```
//将一整型数组正负数集中在两端
void Divide_Positive_Negative_int(int* data, int n)
{
    int i = 0, j = n - 1;
    while (i < j)
    {
        while (i < j && data[j] >= 0)
            j--;
        if (i < j)
            swap(data[i++], data[j]);
        while (i < j && data[i] <= 0)
            i++;
        if (i < j)
            swap(data[i], data[j--]);
    }
}

//快速排序的非递归实现
template<class T>
void QuickSort_Stack(T* data, int n)
{
    int i = 0, j = n - 1;
    stack<int> s;
    s.push(i);
    s.push(j);
    while (!s.empty())
    {
        int j = s.top();//后入右侧的，先取右侧
        s.pop();
        int i = s.top();
        s.pop();
        int pivot = QS_partition(data, i, j);
        if (pivot - 1 > i)
        {
            s.push(i);
            s.push(pivot - 1);
        }
        if (pivot + 1 < j)
        {
            s.push(pivot + 1);
            s.push(j);
        }
    }
}
```

```
template<class T>
int QS_partition(T* data, int i, int j)
{
    T tmp = data[i];
    while (i < j)
    {
        while (i < j && data[j] >= tmp)
            j--;
        if (i < j)
            data[i++] = data[j];
        while (i < j && data[i] <= tmp)
            i++;
        if (i < j)
            data[j--] = data[i];
    }
    data[i] = tmp;
    return i;
}

//已知k1-kn是堆，加上k(n+1)，使其仍为堆
template<class T>
void HeapAdd(T* data, int n)
{
    int i;
    while (i = n / 2)
    {
        if (data[i] < data[n])
        {
            swap(data[i], data[n]);
            n = i;
        }
        else break;
    }
}

//合并两个有序序列
template<class T>
int MergeTwoList(T* data1, T* data2, T* rst, int len1, int len2)
{
    int i = 0, j = 0;
    int k = 0;
    while (i < len1 && j < len2)
        if (data1[i] <= data2[j])
            rst[k++] = data1[i++];
```

```
            else rst[k++] = data2[j++];
        while (i < len1)
            rst[k++] = data1[i++];
        while (j < len2)
            rst[k++] = data2[j++];
        return k;
    }
```