Ques 1:

a)

i) $T(n) = 8T\left(\frac{n}{4}\right) + n^2 \log n$

$a = 8 \quad b = 4 \quad k = 2 \quad p = 1$

$a = 8 \quad b^k = 4^2 = 16$

$a < b^k$ and $p > 0$

$T(n) = \Theta(n^k \log^p n) = \Theta(n^2 \log n)$

ii) $T(n) = 9T\left(\frac{n}{3}\right) + \log n$

$a = 9 \quad b = 3 \quad k = 0 \quad p = 1$

$a = 9 \quad b^k = 1$

$a > b^k$

$T(n) = \Theta(n^{\log_3 9}) = \Theta(n^2)$

b) $T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + 1$

$T(n) = 3T\left(\frac{n}{2}\right) + 1$

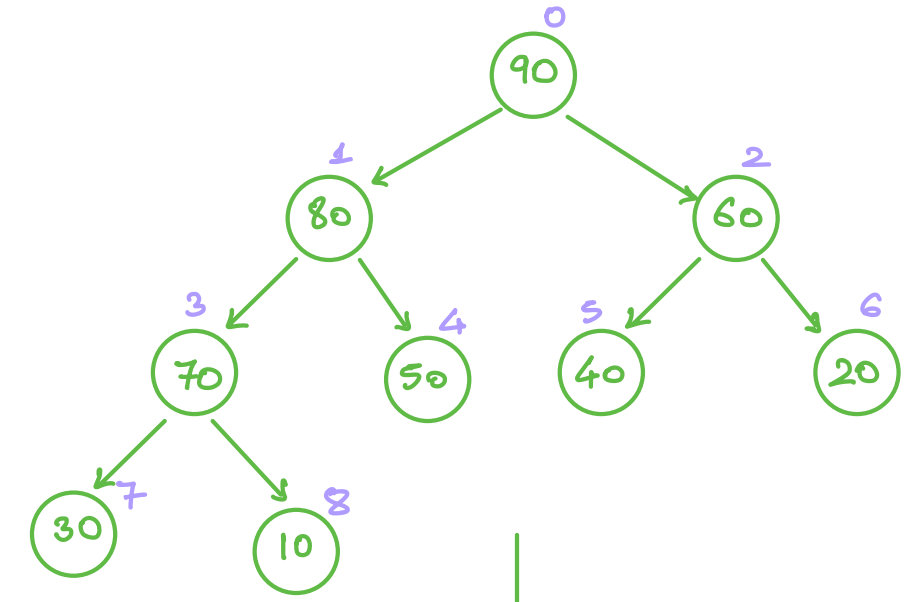
$a = 3 \quad b = 2 \quad k = 0 \quad p = 0$

$a = 3 \quad b^k = 2^0 = 1$

$a > b^k$

$T(n) = \Theta(n^{\log_2 3}) = \Theta(n^{1.58})$

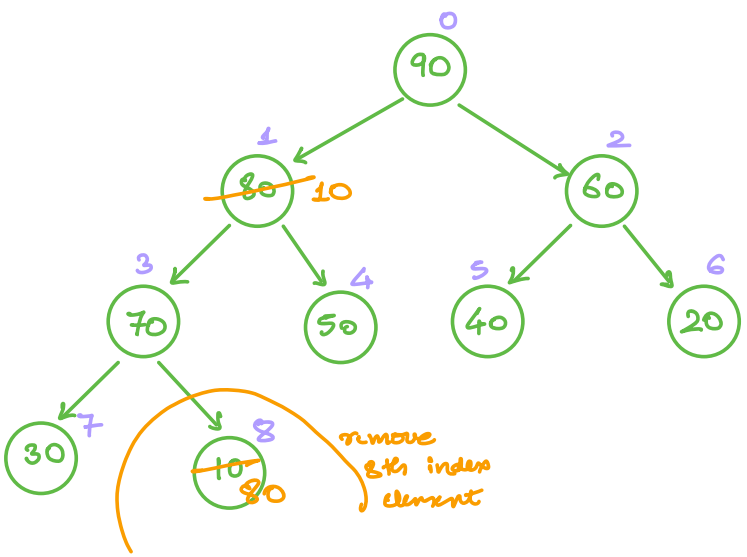
Ques 2:

