HW6

王蝶晟　PB1711614

1.

a.

设最大重叠点为点M，共有K个区间集合过点M。设为$I_1, I_2...I_K$,其中$I_K = [L_K, R_K]$。则对$\forall i \in [1, K]$，$L_i \le M \le R_i$。令$L^* = max(L_1, L_2...L_K)$，则$L_i \le M \le L^*$。假设$\exists i \in [1, K]$，使得$R_i < L^*$，所以$R_i < L^* \le M$,与$\forall i \in [1, K]$，$L_i \le M \le R_i$矛盾。因此，$\forall i \in [1, K]$，$L_i \le L^* \le R_i$。所以共有K个区间集合过点$L^*$，由此得证最大重叠点可以是其中一个区间的左端点，证毕。

b.

用一个红黑树来记录所有的区间的端点，将端点从左到右一个个插入红黑树，当插入的是左端点L_i时，$P(e_i) += 1$，同理当插入的是右端点R_i时，$P(e_i) -= 1$。即插入左端点时重叠数+1，插入右端点时重叠数-1。当多个端点有相同的值时，先把同值的所有左端点都插入然后插入右端点。令$e_1, e_2...e-n$为区间端点的对应序列，S(i,j)为$1 \le i \le j \le n$时$P(e_i) + P(e_{i+1}) + ... + P(e_j)$的和，找一个i使得S(1,i)最大。对于每个节点x，存储V(x)=S(L(x),R(x))，即子树中所有结点的和。M(x)=max(S(L(x),i))，O(x)为对应的i的值。

$V(x) = V(L(x)) + P(x) + V(R(x))$

$M(x) = max\{M(L(x)), V(L(x)) + P(x), V(L(x)) + P(x) + M(R(x))\}$

计算O(x)可以通过算得的M(x)来计算。

INTERVAL_INSERT和INTERVAL_DELETE可以通过调用红黑树的插入和删除操作，然后进行对V(x),M(x),O(x)的维护来实现。返回最大重叠点FIND_POM只需要寻找满足O(x)的x对应点，即为最大重叠点，所需要的时间复杂度为O(1)

2.

a.

对于将要被删除的结点x，假设原斐波那契堆中有n个元素，则x的最大度数为lgn。所以x的子结点个数$x.degree$最多有lgn个，所以第八行将x的子结点插入到H的根链表中操作的时间复杂度应该是$O(x.degree)$

b.

第八行将x的子结点插入根链表需要$O(x.degree)$的时间，而c次调用CASCADING_CUT需要$O(C)$的时间。所以该算法的总运行时间为$O(c + x.degree)$

3.

MAKE-SET:

function MAKE-SET(x)

  $S.head = x$

  $S.tail = x$

  $x.next = NIL$

  $x.rank = 0$

end function

FIND-SET:

function FIND-SET(x)

  return $rep[x]$ //rep[x]表示包含x的集合对象

end function

UNION:

function UNION(x,y)

  if $x.size > y.size$ **then**

    $Link \quad y \quad to \quad rep[x].tail$

    $x.rank+ = y.rank$

  else

    $Link \quad x \quad to \quad rep[y].tail$

    $y.rank+ = x.rank$