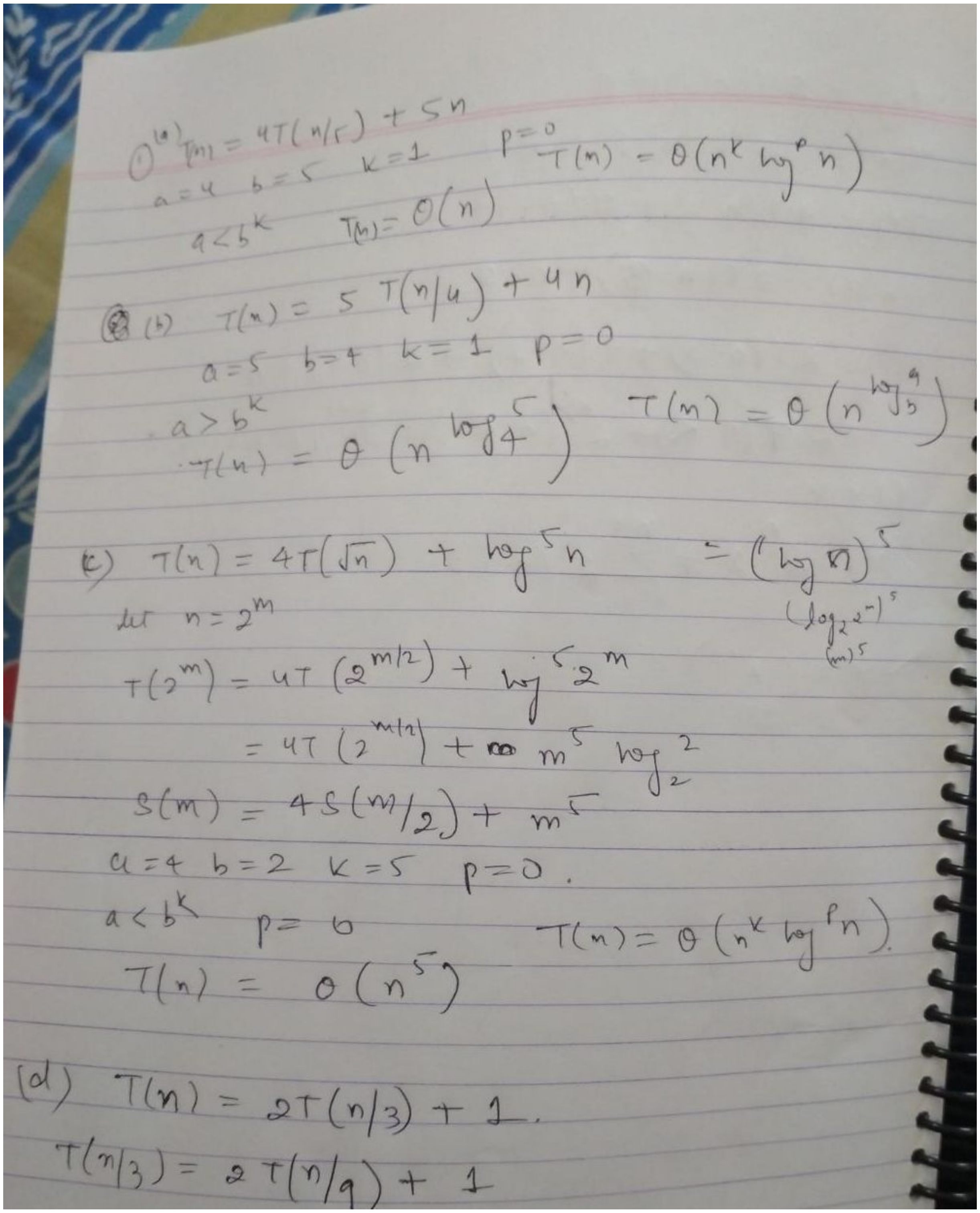
Problem-1



① (a) $T(n) = 4T(n/5) + 5n$

$a=4 \quad b=5 \quad k=1 \quad p=0 \qquad T(n) = \theta(n^k \log^p n)$

$a<b^k \qquad T(n)= \theta(n)$

(b) $T(n) = 5\,T(n/4) + 4n$

$a=5 \quad b=4 \quad k=1 \quad p=0$

$a>b^k \qquad T(n) = \theta(n^{\log_b^a})$

$T(n) = \theta(n^{\log_4^5})$

(c) $T(n) = 4T(\sqrt{n}) + \log^5 n \qquad = (\log n)^5$

$(\log_2 2^m)^5$

$(m)^5$

let $n = 2^m$

$T(2^m) = 4T(2^{m/2}) + \log^5 2^m$

$= 4T(2^{m/2}) + m^5 \log_2 2$

$S(m) = 4S(m/2) + m^5$

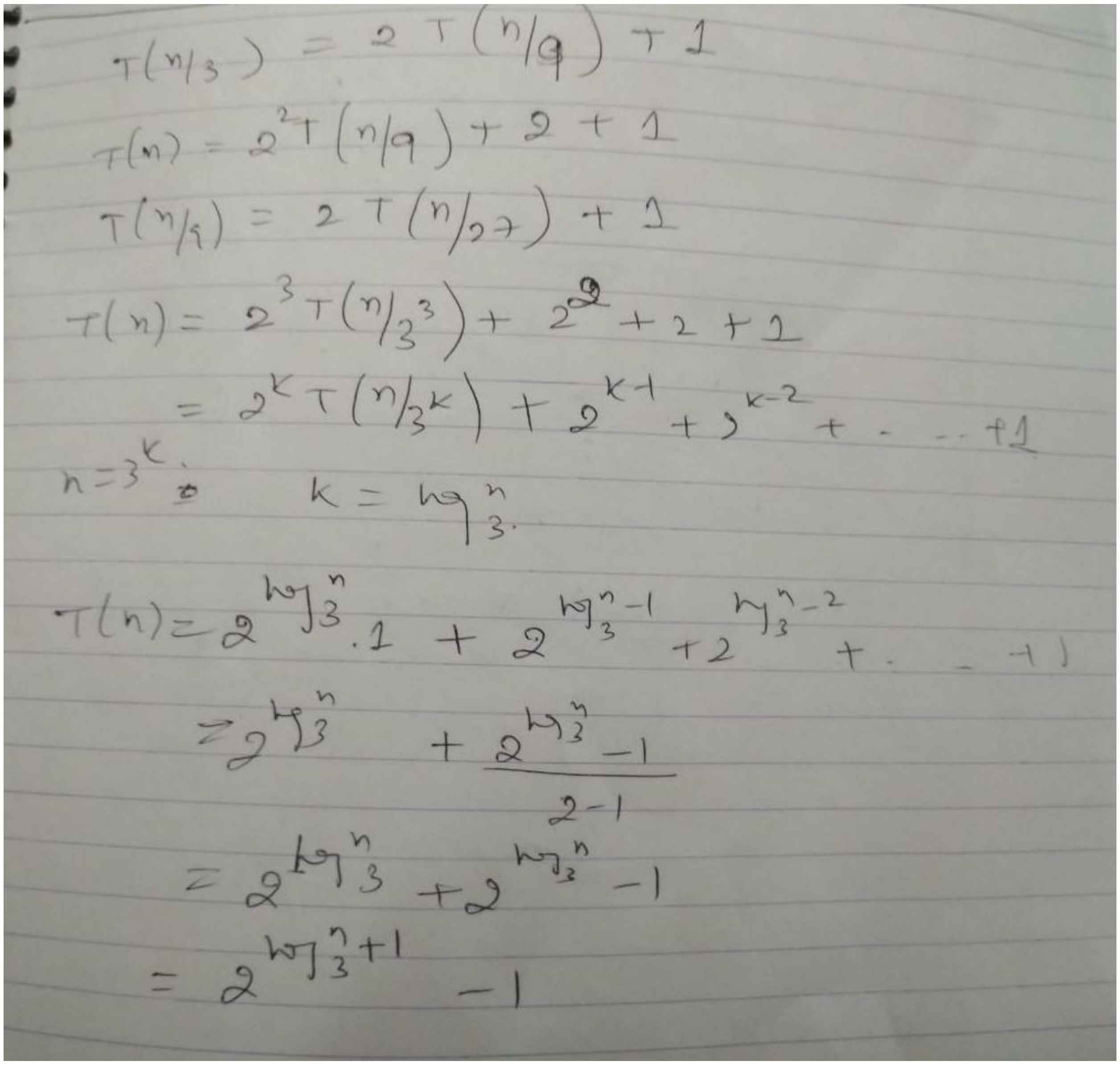
$a=4 \quad b=2 \quad k=5 \quad p=0.$

$a<b^k \quad p=0 \qquad T(n)= \theta(n^k \log^p n).$

$T(n) = \theta(n^5)$

(d) $T(n) = 2T(n/3) + 1.$

