NOTE: Answers must be hand-written, capture the image and post below before Monday 24th March 2025.

Tutorial 1 (Nurul Fariza , A194778)

1. a) Find gcd(85, 34) by tracing the :

   i. Euclid algorithm

gcd (85, 34)

i) Euclid Algorithm

Step	m	n	m mod n	r	new (a, b)
1	85	34	85 mod 34	17	(34, 17)
2	34	17	34 mod 17	0	(17, 0)

∴ gcd (85, 34) = 17

   ii. Consecutive integer checking algorithm

b) Consecutive integer

step	q	85 mod q	34 mod q
1	34	85 mod 34 = 17	34 mod 34 = 0
2	33	85 mod 33 = 19	34 mod 33 = 1
3	32	85 mod 32 = 21	34 mod 32 = 2
4	31	85 mod 31 = 23	34 mod 31 = 3
5	30	85 mod 30 = 25	34 mod 30 = 4
6	29	85 mod 29 = 27	34 mod 29 = 5
7	28	85 mod 28 = 1	34 mod 28 = 6
8	27	85 mod 27 = 4	34 mod 27 = 7
9	26	85 mod 26 = 7	34 mod 26 = 8
10	25	85 mod 25 = 10	34 mod 25 = 9
11	24	85 mod 24 = 13	34 mod 24 = 10
12	23	85 mod 23 = 16	34 mod 23 = 11
13	22	85 mod 22 = 19	34 mod 22 = 12
14	21	85 mod 21 = 1	34 mod 21 = 13
15	20	85 mod 20 = 5	34 mod 20 = 14
16	19	85 mod 19 = 9	34 mod 19 = 15
17	18	85 mod 18 = 13	34 mod 18 = 16
18	17	85 mod 17 = 0	34 mod 17 = 0

∴ gcd (85, 34) = 17

*use the algorithms from the lecture slide

b. Compare the number of steps taken for both algorithms

b) Euclidean Algorithm has 2 steps taken while consecutive interger checking algorithm has 18 steps taken.

(5 marks)

2. Discuss the importance of sorting in your daily life by giving an example.

2) Sorting helps to organize information efficiently and make taks easier and saving times. Example, Shoppee uces sorting to help user find product efficiently and can sort result by price (low to high or high to low), popularity, ranting or newest arrivals.

3. Consider the algorithm for the sorting problem that sorts an array by counting, for each of its elements, the number of smaller elements and then uses this information to put the element in its appropriate position in the sorted array:

```
ALGORITHM ComparisonCountingSort(A[0..n − 1])
//Sorts an array by comparison counting
//Input: Array A[0..n − 1] of orderable values
//Output: Array S[0..n − 1] of A’s elements sorted
// in nondecreasing order
for i ←0 to n − 1 do
      Count[i]←0
for i ←0 to n − 2 do
       for j ←i + 1 to n − 1 do
               if A[i]<A[j ]
                       Count[j ]←Count[j ]+ 1
               else Count[i]←Count[i]+ 1
for i ←0 to n − 1 do
```

```
    S[Count[i]]←A[i]
return S
```

Apply this algorithm to sort the list 52, 16, 81, 98, 41, 74.

3) A = [52, 16, 81, 98, 41, 74]

count = [0, 0, 0, 0, 0, 0]

i	j	A[i]	A[j]	comparision	count
0	1	52	16	52 > 16	count[0] += 1
0	2	52	81	52 < 81	count[2] += 1
0	3	52	98	52 < 98	count[3] += 1
0	4	52	41	52 > 41	count[0] += 1
0	5	52	74	52 < 74	count[5] += 1
1	2	16	81	16 < 81	count[2] += 1
1	3	16	98	16 < 98	count[3] += 1
1	4	16	41	16 < 41	count[4] += 1
1	5	16	74	16 < 74	count[5] += 1
2	3	81	98	81 < 98	count[3] += 1
2	4	81	41	81 > 41	count[2] += 1
2	5	81	74	81 > 74	count[2] += 1
3	4	98	41	98 > 41	count[3] += 1
3	5	98	74	98 > 74	count[3] += 1
4	5	41	74	41 < 74	count[5] += 1

count = [2, 0, 4, 5, 1, 3]

A[i]	Count[i]	position in S
52	2	S[2] = 52
16	0	S[0] = 16
81	4	S[4] = 81
98	5	S[5] = 98
41	1	S[1] = 41
74	3	S[3] = 74

sorted array S :

S = [16, 41, 52, 74, 81, 98]