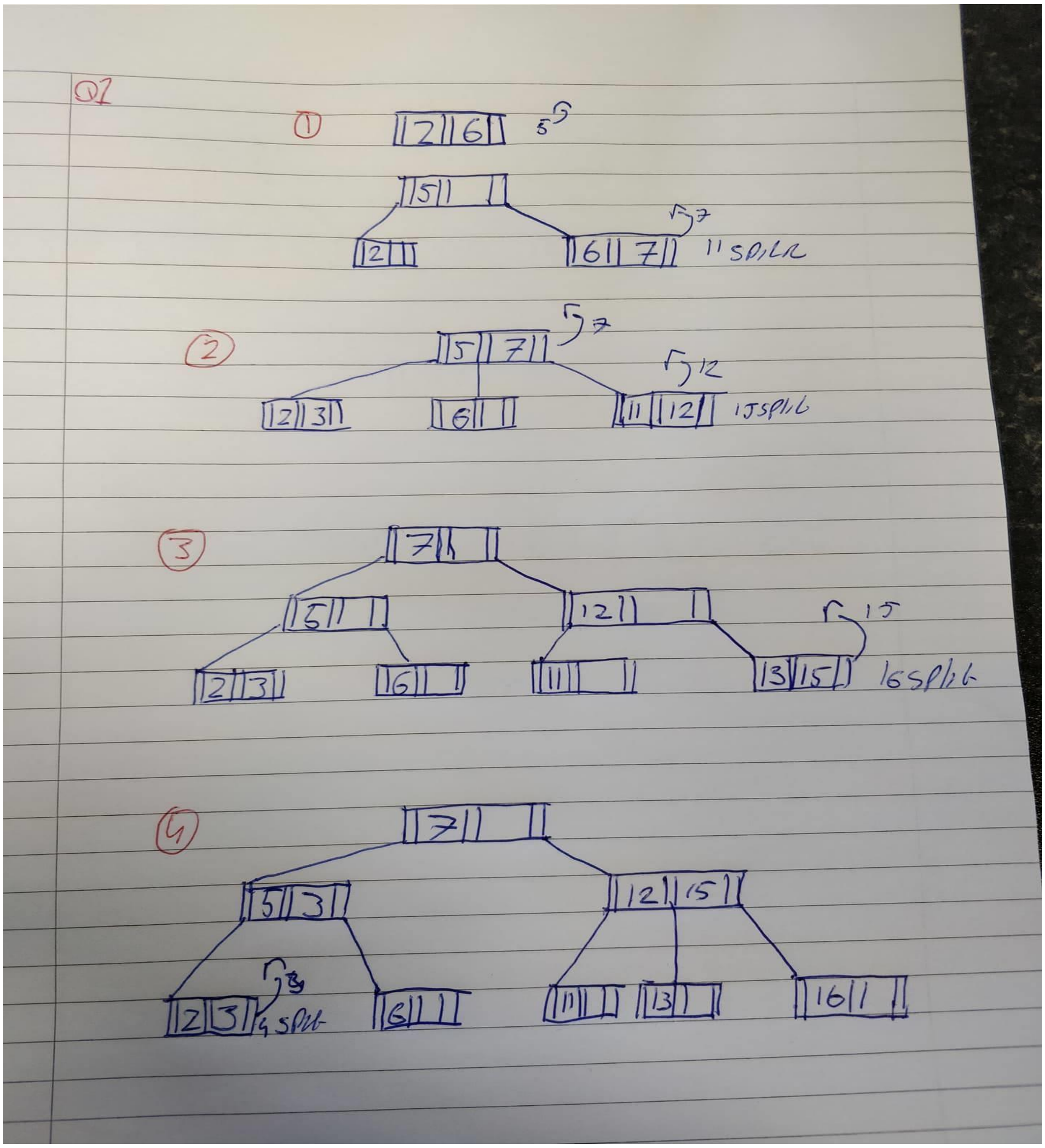
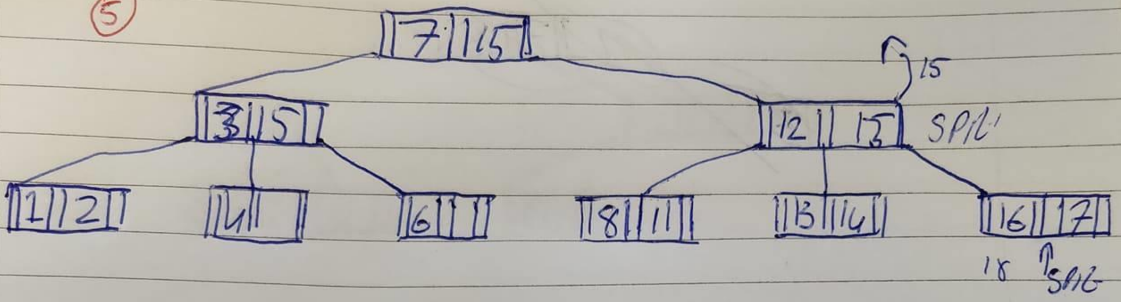
Lab 3 B-Tree

