MergeSort per array

```
void Merge(int A[],int start, int mid,int end){
    int n1,n2,i,j,k;
    n1=mid-start+1; //size of left array
    n2=end-mid;   //size of right array

    /*ora creo i due array per contenere i numeri da ordinare e grazie alla ricorsione
    so che entrambi questi aray sono già ordinati, quindi mi basterà di volta in volta
    controllare quale dei due array ha il primo elemento più piccolo (dato che so per
    certo che gli altri saranno più grandi di questi due). */
    int L[n1],R[n2];
    for(i=0;i<n1;i++){
        L[i]=A[start+i];
    }
    for(j=0;j<n2;j++){
        R[j]=A[mid+j+1];
    }
    i=0,j=0;

    /*mi basterà quindi di volta in volta controllare quale dei due array ha il primo
    elemento più piccolo, dato che so per certo che gli altri saranno più grandi di
    questi due */
    for(k=start;i<n1&&j<n2;k++){
        if(L[i]<R[j]){
            A[k]=L[i++];
        }
        else{
            A[k]=R[j++];
        }
    }
    /*se infine l'array sinistro ha più elementi di quello destro
    allora tutti i restanti elementi verrano inseriti in A[] */
    while(i<n1)
        A[k++]=L[i++];

    /*se infine l'array destro ha più elementi di quello sinistro
    allora tutti i restanti elementi verrano inseriti in A[] */
    while(j<n2)
        A[k++]=R[j++];
}

/*la funzione principale prima divide l'array in due parti, richiama se stessa su
queste due parti ed infine li fonde con la Merge() */
void MergeSort(int A[],int start,int end){
    int mid;                        //va chiamata nel main con:
    if(start<end){                  //start=0 e end=size-1
        mid=(start+end)/2;
        MergeSort(A,start,mid);
        MergeSort(A,mid+1,end);
        Merge(A,start,mid,end);
    }
}
```