$T(n/3) = 2T(n/9) + 1$



$$T(n/3) = 2T(n/9) + 1$$

$$T(n) = 2^2 T(n/9) + 2 + 1$$

$$T(n/9) = 2T(n/27) + 1$$

$$T(n) = 2^3 T(n/3^3) + 2^2 + 2 + 1$$

$$= 2^k T(n/3^k) + 2^{k-1} + 2^{k-2} + \dots + 1$$

$n = 3^k \therefore \quad k = \log_3^n$

$$T(n) = 2^{\log_3^n} \cdot 1 + 2^{\log_3^n - 1} + 2^{\log_3^n - 2} + \dots + 1$$

$$= 2^{\log_3^n} + \frac{2^{\log_3^n} - 1}{2-1}$$

$$= 2^{\log_3^n} + 2^{\log_3^n} - 1$$

$$= 2^{\log_3^n + 1} - 1$$

e)

$$
\begin{aligned}
T(n) &= 2T(n-1) + n \\
&= 4T(n-2) + 2(n-1) + n \\
&= 8T(n-3) + 4(n-2) + 2(n-1) + n \\
&\vdots \\
&= 2^k T(n-k) + \sum_{i=0}^{k-1} 2^i (n-i) \\
&\vdots \\
&= 2^n T(0) + \sum_{i=0}^{n-1} 2^i (n-i) \\
&= 2^n + n(2^n - 1) - \sum_{i=0}^{n-1} i2^i \\
&= 2^n + n(2^n - 1) - \sum_{i=1}^{n-1} \sum_{j=1}^{i} 2^i \\
&= 2^n + n(2^n - 1) - \sum_{j=1}^{n-1} \sum_{i=j}^{n-1} 2^i \\
&= 2^n + n(2^n - 1) - \sum_{j=1}^{n-1} (2^n - 2^j) \\
&= 2^n + n(2^n - 1) - n2^n + \sum_{j=1}^{n-1} 2^j \\
&= 2^n - n + 2^n - 2 \\
&= 2^{n+1} - n - 2\ .
\end{aligned}
$$

Rubric :
2 points for each part .
Total 10 marks

Problem-2

```
Transition(A, i, j)
          - mid ← b(i + j)/2c
          - if (A[mid] = 0) - if (A[mid + 1] = 1)
                    return(mid + 1)
          - else
                    return(Transition(A, mid + 1, j))
- return(FindTransition(A, i, mid))
```

Recurrence relation: T(n) ≤ T($\lceil n/2 \rceil$) + c; T(2) = d
Running time: T(n) = O(log n).