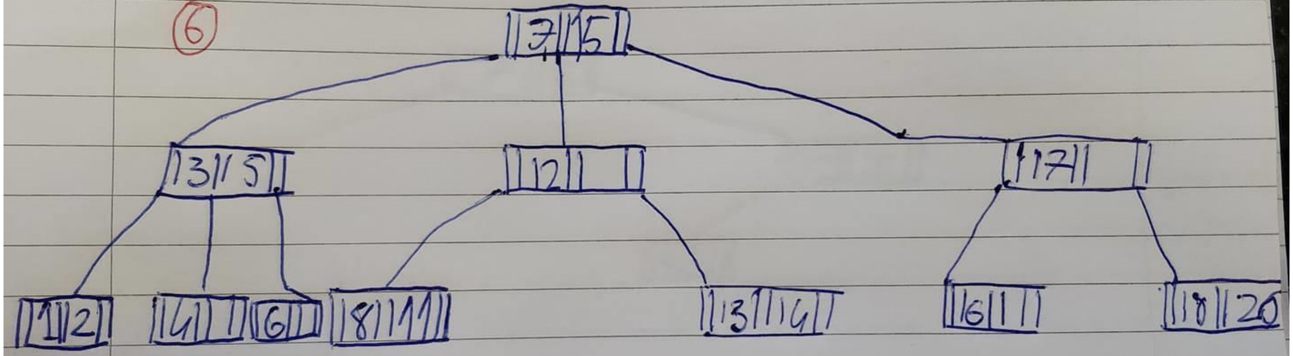
1.


Q1
①
2 6
5
5
2
6 7
7
11 split
②
5 7
7
2 3
6
11 12
12
15 split
③
7
5
12
2 3
6
11
13 15
15
16 split
④
7
5 3
12 15
2 3
3
4 split
6
11
13
16

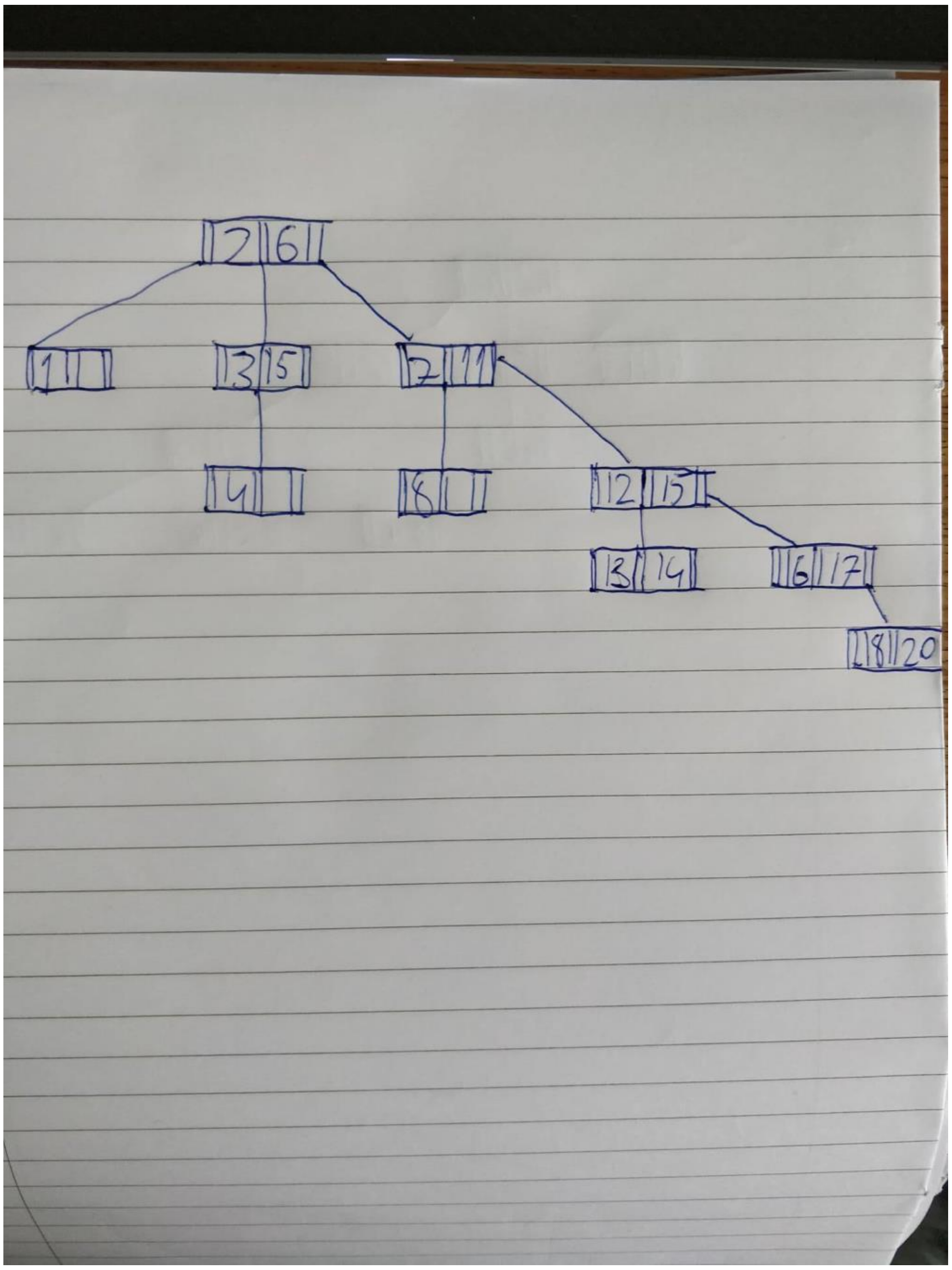
5


7 | 15
3 | 5
12 | 15
15
SPLIT
1 | 2
4
6
8 | 11
13 | 14
16 | 17
18
SPLIT

6


7 | 15
3 | 5
12
17
1 | 2
4
6
8 | 11
13 | 14
16
18 | 20

2.



The difference in these two trees is that one is balanced and the other is unbalanced. The second tree (simple) skews to the right as you can see above. This is because there are no rules to sort the numbers as they are inserted. It is beginning to become a list. It will be a lot more time consuming to have it this way as it is not shorted. It will have to search the hole tree as a result. However, if it was a B-Tree with rules it is a lot less resourceful to find the number you are looking for because it can narrow down the results. A B-tree is a self-balancing tree data structure that keeps data sorted and allows searches, sequential access and insertions. This is a much more efficient way to store and sort data.

3. What is a bitmap index:
   Bitmap index is a two-dimensional structure whereby one column for every row in the table is indexed.

   How does it work:
   The two-dimensional array uses each value within the index multiplied by the number of rows in the table. When retrieving the rows, Oracle decomposes the bitmap into the RAM data buffers and it is scanned for the matching values. The matching values are generated in the form of a ROW-ID list

   What are the advantages and disadvantages:
   The advantages of bitmap index are that each individual column may have a low cardinality when one table includes multiple bitmapped indexes. It also provides a rapid response to difficult SQL queries.

   How big (in Mbytes) is a Bitmap index for a field "Month" (12 values) of a table of 5 million records?
   5,000,000 x 12 = 60,000,000 bits
   60,000,000 bits = 7.5MB

4. Average number of nodes for b-tree will visit is:
   1:3
   2:3
   3:2
   4:3
   5:2
   6:3
   7:1
   8:3
   9:3
   10:3
   11:3
   12:2
   13:3
   14:3
   15:1
   16:3
   17:2
   18:3
   19:3
   20:3
   (3+3+2+3+2+3+1+3+3+3+3+2+3+3+1+3+2+3+3+3 = 52)
   52/20 = 2.6

Average number of nodes for Simple-tree will visit is:
1:2
2:1
3:2
4:3
5:2
6:1
7:2
8:3
9:3
10:3
11:2
12:3
13:4
14:4
15:3
16:4
17:4
18:5
19:5
20:5
(2+1+2+3+2+1+2+3+3+3+2+3+4+4+3+4+4+5+5+5 = 61)
61/20 = 3.05

2.6 / 3.05 = 0.852459 * 100 = 85.24
100 – 85.24 = 14.76
This shows there is an 14.76% gain in performance using the B-tree.