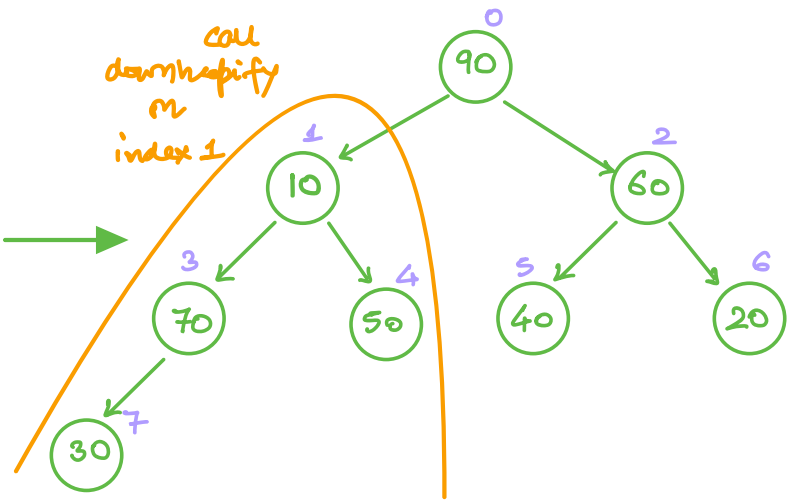


Logic:

If 1st index element has to be removed then:

- Swap 1st and last index element (i.e. 1st and 8th)
- Remove last element
- Call downheapify on index 1






```
void HEAP_DELETE (int *A, int i)
{
    swap (arr[i], arr[N-1]);
    N--;
    DOWNHEAPIFY (A, i);

}

void  DOWNHEAPIFY (int *A, int pi)
{
    int lci = 2*pi+1;
```

```
int lci = 2*pi+2;

int maxi = pi;

if (lci < N && A[lci] > A[maxi])
    maxi = lci;

if (rci < N && A[rci] > A[maxi])
    maxi = rci;

if( maxi != pi)

{
    swap (A[maxi], A[pi]);
    DOWNHEAPIFY (A, maxi);

}

}
```

Ques 3:

Logic 1:
- Count the no. of true and false by applying one loop. Let us say x no. of true and n-x false.
- Put false from index 0 to x-1 and true from index x to n-1

Logic 2: Two Pointer Approach.

similar to partitioning step of quick sort.

- Start one variable from left and one from right.

On left we want false, in case we get a true that is a problem, stop at problem.

On right we want true, in case we get a false that is a problem, stop at problem.

If both left and right are at problem then swap.

```
int left=0 ;
int right= N-1 ;

while (left ≤ right)

{
      while (A[left] == false)
          left++;

      while (A[right] == true)
          right--;

      if (left ≤ right)
      {
          swap (A[left], A[right]);
          left++;
          right--;
      }
}
```

Ques 4:

	I_0	I_1	I_2	I_3	I_4	I_5	I_6
Profit =	10	5	15	7	6	18	3
weight =	2	3	5	7	1	4	1
$\frac{\text{Profit}}{\text{weight}}$ =	5	1.67	3	1	6	4.5	3

