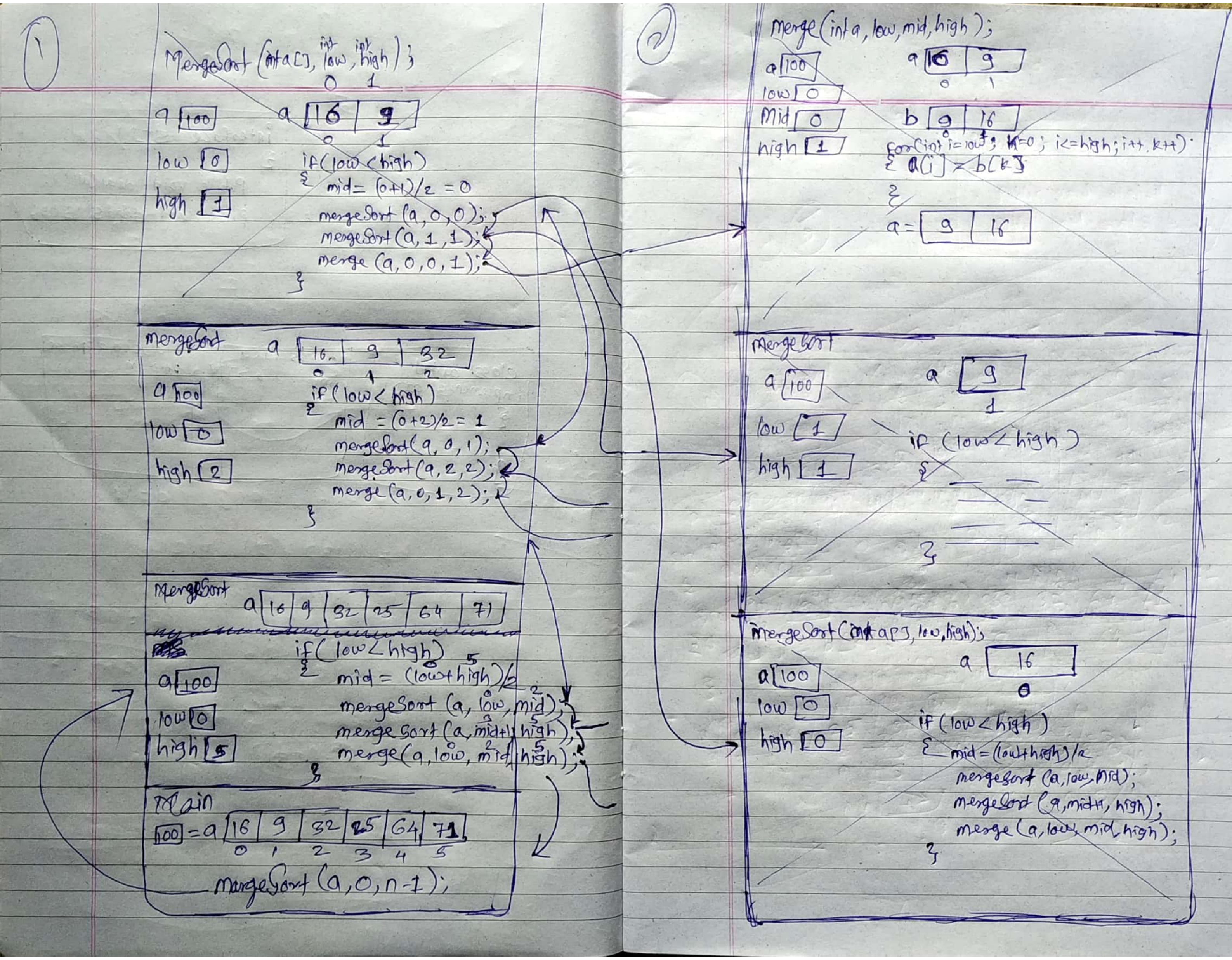


①

```
MergeSort (int a[], int low, int high) ;
                          0        1
a [100]          a [ 16 | 9 ]
                     0    1
low [0]          if(low < high)
                 {  mid = (0+1)/2 = 0
high [1]            mergeSort (a, 0, 0);
                    mergeSort (a, 1, 1);
                    merge (a, 0, 0, 1);
                 }
```

```
mergeSort        a [ 16 | 9 | 32 ]
                     0    1   2
a [100]          if (low < high)
                 {
low [0]             mid = (0+2)/2 = 1
                    mergeSort(a, 0, 1);
high [2]            mergeSort(a, 2, 2);
                    merge (a, 0, 1, 2);
                 }
```

```
MergeSort     a [ 16 | 9 | 32 | 25 | 64 | 71 ]

                 if( low < high)
                 {                0     5
a [100]             mid = (low+high)/2
                                   0    2
                    mergeSort (a, low, mid);
low [0]                            3     5
                    merge sort (a, mid+1, high);
high [5]                       0    2     5
                    merge(a, low, mid, high);
                 }
```

```
Main
[100] = a [ 16 | 9 | 32 | 25 | 64 | 71 ]
            0    1   2    3    4    5

     mergeSort (a, 0, n-1);
```

