list1[i] > list1[i+1]:
7                 # Swap
8                 list1[i], list1[i+1] = list1[i+1], list1[i]
9                 has_swapped = True
10                total_iteration += 1
11            print("The number of iteration: ", total_iteration)
12            return list1
13
14 print("Practical 5A - 2")
15 list1 = [5, 3, 8, 6, 7, 2]
16 print("The unsorted list is: ", list1) # Calling the bubble sort function

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\acer\Desktop\python> & C:\Users\acer\AppData\Local\Programs\Python\Python310\python.exe c:/Users/acer/Desktop/python/college/1.py
Practical 5A - 2
The unsorted list is: [5, 3, 8, 6, 7, 2]
The number of iteration: 5
The sorted list is: [3, 5, 6, 7, 2, 8]
PS C:\Users\acer\Desktop\python>
```

```
File Edit Selection View Go Run Terminal Help
1.py - python - Visual Studio Code

college > 1.py > selectionSort
1 def selectionSort( itemList ):
2     n = len( itemList )
3     for i in range( n - 1 ):
4         minValueIndex = i
5         for j in range( i + 1, n ):
6             if itemList[j] < itemList[minValueIndex] :
7                 minValueIndex = j
8
9         if minValueIndex != i :
10            temp = itemList[i]
11            itemList[i] = itemList[minValueIndex]
12            itemList[minValueIndex] = temp
13        return itemList
14
15 print("Prcticalm 5B")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\acer\Desktop\python> & C:\Users\acer\AppData\Local\Programs\Python\Python310\python.exe c:/Users/acer/Desktop/python/college/1.py
Prcticalm 5B
[3, 6, 9, 21, 33]
PS C:\Users\acer\Desktop\python>
```

```
File Edit Selection View Go Run Terminal Help
1.py - python - Visual Studio Code

college > 1.py > insertion_sort
1 def insertion_sort(list1):
2     for i in range(1, len(list1)):
3         value = list1[i]
4         j = i - 1
5         while j >= 0 and value < list1[j]:
6             list1[j + 1] = list1[j]
7             j -= 1
8         list1[j + 1] = value
9     return list1
10 # Driver code to test above
11
12 print("Prctical 5C")
13 list1 = [10, 5, 13, 8, 2]
14 print("The unsorted list is:", list1)
15 print("The sorted list1 is:". insertion_sort(list1))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\acer\Desktop\python> & C:\Users\acer\AppData\Local\Programs\Python\Python310\python.exe c:/Users/acer/Desktop/python/college/1.py
Prctical 5C
The unsorted list is: [10, 5, 13, 8, 2]
The sorted list1 is: [2, 5, 8, 10, 13]
PS C:\Users\acer\Desktop\python>
```