Sort in Dec Order →

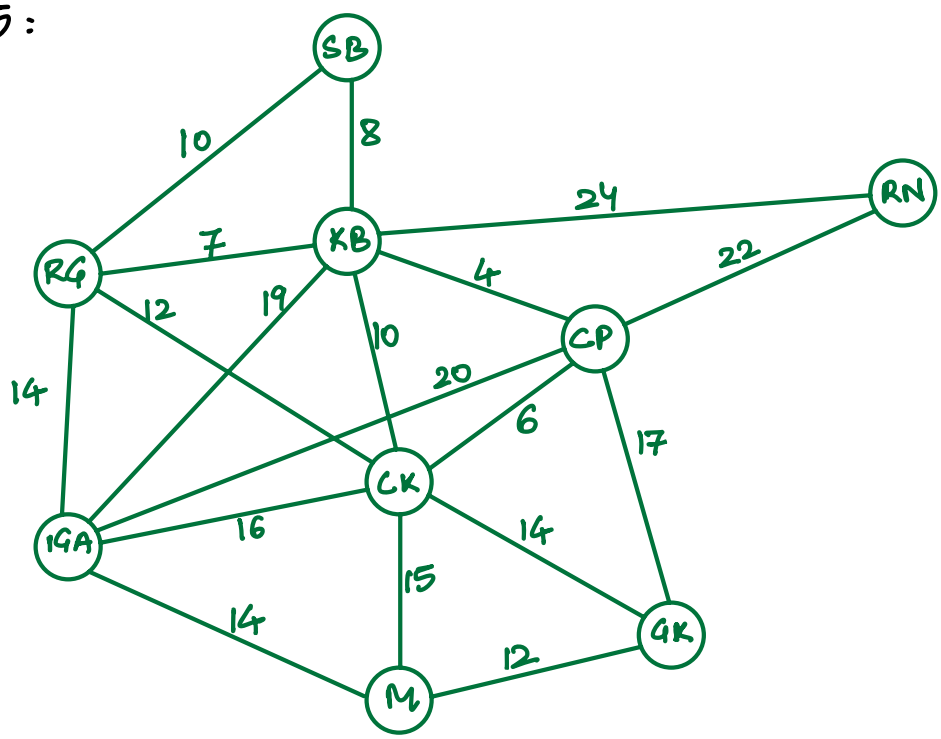
I_4 I_0 I_5 I_2 I_6 I_1 I_3

Item	Remaining Capacity	Profit
–	15	0
I_4	15-1=14	6
I_4 I_0	14-2=12	6+10=16

I_4 I_0 I_5	12-4 = 8	16+18 = 34
I_4 I_0 I_5 I_2	8-5 = 3	34+15 = 49
I_4 I_0 I_5 I_2 I_6	3-1 = 2	49+3 = 52
I_4 I_0 I_5 I_2 I_6 I_1	we can't put entire I_1 fraction = $\frac{\text{Remaining Capacity}}{\text{wt. of } I_1} = \frac{2}{3}$	$52 + \frac{2}{3} \times 5$ ↓ 55.33