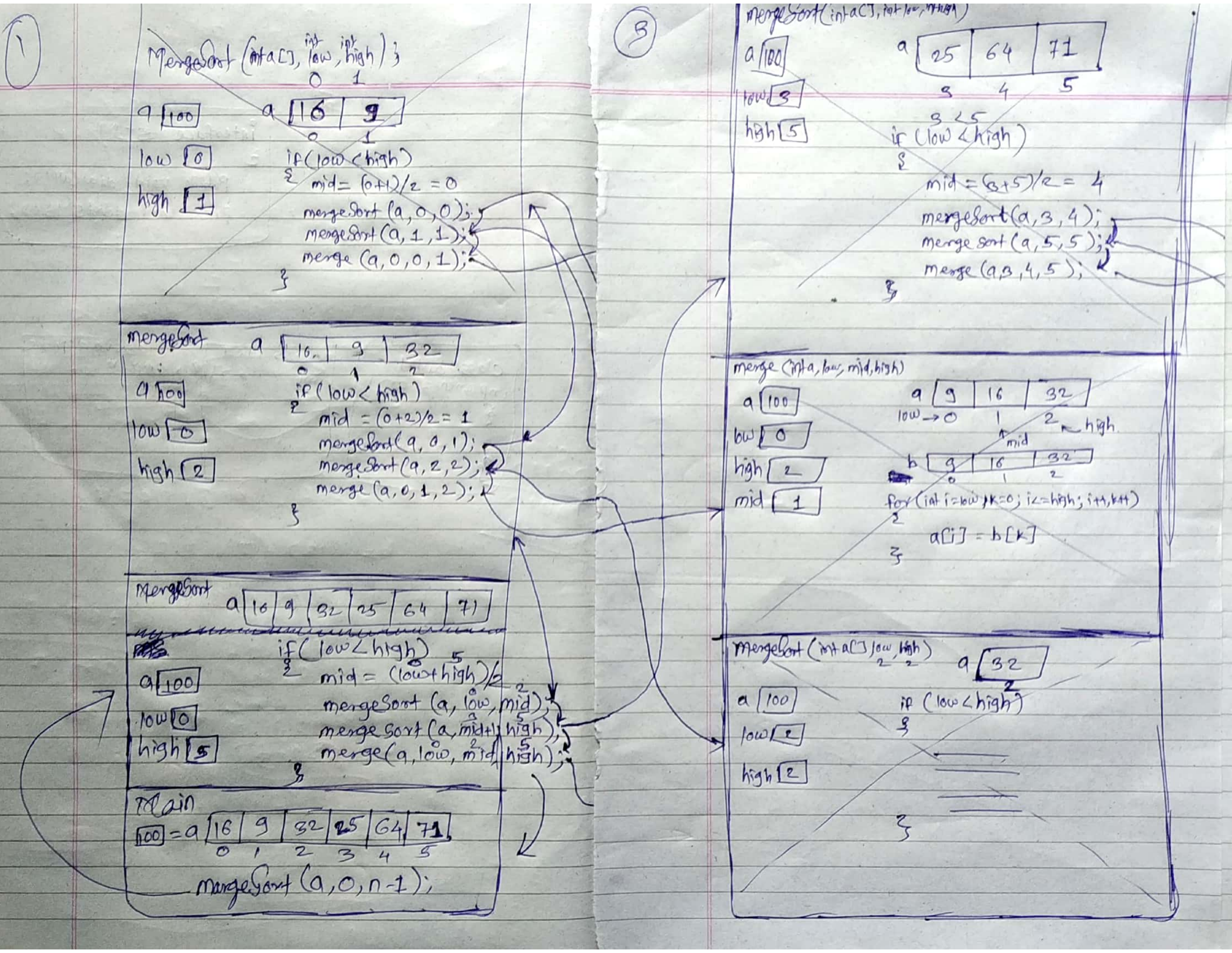
②

```
merge(int a, low, mid, high );
a [100]          a [ 16 | 9 ]
                      0   1
low [0]
Mid [0]          b [ 9 | 16 ]
                     0    1
high [1]         for(int i=low; k=0; i<=high; i++, k++)
                 { a[i] = b[k]
                 }
                 a = [ 9 | 16 ]
```

```
MergeSort
a [100]          a [ 9 ]
                     1
low [1]          if (low < high)
high [1]         {
                 }
```

```
mergeSort(int a[], low, high);
                 a [ 16 ]
a [100]              0
low [0]
high [0]         if ( low < high )
                 { mid = (low+high)/2
                   mergeSort (a, low, mid);
                   mergeSort (a, mid+1, high);
                   merge (a, low, mid, high);
                 }
```

① ③



```
MergeSort (int a[], int low, int high) {
                        0     1
a [100]        a [16 | 9 ]
                   0   1
low [0]        if(low<high)
               {  mid= (0+1)/2 =0
high [1]          mergeSort (a,0,0);
                  mergeSort (a,1,1);
                  merge (a,0,0,1);
               }
```

```
mergeSort      a [16. | 9 | 32]
                  0     1    2
a [100]        if (low< high)
               {  mid = (0+2)/2 = 1
low [0]           mergeSort(a,0,1);
                  mergeSort(a,2,2);
high [2]          merge (a,0,1,2);
               }
```

```
MergeSort   a [16 | 9 | 32 | 25 | 64 | 71]

               if( low<high)
a [100]        {  mid = (low+high)/2
                  mergeSort (a, low, mid);
low [0]           merge sort (a, mid+1, high);
                  merge(a, low, mid, high);
high [5]       }
```

```
Main
[100] = a [16 | 9 | 32 | 25 | 64 | 71]
           0    1   2    3    4    5

mergeSort (a,0,n-1);
```

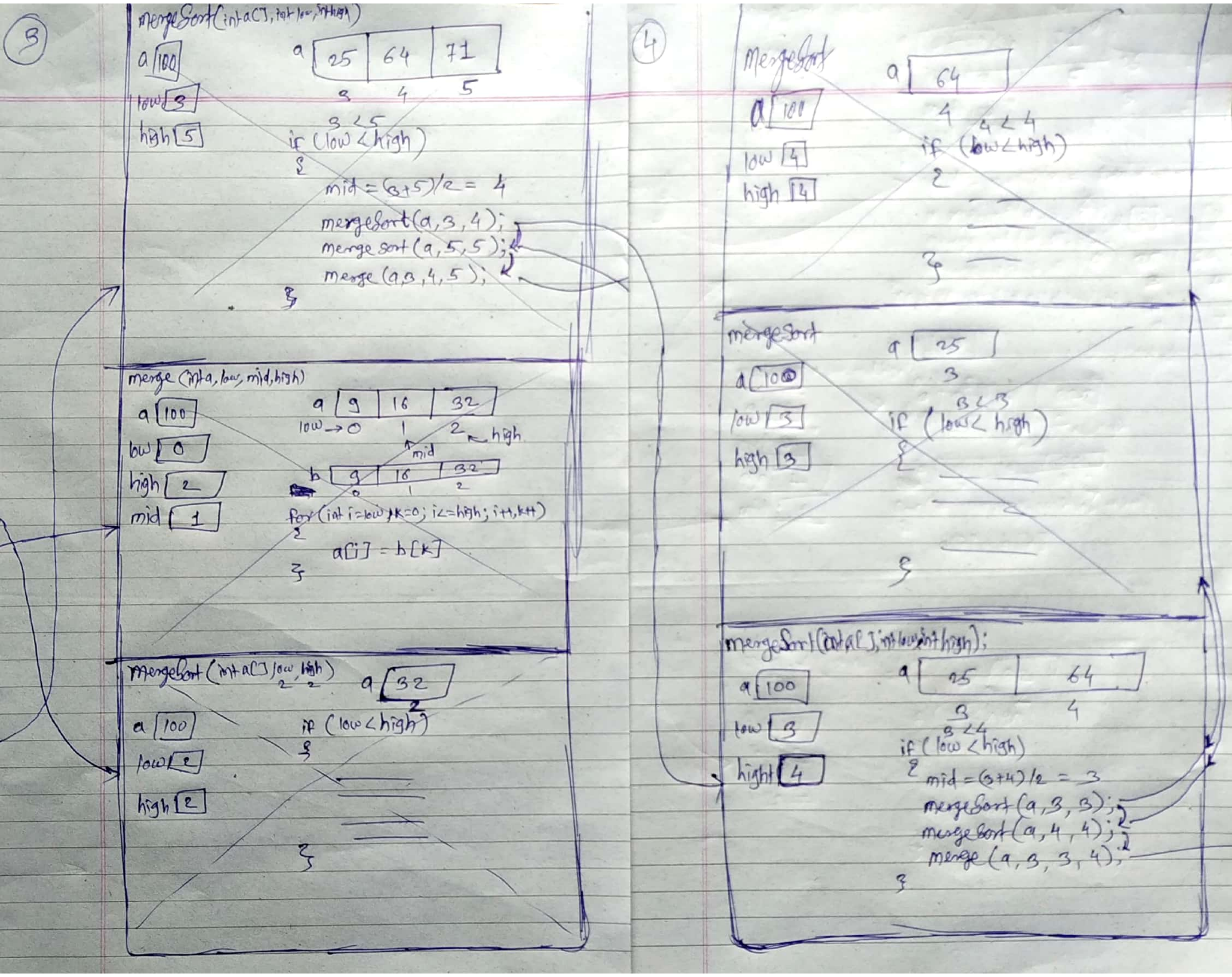
```
mergeSort(int a[], int low, int high)
a [100]        a [25 | 64 | 71]
                  3    4    5
low [3]
                  3 <5
high [5]       if (low < high)
               {
                  mid = (3+5)/2 = 4
                  mergeSort(a,3,4);
                  merge sort (a,5,5);
                  merge (a,3,4,5);
               }
```

```
merge (int a, low, mid, high)
a [100]        a [9 | 16 | 32]
               low→0   1    2 ← high
low [0]                ↑mid
               b [9 | 16 | 32]
high [2]          0   1    2
               for (int i=low, k=0; i<=high; i++, k++)
mid [1]        {
                  a[i] = b[k]
               }
```

```
MergeSort (int a[] low, high)
                    2    2
                            a [32]
a [100]                         2
               if (low<high)
low [2]        {

high [2]

               }
```



③

```
mergeSort(int a[], int low, int high)
```

a 100

low 3

high 5

a	25	64	71
	3	4	5

```
   3 < 5
if (low < high)
{
    mid = (3+5)/2 = 4
    mergeSort(a,3,4);
    merge sort(a,5,5);
    merge(a,3,4,5);
}
```

```
merge(int a, low, mid, high)
```

a 100

low 0

high 2

mid 1

a	9	16	32
low →	0	1 ↑ mid	2 ← high

b	9	16	32
	0	1	2

```
for (int i=low, k=0; i<=high; i++,k++)
{
    a[i] = b[k]
}
```

```
mergeSort(int a[] low, high)
                   2     2
```

a 100

low 2

high 2

a	32
	2

```
if (low<high)
{

}
```

④

```
mergeSort
```

a 100

low 4

high 4

a	64
	4

```
    4 < 4
if (low<high)
{

}
```

```
mergeSort
```

a 100

low 3

high 3

a	25
	3

```
     3 < 3
if (low< high)
{

}
```

```
mergeSort(int a[], int low, int high);
```

a 100

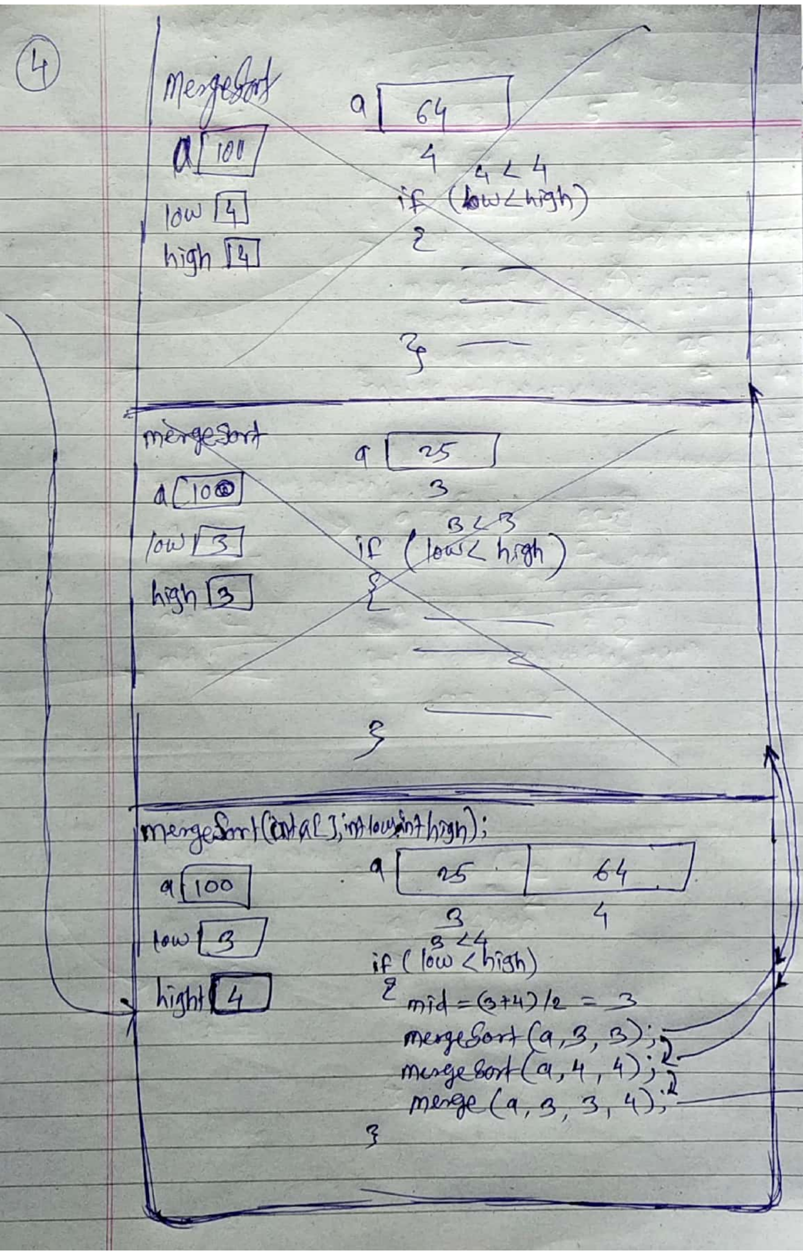
low 3

hight 4

a	25	64
	3	4

```
    3 < 4
if (low < high)
{
  mid = (3+4)/2 = 3
  mergeSort(a,3,3);
  mergeSort(a,4,4);
  merge(a,3,3,4);
}
```

④

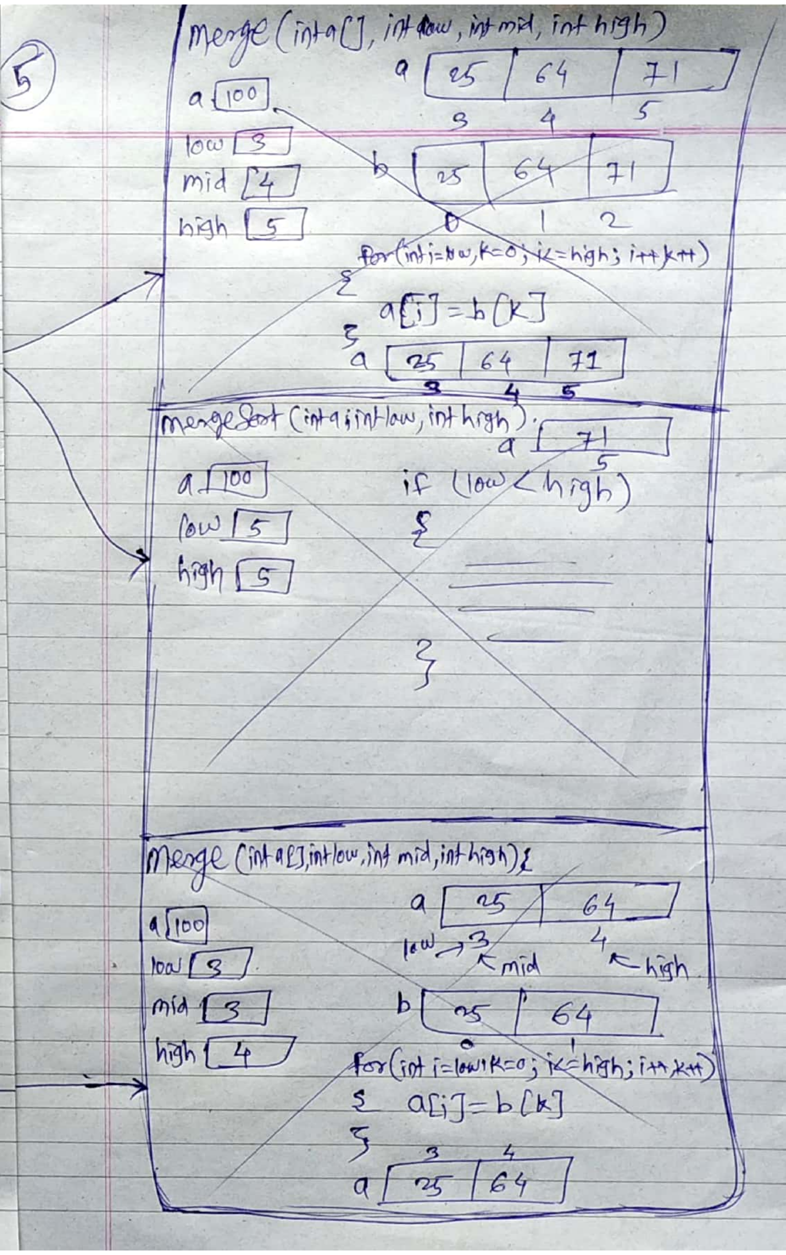


```
mergeSort
a [100]              a [ 64 ]
                         4
low [4]                    4 < 4
                     if (low<high)
high [4]             {
                       ___
                       ___
                     }  ___
```

```
mergeSort
                     a [ 25 ]
a [100]                  3
                           3 < 3
low [3]              if (low< high)
                     {
high [3]               ___
                       ___
                       ___
                     }
```

```
mergeSort(int a[], int low, int high);
a [100]              a [ 25 | 64 ]
                          3    4
low [3]                  3 < 4
                     if ( low < high)
hight [4]            {
                        mid = (3+4)/2 = 3
                        mergeSort (a, 3, 3); 2
                        mergeSort (a, 4, 4); 2
                        merge (a, 3, 3, 4); 2
                     }
```

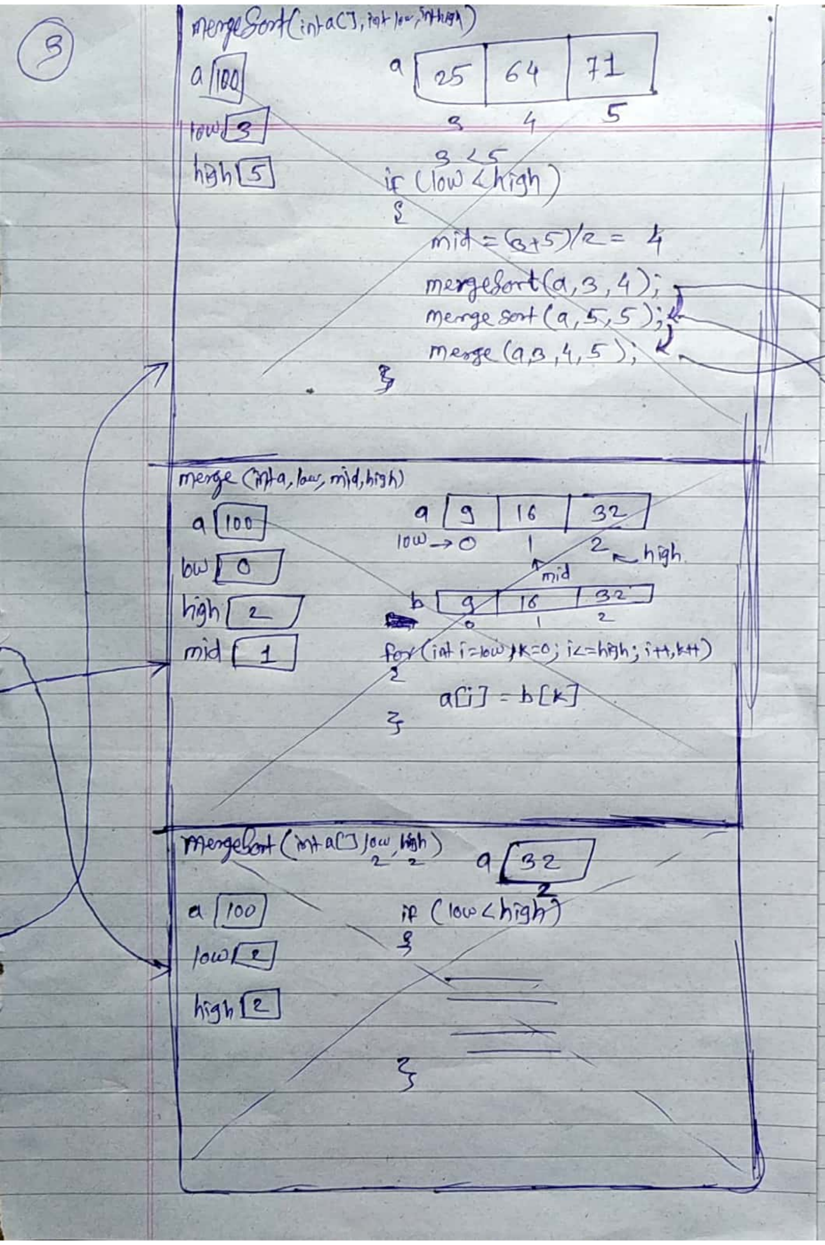
⑤



```
Merge (int a[], int low, int mid, int high)
                     a [ 25 | 64 | 71 ]
a [100]                  3    4    5
low [3]
                     b [ 25 | 64 | 71 ]
mid [4]                  0    1    2
high [5]
       for(int i=low, k=0; k<=high; i++ k++)
       {
          a[i] = b[k]
       }
       a [ 25 | 64 | 71 ]
           3    4    5
```

```
mergeSort (int a, int low, int high).
                          a [ 71 ]
                                5
a [100]              if (low < high)
low [5]              {
high [5]               ___
                       ___
                       ___
                       ___
                     }
```

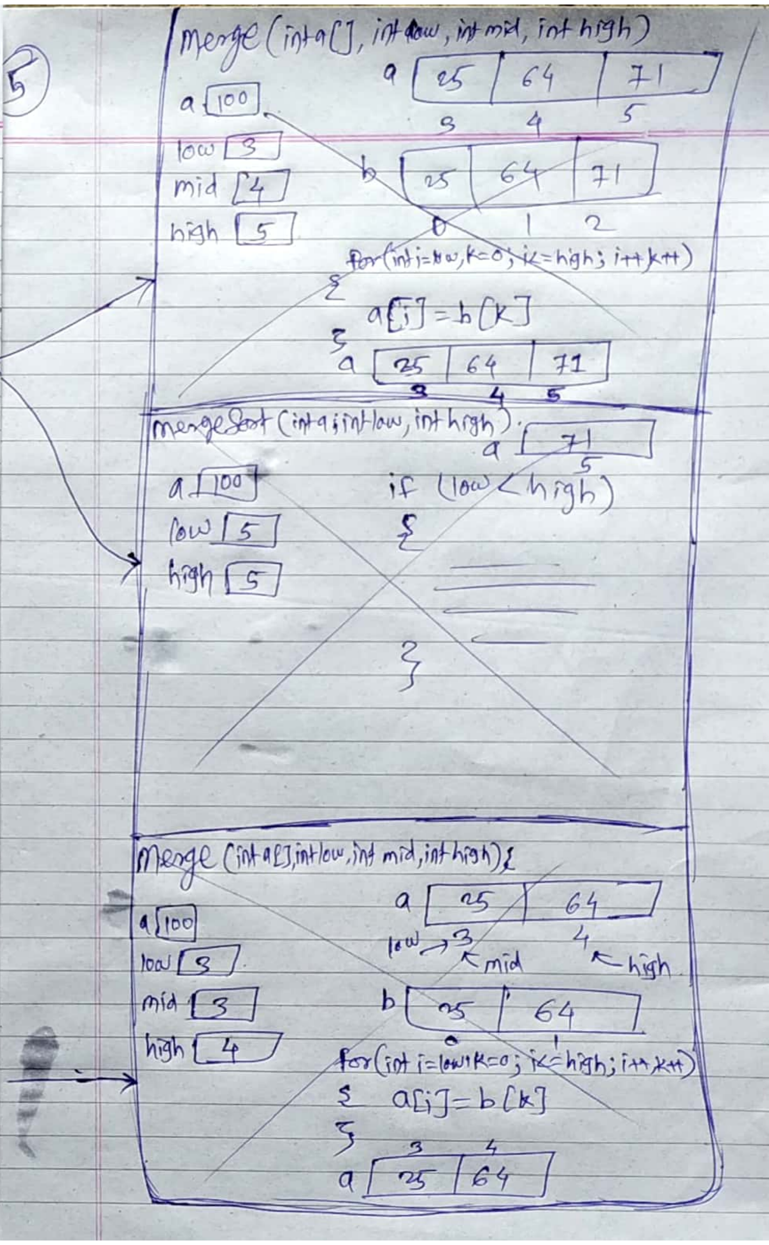
```
Merge (int a[], int low, int mid, int high){
                     a [ 25 | 64 ]
a [100]          low → 3    4 ← high
                       ↖ mid
low [3]
                     b [ 25 | 64 ]
mid [3]                  0    1
high [4]
       for(int i=low, k=0; k<=high; i++, k++)
       {  a[i] = b[k]
       }
           3    4
       a [ 25 | 64 ]
```


3
a 100
low 3
high 5
a 25 64 71
3 4 5
3 < 5
a 100
low 0
high 2
mid 1
a 9 16 32
low → 0 1 2 ← high
↑ mid
b 9 16 32
0 1 2
a 32
2
a 100
low 2
high 2

```
mergeSort(int a[], int low, int high)
if (low < high)
{
    mid = (3+5)/2 = 4
    mergeSort(a,3,4);
    merge sort(a,5,5);
    merge(a,3,4,5);
}
```

```
merge(int a, low, mid, high)
for(int i=low,k=0; i<=high; i++,k++)
{
    a[i] = b[k]
}
```

```
mergeSort(int a[] low, high)
                   2     2
if (low<high)
{
}
```