Rubric:
4 marks (algo)
1 mark(time complexity)

Problem-3

```
CheckSum(A, w, i, j)
    - if ($j \le i$)
        return(“No”)
    - if ($A[i] + A[j] = w$)
        return(“Yes”)
    - if ($A[i] + A[j] > w$)
        return(CheckSum(A, w, i, $j - 1$))
    - else
        return(CheckSum(A, w, $i + 1$, j))
```

Running time: The recurrence relation for the running time $T(n)$ is given by: $T(n) = T(n - 1) + \Theta(1)$; $T(1) = \Theta(1)$. Solving the recurrence we get that $T(n) = \Theta(n)$.

We can prove correctness of the algorithm using induction. Let $P(k)$ denote the proposition that for all $i < j$ such that $j - i = k$ the algorithm correctly determines if there are two indices in the subarray $A[i]...A[j]$ which sum to w.

Base case: $k = 0$ is trivial since the algorithm will correctly return “No”.

Inductive step: Assume that $P(0), ..., P(k)$ are true. Let us try proving $P(k+ 1)$. Consider arbitrary indices $i < j$ such that $j - i = k + 1$. If $A[i] + A[j] = w$, then the algorithm correctly outputs “yes” in line 2. If $A[i] + A[j] > w$, then there cannot exist index $i 0 > i$ such that $A[i 0] + A[j] = w$, so a suitable index pair in subarray $A[i]...A[j]$ exists if and only if a suitable index pair in subarray $A[i]...A[j - 1]$ exists. So, again the algorithm in this case returns the correct answer because of our induction hypothesis. The final case $A[i] + A[j] < w$ is symmetric to the previous case.

Rubric :

Algo / Pseudo 2 Marks
Time Complexity 2 Marks
Correctness 1 Mark

Problem-4

Best Case analysis

Naturally, the algorithm iterates over the preference list of men. So, in the best case, it can happen that every man gets paired with the first woman in their preference list. Since there are n men, the algorithm takes O(n) time to run, in the best case Scenario.

Worst Case analysis

This case is the extreme opposite of the previous argument where every man's stable pair is the last woman in his list. In this case, the algorithm eventually checks each and every woman in a man's preference list and tries to match with the man. This takes O(n) time as every man proposes a woman at most once (Observation 2). Now, since there are n such men, the worst case runtime of Stable Marriage algorithm is $O(n^2)$

Average Case analysis

So, on an average, it takes Θ(n*(n+1)/2n) = Θ((n+1)/2) = Θ(n) time to find a stable partner for a man i. Since there are n men and all of them end up getting a partner, the algorithm takes $\Theta(n^2)$ time to run on an average.

Rubric :

Worst Case 2 Marks

Best Case 2 Marks

Avg Case 1 Mark

Problem-5

First Part:

Ceiling($Log_3 n$)

Second Part:

1. Suppose Array 'a' contains n balls.
2. First divide 'a' either in three equal parts or two of same size and third one with one element difference. Compare equal size array using scale, if one out of these two is heavy then return heaviest otherwise return third one. Update 'a' with returned value. Repeat step 2 until single ball remains in array.
3. That remaining single ball would be heavy ball.

Recursion Relation:

T(n)=T(n/3)+1

T(1)=1

T(2)=1

T(3)=1

Solving recursion relation:

T(n)=(T(n/9)+1)+1   as T(n/3)=T(n/9)+1

T(n)=((T(n/27)+1)+1)+1 as T(n/9)=T(n/27)+1

T(n)=(((T(n/81)+1)+1)+1)+1 as T(n/27)=T(n/81)+1

Observed

T(n)=T(n/ 3^k)+k

Suppose n= 3^k then k= $Log_3 n$

T(3^k)=T(1)+k

T(3^k)= $Log_3 n$

T(n)= $Log_3 n$

Rubric: if First and second part correct and complete (15)

If only first part complete and second part is paritaily right (10)

If only first part is correct or try to attempt means nearest to answer (5)

If not attempt, not submitted, both parts wrong, first part wrong (0)