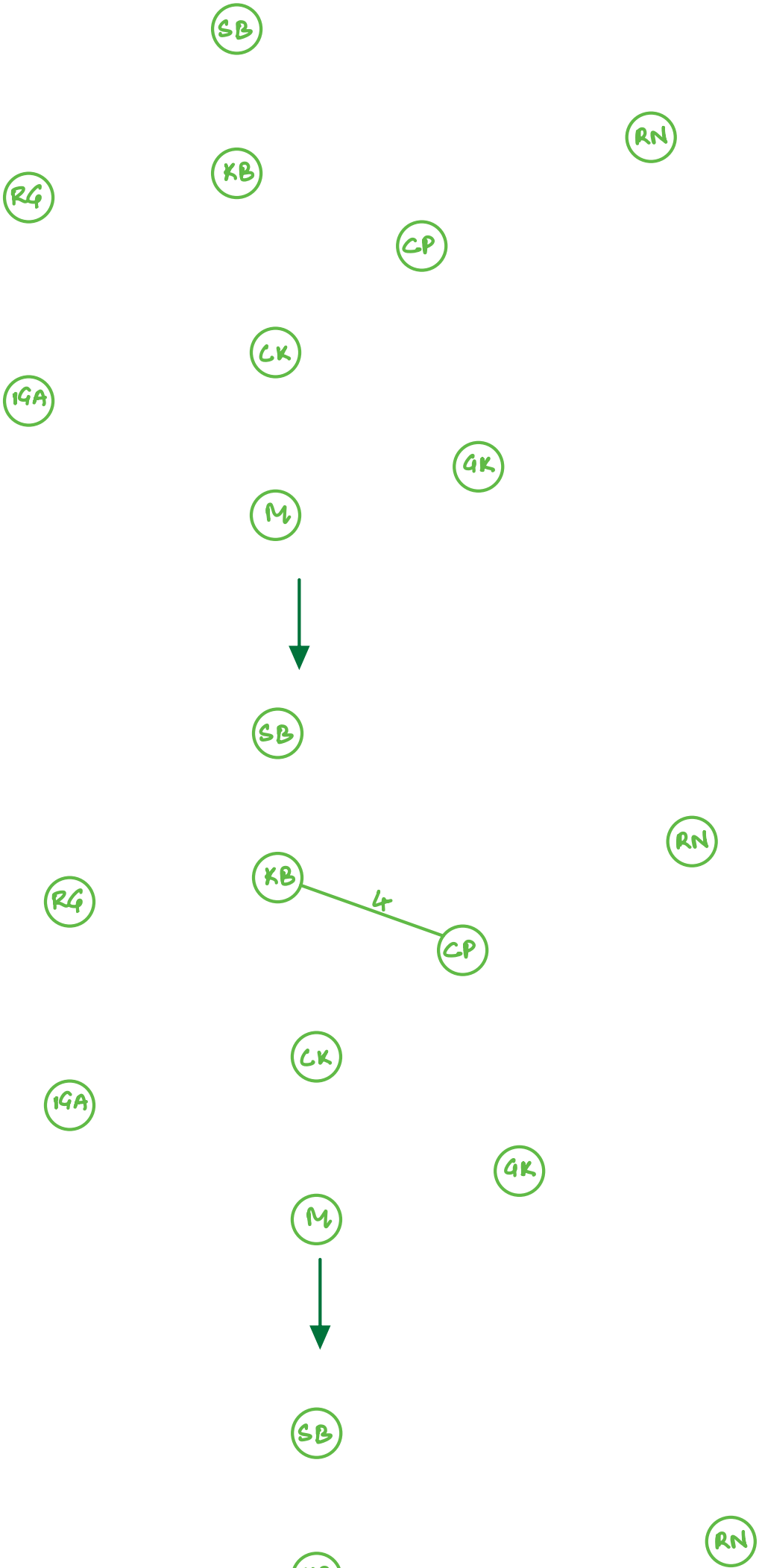
Profit = 55.33

