10.1

(1)直接插入排序

```
(5,1,6,0,9,2,8,3,7,4)
(1,5,6,0,9,2,8,3,7,4)
(1,5,6,0,9,2,8,3,7,4)
(0,1,5,6,9,2,8,3,7,4)
(0,1,2,5,6,9,8,3,7,4)
(0,1,2,5,6,8,9,3,7,4)
(0,1,2,3,5,6,8,9,7,4)
(0,1,2,3,5,6,7,8,9,4)
(0,1,2,3,4,5,6,7,8,9)
```

(2)希尔排序

```
(5,1,6,0,9,2,8,3,7,4)
增量5排序结果：
(2,1,3,0,4,5,8,6,7,9)
增量3排序结果:
(0,1,3,2,4,5,8,6,7,9)
增量1排序结果:
(0,1,2,3,4,5,6,8,7,9)
最后一趟直接插入排序:
(0,1,2,3,4,5,6,7,8,9)
```

(3)快速排序

```
(5,1,6,0,9,2,8,3,7,4)
(5,1,[6],0,9,2,8,3,7,[4])
(5,1,[4],0,9,2,8,3,7,[6])
(5,1,4,0,[9],2,8,[3],7,6)
(5,1,4,0,[3],2,8,[9],7,6)
(5,1,4,0,3,[2],8,9,7,6)
(2,1,4,0,3,[5],8,9,7,6)
递归
(2,1,[4],[0],3,|5|,8,[9],7,[6])
(2,1,[0],[4],3,|5|,8,[6],7,[9])
(2,1,[0],4,3,|5|,8,6,[7],9)
(0,1,|2|,4,3,|5|,7,6,|8|,9)
(0,1,|2|,3,4,|5|,6,7,|8|,9)
递归
```

(4)冒泡排序

```
(5,1,6,0,9,2,8,3,7,4)
(1,5,0,6,2,8,3,7,4,9)
(1,0,5,2,6,3,7,4,8,9)
(0,1,2,5,3,6,4,7,8,9)
(0,1,2,3,5,4,6,7,8,9)
(0,1,2,3,4,5,6,7,8,9)
(0,1,2,3,4,5,6,7,8,9)
```

(5)归并排序

```
(5,1,6,0,9,2,8,3,7,4)
((1,5),(0,6),(2,9),(3,8),(4,7))
((0,1,5,6),(2,3,8,9),(4,7))
((0,1,2,3,5,6,8,9),(4,7))
(0,1,2,3,4,5,6,7,8,9)
```

(6)堆排序

```
(5,1,6,0,9,2,8,3,7,4)
建立初始堆：
(5)
(5,1)
(6,1,5)
(6,1,5,0)
(9,6,5,0,1)
(9,6,5,0,1,2)
(9,6,8,0,1,2,5)
(9,6,8,3,1,2,5,0)
(9,7,8,6,1,2,5,0,3)
(9,7,8,6,1,2,5,0,3,4)
pop 9
(8,7,5,6,1,2,4,0,3)
pop 8
(7,6,5,3,1,2,4,0)
pop 7
(6,3,5,0,1,2,4)
pop 6
(5,3,4,0,1,2)
pop 5
(4,3,2,0,1)
pop 4
(3,1,2,0)
pop 3
(2,1,0)
pop 2
(1,0)
pop 1 pop 0
```

10.3

(1)大顶堆 (2)(12,24,33,65,24,33,48,92,86,70) 小顶堆

10.5

```
typedef struct lnode
{
    int data;
    struct lnode *link;
} lnode, *linknode;

typedef struct
{
    linknode head, rear;
    int len;
} linklist;

void sort(linklist &l)
{
    int i = 0;
    linknode p, q, ins, min, temp;
    ins = l.head;
    while (ins->link)
    {
        p = ins->link;
        min = p;
        while (p)
        {
            if (min->data > p->data)
            {
                temp = q;
                min = p;
                i = 1;
            }
            q = p;
            p = p->link;
        }
        if (i == 1)
        {
            i = 0;
            temp->link = min->link;
            min->link = ins->link;
            ins->link = min;
        }
        ins = min;
    }
}
```