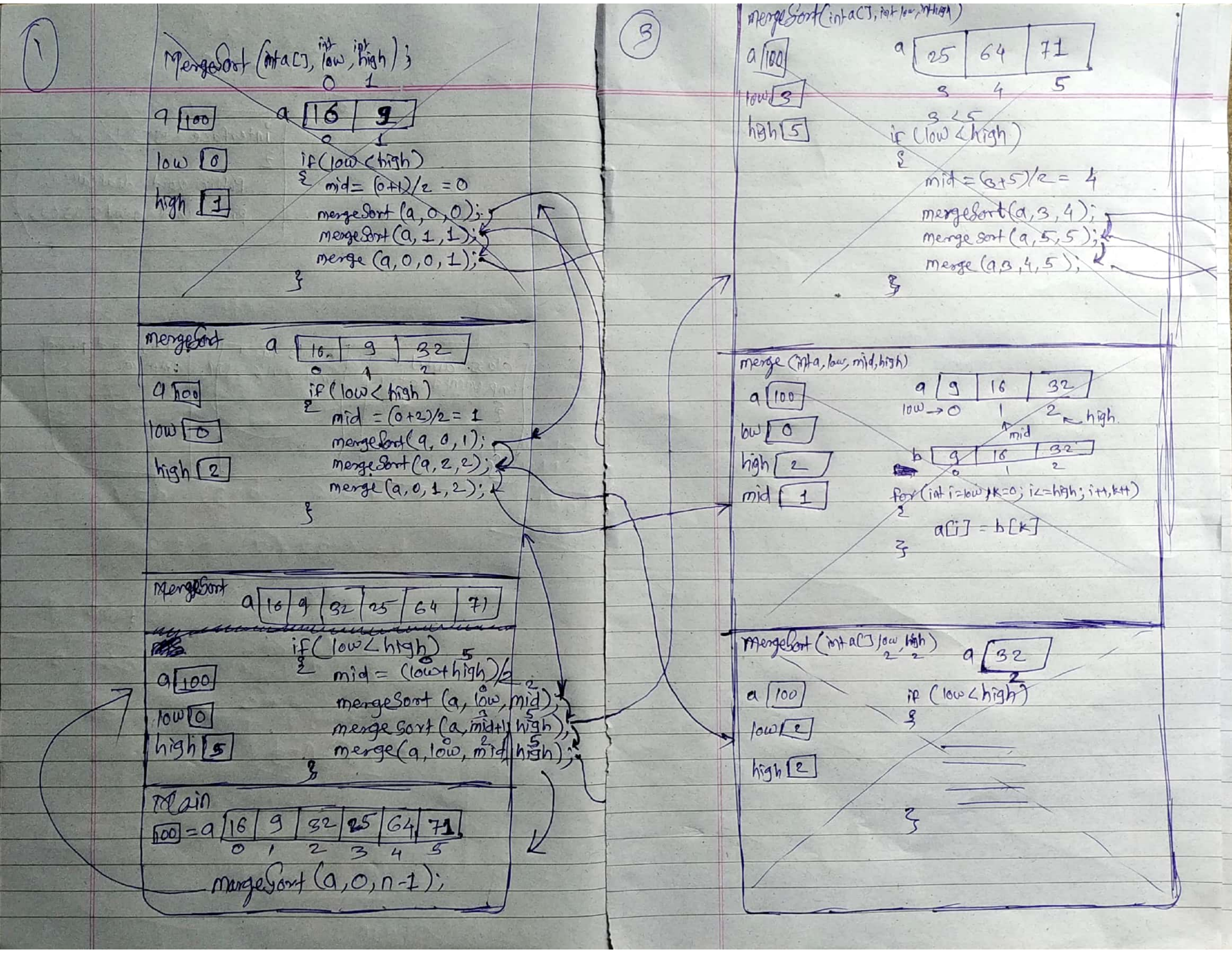

5
a 100
low 3
mid 4
high 5
a 25 64 71
3 4 5
b 25 64 71
0 1 2
a 25 64 71
3 4 5
a 71
5
a 100
low 5
high 5
a 25 64
low → 3 ← mid
4 ← high
a 100
low 3
mid 3
high 4
b 25 64
0 1
a 25 64
3 4

```
merge(int a[], int low, int mid, int high)
for(int i=low,k=0; i<=high; i++ k++)
{
    a[i] = b[k]
}
```

```
mergeSort(int a; int low, int high)
if (low < high)
{
}
```

```
merge(int a[], int low, int mid, int high){
for(int i=low,k=0; i<=high; i++,k++)
{   a[i] = b[k]
}
```

Scanned by CamScanner



①

```
MergeSort (int a[], int low, int high) }
                    0        1
a [100]      a [16 | 9]
                0    1
low [0]      if(low<high)
             {  mid= (0+1)/2 =0
high [1]        mergeSort (a,0,0);
                mergeSort(a,1,1);
                merge (a,0,0,1);
             }
```

```
mergeSort    a [16 | 9 | 32]
                0    1    2
a [100]      if(low<high)
             {  mid =(0+2)/2= 1
low [0]         mergeSort(a,0,1);
                mergeSort(a,2,2);
high [2]        merge (a,0,1,2);
             }
```

```
MergeSort   a [16 | 9 | 32 | 25 | 64 | 71]

             if( low<high)
             {  mid = (low+high)/2
a [100]         mergeSort (a, low, mid);
low [0]         merge sort (a, mid+1, high);
high [5]        merge(a, low, mid, high);
             }
```

```
Main
[100] = a [16 | 9 | 32 | 25 | 64 | 71]
           0    1    2    3    4    5

   mergeSort (a,0,n-1);
```