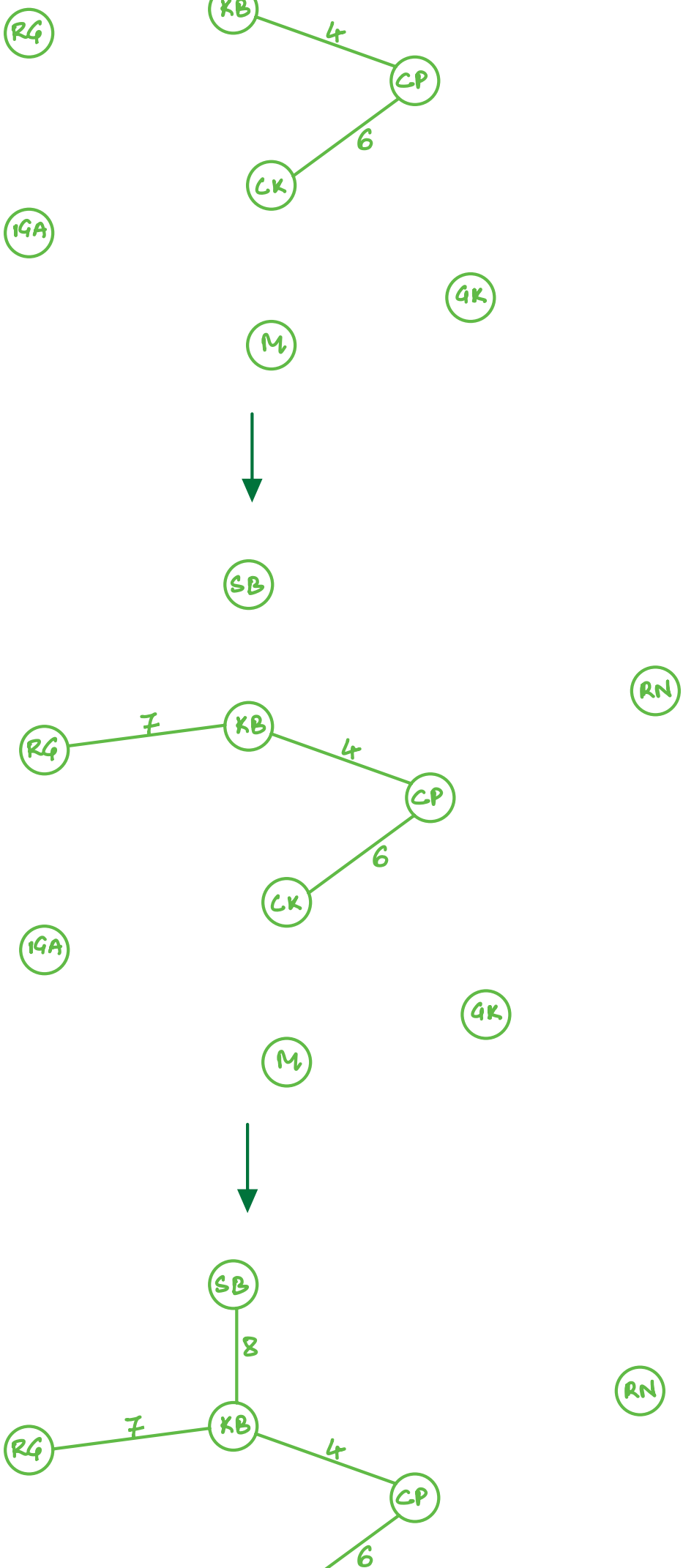
Ques 5:

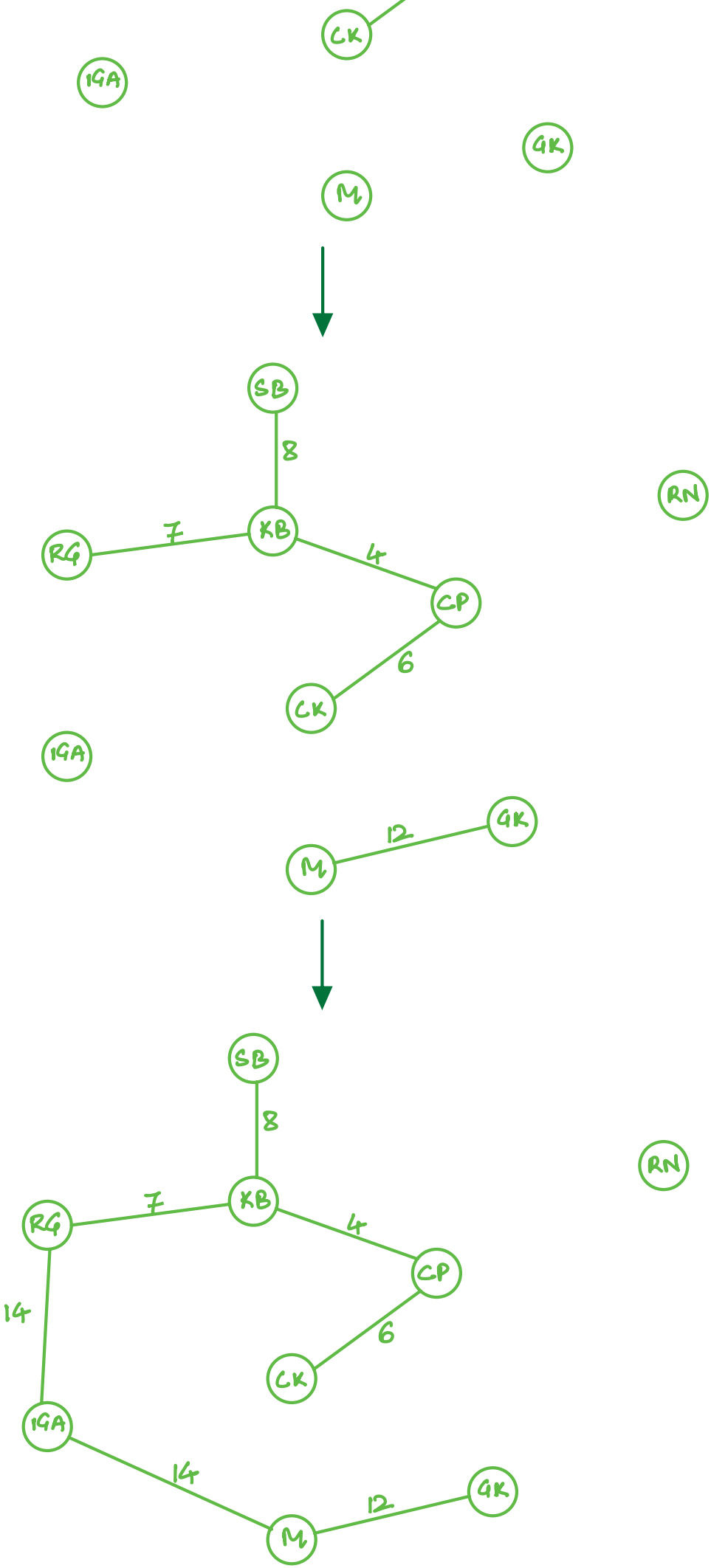


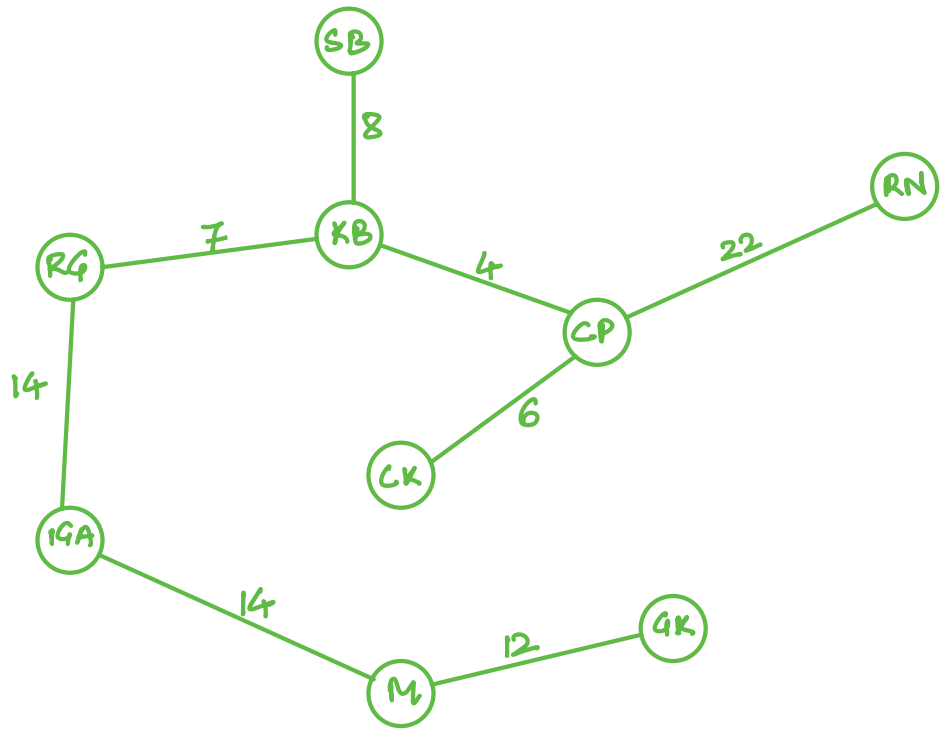
Logic: (Applying Kruskal)

Pick up the edges in increasing order of weight such that no cycle is formed. Since there are 9 vertex we need to select 8 edges.


SB
RN
KB
RG
CP
CK
IGA
GK
M
SB
RN
KB
4
RG
CP
CK
IGA
GK
M
SB
RN


RG
KB
4
CP
6
CK
IGA
GK
M
SB
RN
7
KB
4
RG
CP
6
CK
IGA
GK
M
SB
8
RN
7
KB
4
RG
CP
6


CK
IGA
GK
M
SB
8
RN
7
KB
RG
4
CP
6
CK
IGA
12
GK
M
SB
8
RN
7
KB
RG
4
CP
14
6
CK
IGA
14
12
GK
M



MST weight : 4 + 6 + 7 + 8 + 12 + 14 + 14 + 22 = 87

Other MSTs possible: