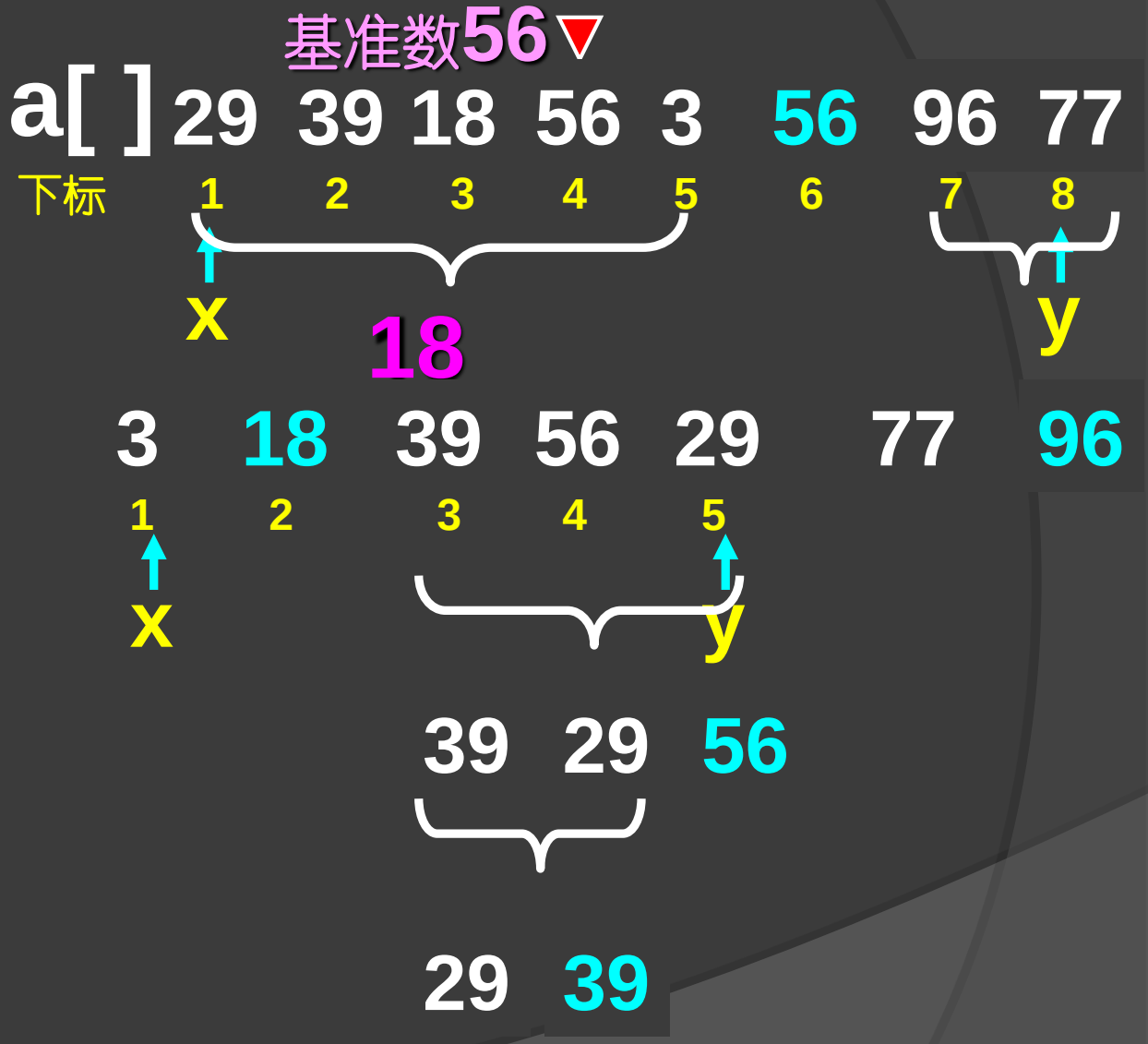
快速排序

quiksort

快速排序 将 29 96 18 56 3 56 39 77 按由小到大排序

思想：每次任找一个数为基准，把小于等于它的放到它的左边，把大于等于它的放到它的右边

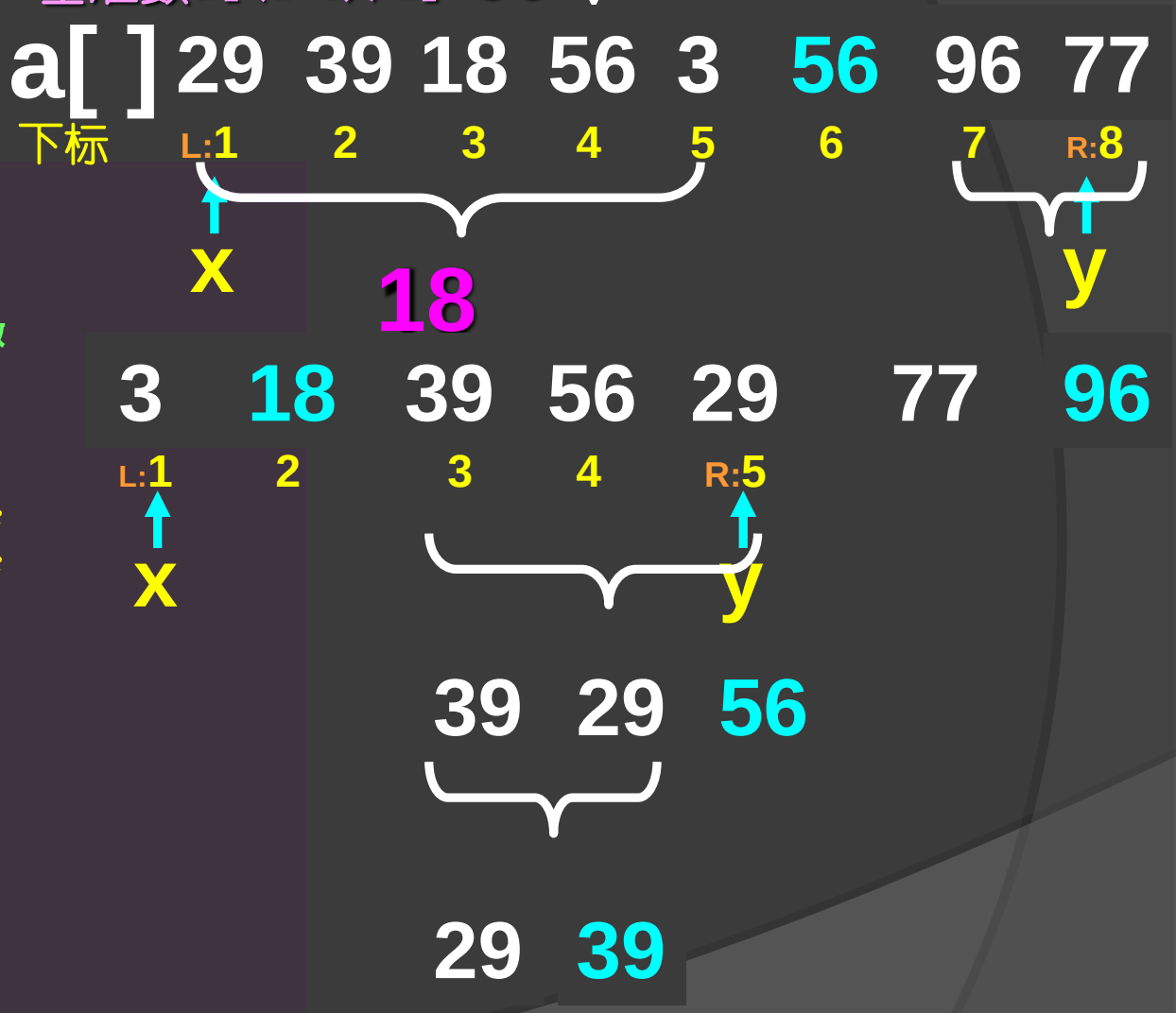


快速排序 将 29 96 18 56 3 56 39 77 按由小到大排序

思想：每次任找一个数为基准，
把小于等于它的放到它的左边，
把大于等于它的放到它的右边

```
void qsort(int L,int R)
{
  int x,y,mid,temp;
  mid=a[(L+R)/2];//选基准数
  x=L;   y=R;
  while(x<=y)
  {
    while(a[x]<mid)x++;
    while(a[y]>mid)y--;
    if (x<=y)
    {
      temp=a[x];
      a[x]=a[y];
      a[y]=temp;
      x++;   y--;
    }
  }
    if(l<y)qsort(l,y);
    if(x<r)qsort(x,r);
}
```

基准数a[(1+8)/2]=56



```
#include <iostream>
using namespace std;
int a[30001];
int i,n;
void qsort(int l,int r){
  int x,y,mid,temp;
  x=l;  y=r;  mid=a[(l+r)/2];
  while(x<=y){
    while(a[x]<mid)x++;
    while(a[y]>mid)y--;
    if (x<=y){
      temp=a[x];  a[x]=a[y];  a[y]=temp;
      x++;  y--;
    }
    if(l<y)qsort(l,y);
    if(x<r)qsort(x,r);
}
int main(){
  cin>>n;
  for(i=1;i<=n;i++)cin>>a[i];
  qsort(1,n);
  for(i=1;i<=n;i++)cout<<a[i]<<" ";
  return 0;
}
```

对a数组的L到R这段区间的数字进行排序

mid为当前这趟排序所选基准值

平均时间复杂度：

$O(n\log_2 n)$

稳定性：不稳定

L到R这段区间被划分成了
L到Y，和X到R两段更小的区间
以递归的方式对这两个区间排序

五种排序算法效率对比：对100000个随机整数进行排序

名称	排名	得分	用时
冒泡排序	5	100	342.82
选择排序	4	100	338.46
插入排序	3	100	75.58
归并排序	2	100	0.77
快速排序	1	100	0.76