1. (A) 請以虛擬碼描述「改良氣泡排序演算法」  
   Algorithm BubbleSort(A,n)  
   Input:有n個整數的array A  
   Output: array A(由小排至大)  
   int flag = 0//紀錄是否swap，如果該次迴圈有swap紀錄1，無則記0  
   for i=n-1 to i=1  
    for j=0 to j=i-1  
    if(A[j]>A[j+1])  
    swap(A[j], A[j+1])  
    flag = 1  
    end if  
    end for  
    if(flag==0)  
    break  
    flag = 0

end for  
return A

1. (B) 分析改良氣泡排序演算法最佳、最差與平均時間複雜度  
    最佳: n-1-0= n-1次 為O(n)  
    最差:(n-1+1)(n-1)/2 = n(n-1)/2次 為O(n^2)  
    平均: = O(n^2)
2. (C) 使用虛擬碼描述堆積排序(heap sort)演算法  
    Algorithm heapsort(A,n)  
    Input:有n個整數的array A  
    Output: array A(由小排至大)  
      
    MaxHeap(A,n)  
    for i=floor(n/2) to i=1 //有floor(n/2)個內部節點當parent  
    Heap(A,i,n)  
    end for  
      
    HeapSort(A,n)  
    MaxHeap(A,n)  
    for i=n to i=2  
    swap(A[1],A[i])  
    Heap(A,1,n)  
    end for  
    return A  
      
    Heap(A, int i, n)  
    left = 2i  
    right = 2i+1  
    if(A[left]>A[i]) and (left<=n)  
    Max = left  
    else   
    Max = i  
    end if  
    if(A[right]>A[Max]) and (right<=n)  
    Max = right  
    end if  
    if(Max != i)  
    swap(A[i],A[Max])//若左或右node大於parent，則做交換  
    else  
    return  
    end if  
    Heap(A,Max,n)
3. (D) 分析堆積排序演算法時間空間複雜度  
    時間複雜度: (1) MaxHeap: floor(n/2) = O(n)  
    (2) HeapSort: (n-1)logn = O(nlogn)  
    -> c1n + c2nlogn = O(nlogn)  
    空間複雜度: O(1) //無複製陣列
4. (E) 使用虛擬碼描述基數(radix sort)演算法  
    Algorithm RadixSort(A,n)  
    Input:有n個整數的array A, array中數的上限Max  
    Output: array A(由小排至大)   
    int key, counter[10],container[10] [n]  
    // 儲存0-9, 儲存mod基數後0-9的個數,   
    int r = 1 // 基數  
    while(r<=Max)  
    {  
    \*分配  
    for i=0 to i=n-1  
    key = A[i]/r %10  
    container[key][counter[key]] = A[i]  
    counter[key]++  
    end for  
    r\*=10 //更新基數  
      
    \*合併  
    int size = 0//紀錄array放了幾個數  
    for i=0 to i=9  
    if(counter[i]!=0)  
    for j=0 to j=counter[i]-1  
    A[size] = container[i][j]  
    size++  
    end for  
    end if  
    counter[i] = 0//歸零做下一個基數為單位放入使用  
    end for  
    }  
    return A
5. (F) 分析基數排序演算法時間空間複雜度

時間複雜度:分配: 3n = O(n)  
 合併: O(r)  
 共執行 k(log10Max) 回合  
 -> O(k(n+r))  
 空間複雜度: 需要r個容器，其容量為n  
 ->O(rn)