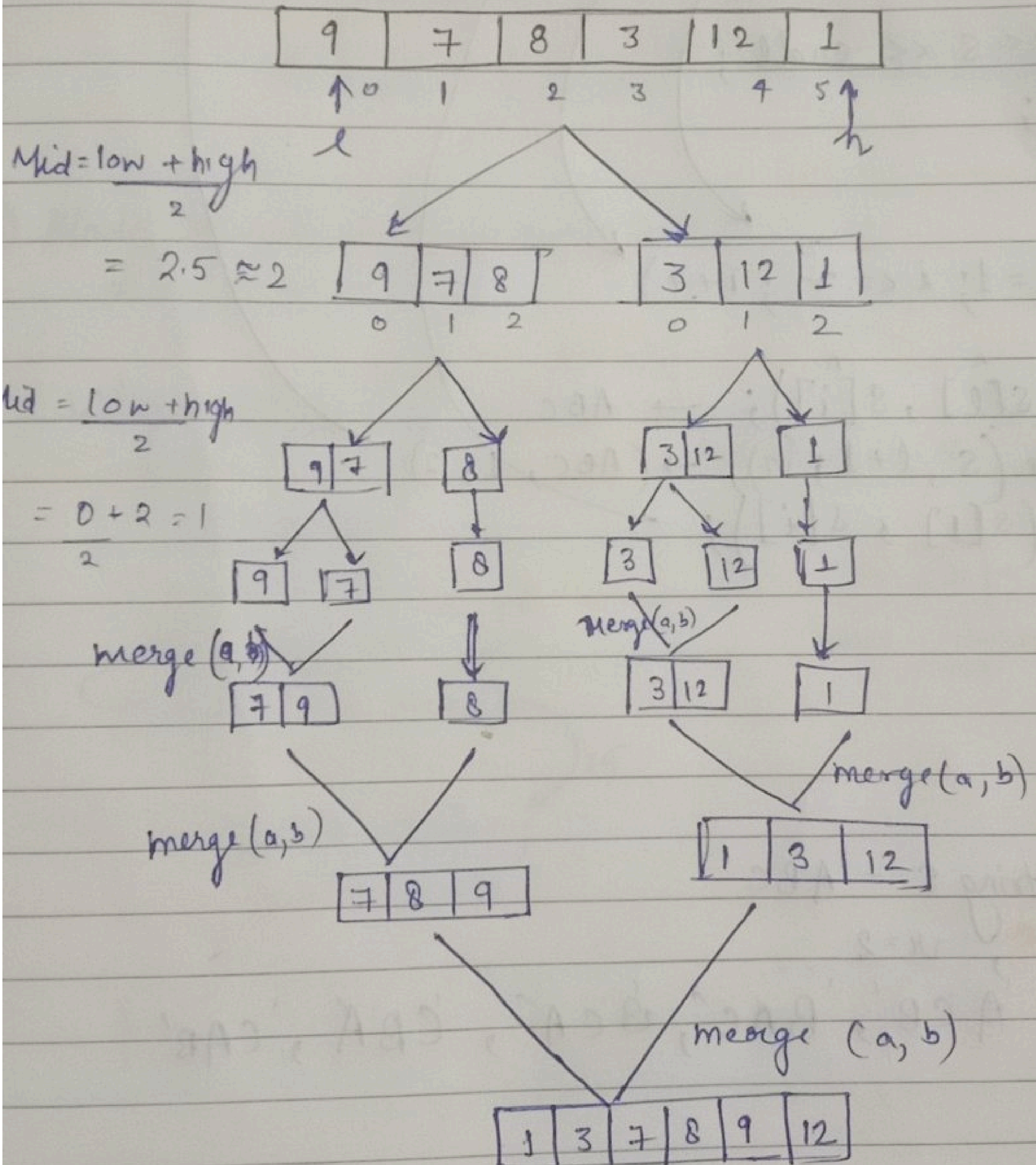
Dry Run

Merge Sort, Permutations, Reverse Words of S
Node structure of Linked List.

① Merge Sort.



② Permutation

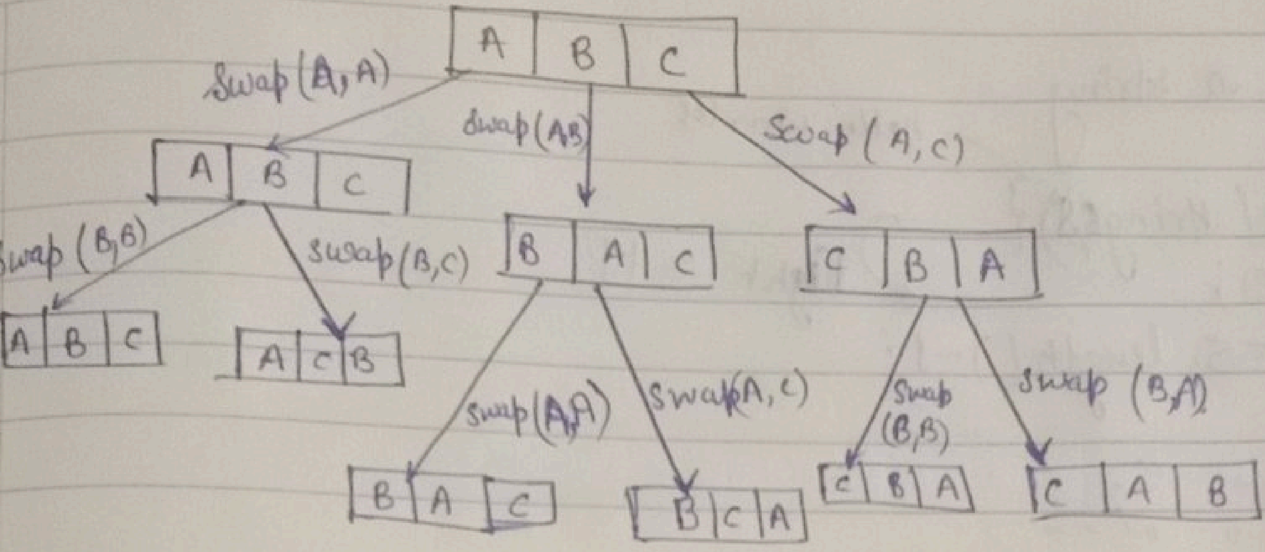
```
                         ABC      0 1        2
void permute (string s, int l, int r)
 {
 if (l==r)
  {
   cout << s << endl;
   return;
  }

 for(int i = 1; i <= r ; i++)
  {
   swap(s[l], s[i]); → ABC
   permute(s, l+1, r); → (ABC, 1, 2)
    swap(s[l], s[i]);
  }
 }
```

① Dry Run

Let String s = 'ABC'

l = 0 , r = 2

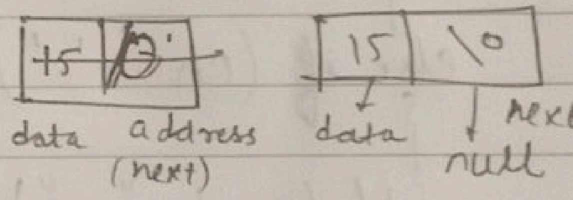
'ABC' , 'ACB', 'BAC', 'BCA' , 'CBA' , 'CAB'



3) Node structure of Linked List.
Dry Run.

```
class node {
 public:
  int data;
  node* next;      15
  //constructor
  node (int Value)
    {                15
      data = value;
      next = NULL;
    }
}
```

let value = 15



④ Reverse a string

Hello world

right = 11

```
String solve( string s){
int left=0;
int right = s.length()-1;

String temp=" ";
string res=" ";
while (left <= right)
{
  Char ch = s[left];
  if (char != " ")
  {
    temp += ch;
  }
  else if (ch == ' ')
  {
    if (res != " ")
    {
      res = temp + " " + res;
    }
    else
      res = temp;
    temp = " ";
  }
  left++;
}
}
```

Hello
World

~~World~~

Hello.

```
if(temp != "  ")
 {
  if(res != "  ")
   {
    res = temp + "  " + res;
   }
  else
  res = temp;
 }
return res;
}
}
```

Dry Run.

S = Hello.World

Output

World Hello.