Lab 4 – HA – Python Basics

A1.

```
import random

def quickselect(arr, k):
if len(arr) == 1:
return arr[0]
pivot = random.choice(arr)
lows = [el for el in arr if el < pivot]
highs = [el for el in arr if el > pivot]
pivots = [el for el in arr if el == pivot]
if k < len(lows):
return quickselect(lows, k)
elif k < len(lows) + len(pivots):
return pivots[0]
else:
return quickselect(highs, k - len(lows) - len(pivots))

if __name__ == "__main__":
nums = [14, 30, 25, 47, 4, 20, 26]
print(f"Input Data: {nums}")
print("Smallest element:", quickselect(nums, 0))
```

A2.

```
def bubble_sort(arr):
n = len(arr)
for i in range(n):
for j in range(0, n - i - 1):
if arr[j] > arr[j + 1]:
arr[j], arr[j + 1] = arr[j + 1], arr[j]
return arr

if __name__ == "__main__":
user_data = input("Enter numbers to sort separated by spaces: ")
data = [int(x) for x in user_data.split()]
print(f"Input Data: {data}")
print("Sorted array:", bubble_sort(data))
```

A3.

```
def multiply_matrices(A, B):
result = [[0 for _ in range(len(B[0]))] for _ in range(len(A))]
for i in range(len(A)):
for j in range(len(B[0])):
```

```
for k in range(len(B)):
result[i][j] += A[i][k] * B[k][j]
return result

def get_matrix_input(name):
rows = int(input(f"Enter rows for {name}: "))
mat = []
for i in range(rows):
row = [int(x) for x in input(f"Enter row {i+1} values separated by spaces: ").split()]
mat.append(row)
return mat

if __name__ == "__main__":
mat1 = get_matrix_input("Matrix A")
mat2 = get_matrix_input("Matrix B")
print(f"Input Matrix A: {mat1}")
print(f"Input Matrix B: {mat2}")
if len(mat1[0]) != len(mat2):
print("Error: Columns of A must match rows of B.")
else:
print("Resultant Matrix:")
for row in multiply_matrices(mat1, mat2):
print(row)
```

A4.

```
class ValidityChecker:
def is_valid(self, sequence):
stack = []
mapping = {")": "(", "}": "{", "]": "["}
for char in sequence:
if char in mapping.values():
stack.append(char)
elif char in mapping.keys():
if not stack or mapping[char] != stack.pop():
return False
return not stack

if __name__ == "__main__":
checker = ValidityChecker()
user_seq = input("Enter a bracket sequence (e.g., ()[]{}): ")
print(f"Input Data: {user_seq}")
print(f"Is valid: {checker.is_valid(user_seq)}")
```

A5.

```
class StringReverser:
```

```
def reverse_words(self, s):
words = s.split()
return " ".join(reversed(words))

if __name__ == "__main__":
reverser = StringReverser()
user_input = input("Enter a sentence to reverse: ")
print(f"Input Data: {user_input}")
print("Reversed:", reverser.reverse_words(user_input))
```

A6.

```
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

if __name__ == "__main__":
r = float(input("Enter radius: "))
my_circle = Circle(r)
print(f"Input Radius: {r}")
print(f"Area: {my_circle.area():.2f}")
print(f"Perimeter: {my_circle.perimeter():.2f}")
```

Outputs:

```
WP_C1@CL3-23:~/Documents/230905152_WP/L4$ python3 a1.py
Input Data: [14, 30, 25, 47, 4, 20, 26]
Smallest element: 4
WP_C1@CL3-23:~/Documents/230905152_WP/L4$ python3 a2.py
Enter numbers to sort separated by spaces: 12 45 23 78 12 67
Input Data: [12, 45, 23, 78, 12, 67]
Sorted array: [12, 12, 23, 45, 67, 78]
WP_C1@CL3-23:~/Documents/230905152_WP/L4$ python3 a3.py
Enter rows for Matrix A: 2
Enter row 1 values separated by spaces: 1 2
Enter row 2 values separated by spaces: 3 4
Enter rows for Matrix B: 2
Enter row 1 values separated by spaces: 1 2
Enter row 2 values separated by spaces: 3 4
Input Matrix A: [[1, 2], [3, 4]]
Input Matrix B: [[1, 2], [3, 4]]
Resultant Matrix:
[7, 10]
[15, 22]
WP_C1@CL3-23:~/Documents/230905152_WP/L4$ python3 a4.py
Enter a bracket sequence (e.g., ()[]{}): [{[]}]
Input Data: [{[]}]
Is valid: True
WP_C1@CL3-23:~/Documents/230905152_WP/L4$ python3 a5.py
Enter a sentence to reverse: python is great
Input Data: python is great
Reversed: great is python
WP_C1@CL3-23:~/Documents/230905152_WP/L4$ python3 a6.py
Enter radius: 3
Input Radius: 3.0
Area: 28.27
Perimeter: 18.85
WP_C1@CL3-23:~/Documents/230905152_WP/L4$
```