排序算法（C语言实现）

插入排序

```
void insertSort(int* Arry,int Size)
{
    int temp,n,m,time;
    for(m = 0;m<Size;m++)
    {
        temp = Arry[m];
        for(n = m-1;n>=0&&Arry[n]>temp;n--)
        {
            Arry[n+1] = Arry[n];
        }
        Arry[n+1] = temp;
    }
}
```

归并排序

```
void merge(int Arry[],int tempArry[], int Start, int Mid, int End)
{
    int i = Start, j=Mid+1, k = Start;
    while(i!=Mid+1 && j!=End+1)
    {
        if(Arry[i] > Arry[j])
        {
            tempArry[k++] = Arry[j++];
        }
        else
        {
            tempArry[k++] = Arry[i++];
        }
    }
    while(i != Mid+1)
    {
        tempArry[k++] = Arry[i++];
    }
    while(j != End+1)
    {
        tempArry[k++] = Arry[j++];
    }
    for(i=Start; i<=End; i++)
    {
        Arry[i] = tempArry[i];
    }

}

void mergeSort(int Arry[], int tempArry[], int Start, int End)
{
    int Mid;
    if(Start < End)
    {
        Mid = (Start + End) / 2;
        mergeSort(Arry, tempArry, Start, Mid);
        mergeSort(Arry, tempArry, Mid+1, End);
        merge(Arry, tempArry, Start, Mid, End);
    }
}
```

递归快排

```
void qSort(int Arry[],int Size)//这里size和下面end都是数组最后一个的下标
{
    int begin,end,head[Size],tail[Size],flag = 0,trap;
    head[flag] = 0,tail[flag] = Size,flag++;
    while(flag)
    {
        begin = head[flag-1],end = tail[flag-1],flag--;
        if(begin<end)
        {
            int n,temp;
            for(n = trap = begin;n<end;n++)
            {
                if(Arry[n]<Arry[end])
                {
                    temp = Arry[n];
                    Arry[n] = Arry[trap];
                    Arry[trap] = temp;
                    trap++;
                }
            }
            temp = Arry[n];
            Arry[n] = Arry[trap];
            Arry[trap] = temp;
        }
        if(begin>=end)
        {
            continue;
        }
        head[flag] = begin,tail[flag] = trap-1,flag++;
        head[flag] = trap+1,tail[flag] = end,flag++;
    }
}
```

非递归快排（数组栈模拟）

```c
void qSort_Recursion(int Arry[],int begin ,int end)
{
    if(begin<end)
    {
        int n,trap,temp;
        for(n = trap = begin;n<end;n++)
        {
            if(Arry[n]<Arry[end])
            {
                temp = Arry[n];
                Arry[n] = Arry[trap];
                Arry[trap] = temp;
                trap++;
            }
        }
        temp = Arry[n];
        Arry[n] = Arry[trap];
        Arry[trap] = temp;
        qSort_Recursion(Arry,  begin, trap-1);
        qSort_Recursion(Arry, trap+1, end);
    }
}
```