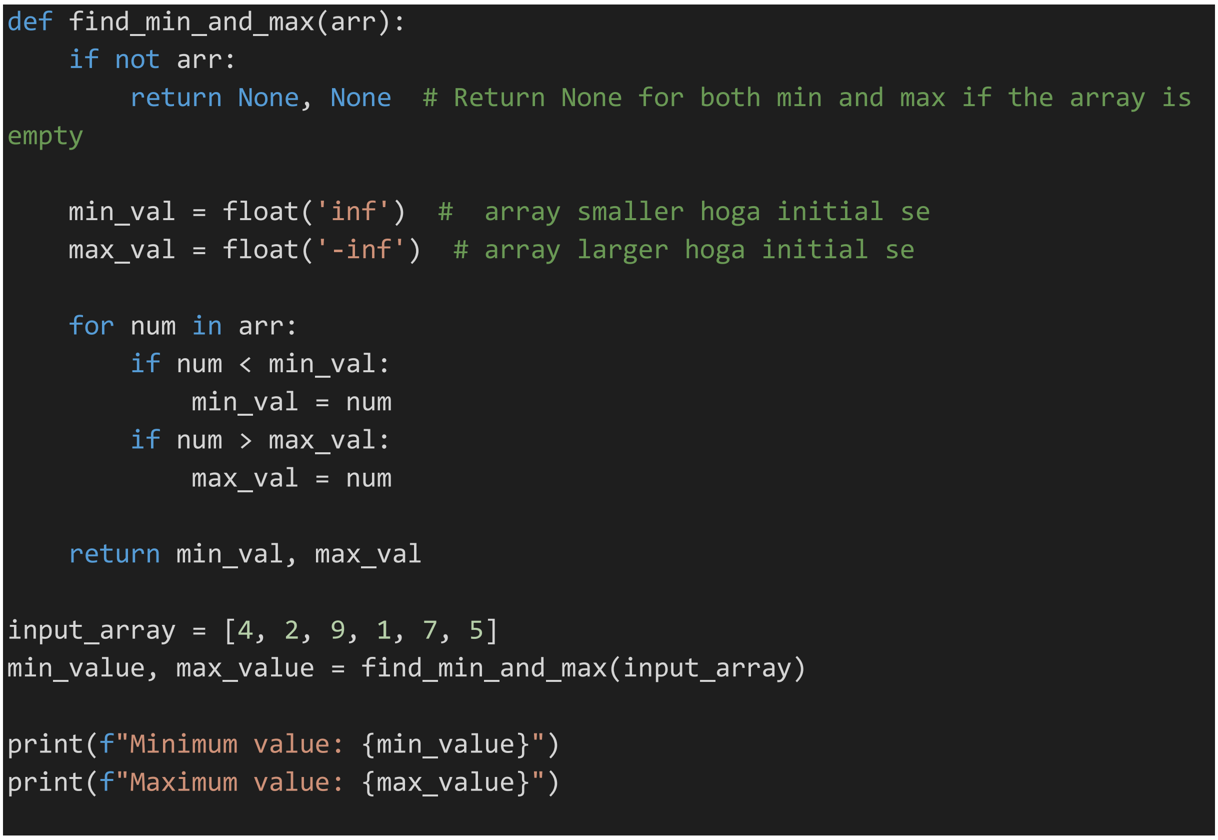
EXPERIMENT 5

PRAKASH P. BISWAS
9947

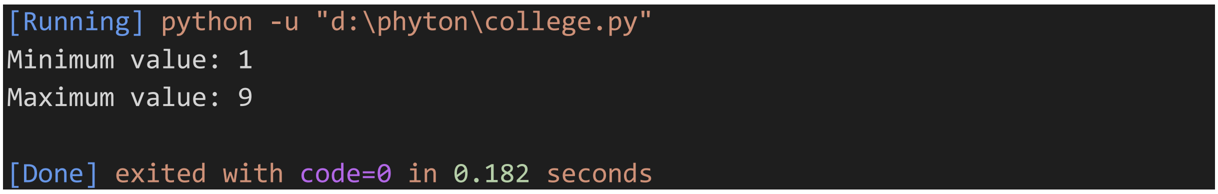
COMPS-B

MINMAX



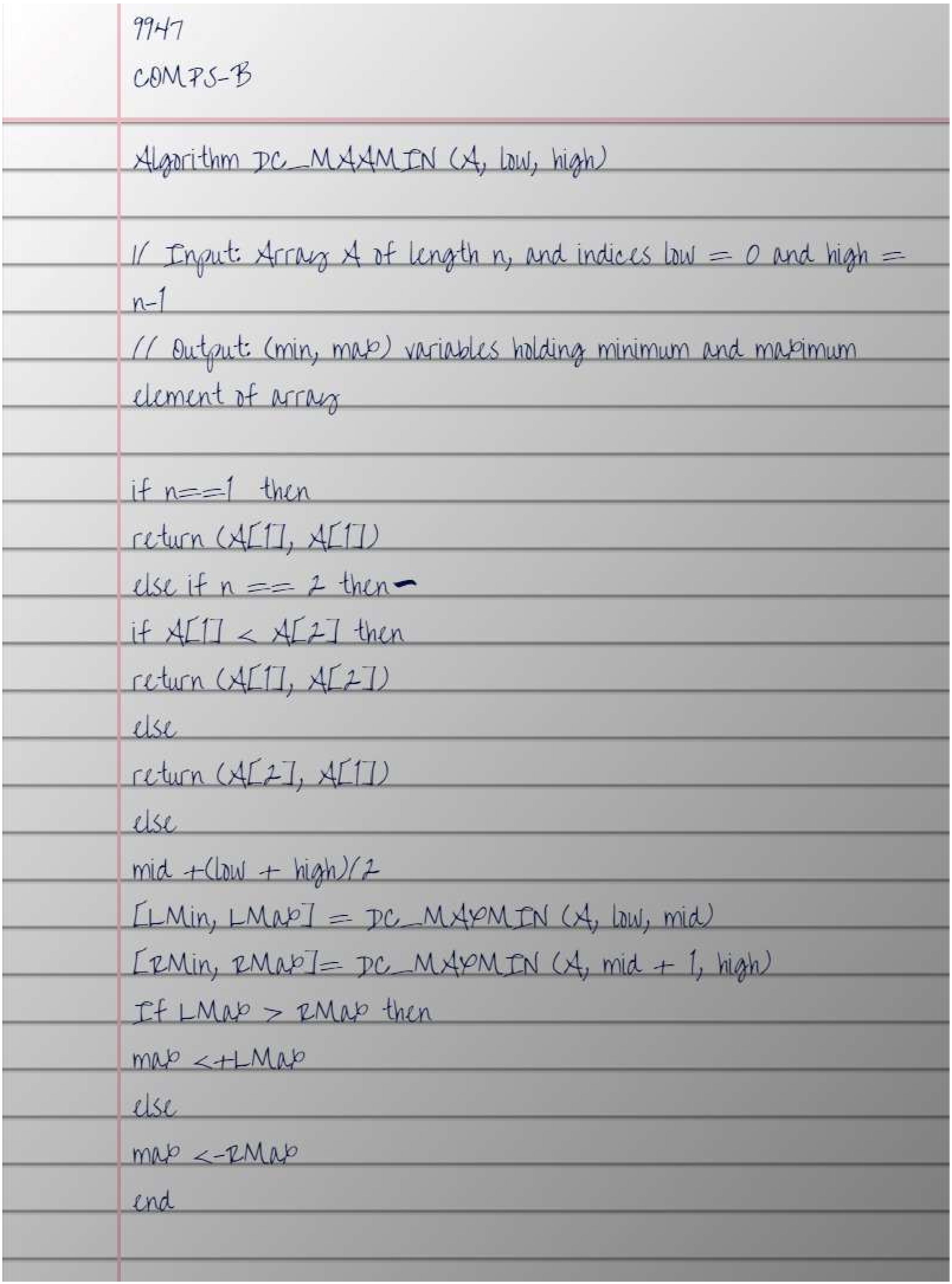
```
def find_min_and_max(arr):
    if not arr:
        return None, None  # Return None for both min and max if the array is
empty

    min_val = float('inf')  #  array smaller hoga initial se
    max_val = float('-inf')  # array larger hoga initial se

    for num in arr:
        if num < min_val:
            min_val = num
        if num > max_val:
            max_val = num

    return min_val, max_val

input_array = [4, 2, 9, 1, 7, 5]
min_value, max_value = find_min_and_max(input_array)

print(f"Minimum value: {min_value}")
print(f"Maximum value: {max_value}")
```

OUTPUT:



```
[Running] python -u "d:\phyton\college.py"
Minimum value: 1
Maximum value: 9

[Done] exited with code=0 in 0.182 seconds
```

POSTLAB:



9947

COMPS-B

Algorithm DC_MAXMIN (A, low, high)

// Input: Array A of length n, and indices low = 0 and high = n-1

// Output: (min, max) variables holding minimum and maximum element of array

```
if n==1  then
return (A[1], A[1])
else if n == 2 then
if A[1] < A[2] then
return (A[1], A[2])
else
return (A[2], A[1])
else
mid ←(low + high)/2
[LMin, LMax] = DC_MAXMIN (A, low, mid)
[RMin, RMax]= DC_MAXMIN (A, mid + 1, high)
If LMax > RMax then
max <-LMax
else
max <-RMax
end
```

```
if LMin < RMin then
min <----LMin
else
min <------- RMin
end
return (min, max)
end
```

*For finding the minimum: O(n)
*For finding the maximum: O(n)
The total time complexity of the straightforward method is O(2n), but we simplify it to O(n) for big-O notation.

Comparison:
Both methods use a linear scan through the array, and their time complexities are both O(n). However, the minimum-maximum algorithm is more efficient because it performs the same task in a single pass through the array, whereas the straightforward method requires two separate passes.