③

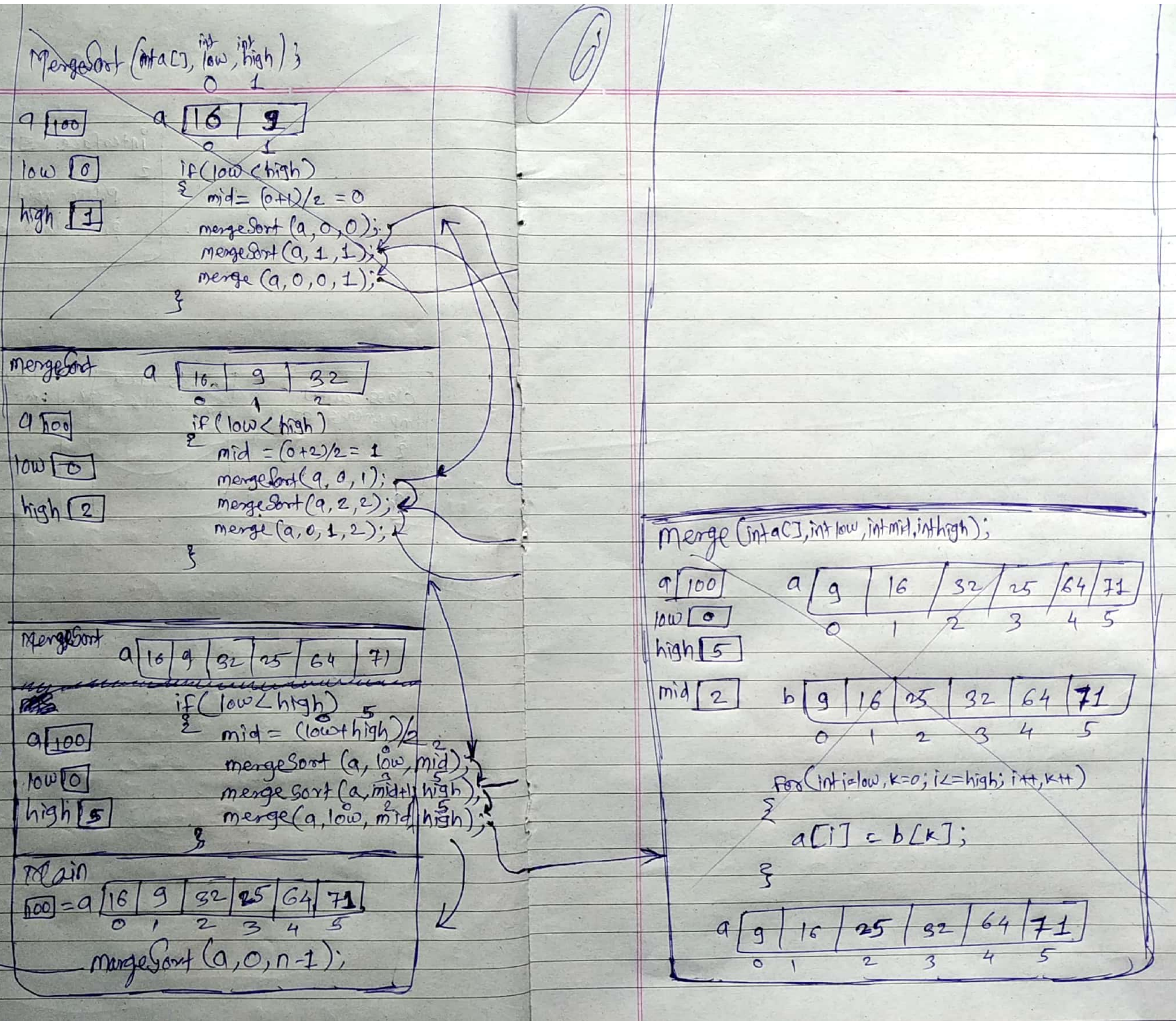
```
mergeSort(int a[], int low, int high)
a [100]          a [25 | 64 | 71]
                    3    4    5
low [3]
                    3 < 5
high [5]         if (low<high)
                 {
                     mid = (3+5)/2 = 4
                     mergeSort(a,3,4);
                     merge sort (a,5,5);
                     merge (a,3,4,5);
                 }
```

```
merge (int a, low, mid, high)
a [100]          a [9 | 16 | 32]
                low → 0   1   2 ← high
low [0]                   ↑
                         mid
high [2]         b [9 | 16 | 32]
                    0   1    2
mid [1]          for (int i=low, k=0; i<=high; i++, k++)
                 {
                     a[i] = b[k]
                 }
```

```
MergeSort (int a[] low, high)       a [32]
                 2     2               2
a [100]          if (low<high)
                 {
low [2]

high [2]

                 }
```



```
MergeSort (int a[], int low, int high) }
                         0         1
a [100]      a [16 | 9]
                 0    1
low [0]      if(low < high)
             {
high [1]       mid= (0+1)/2 = 0
               mergeSort (a,0,0);
               mergeSort(a,1,1);
               merge (a,0,0,1);
             }
```

```
mergeSort    a [16. | 9 | 32]
                 0    1    2
a [100]      if (low< high)
             {
low [0]        mid = (0+2)/2= 1
               mergeSort(a,0,1);
high [2]       mergeSort(a,2,2);
               merge (a,0,1,2);
             }
```

```
MergeSort    a [16 | 9 | 32 | 25 | 64 | 71]

             if( low< high)
             {     0    5
a [100]        mid= (low+high)/2
                             0    2
low [0]        mergeSort (a, low, mid);
                             3      5
high [5]       merge sort (a, mid+1, high);
                          0    2    5
               merge(a, low, mid, high);
             }
```

```
Main
[100] = a [16 | 9 | 32 | 25 | 64 | 71]
            0   1    2    3    4    5
   mergeSort (a,0,n-1);
```

```
Merge (int a[], int low, int mid, int high);

a [100]     a [9 | 16 | 32 | 25 | 64 | 71]
low [0]        0    1    2    3    4    5
high [5]

mid [2]     b [9 | 16 | 25 | 32 | 64 | 71]
               0    1    2    3    4    5

   For(int i=low, k=0; i<=high; i++, k++)
   {
      a[i] = b[k];
   }

a [9 | 16 | 25 | 32 | 64 | 71]
   0    1    2    3    4    5
```