一、由题可得：

$$C(n)-1=\begin{cases}0, & n=1\\ 2\left(C\left(\frac{n}{2}\right)-1\right)+n, & n\geq 2\end{cases}$$

令$\mathrm{F}(n)=C(n)-1$，则

$$F(n)=\begin{cases}0, & n=1\\ 2F\left(\frac{n}{2}\right)+n, & n\geq 2\end{cases}$$

所以$\mathrm{d}=0, a=2, c=2, b=1, x=1$

所以由定理 1 得：

$$F(n)=n\log_2 n$$

所以

$$C(n)=n\log_2 n+1$$

其渐进复杂性为 O($n\log_2 n$)

二、

1. Prim 算法基于顶点集合生成 MST，其复杂度与顶点个数 v 有关，而 Kruskal 算法基于边集合生成 MST，其复杂度与边个数 e 有关。因此顶点少的图应该使用 Prim 算法，边少的图应该使用 Kruskal 算法。
2. 这样集成可以充分利用两种算法在不同问题情形下的优势，实现最优效率。

三、该算法正确。分析如下：

当 n=1 时，生成的排列显然正确

假设 n=k 时，生成的排列正确，只需要证明当 n=k+1 时，

从 1->n 的循环过程中，交换产生的 A[n]值不同。

下面进行证明：

对 n=1->5 的情况进行观察，发现 n=奇数时，该算法生成的最后一个排列是把第一个排列的第一位数和最后一位数交换顺序。n=偶数时，该算法生成的最后一个排列是把第一个排列的数字循环左移。因此对任意 n 的情况进行归纳：

1. 当 n 为偶数时，按照该算法每次生成前 n-1 数的排列时，最后一个排列是把最初排列的第一个数和最后一个数调换顺序。所以对于第 2->n-2 个位置的数，不会发生变化。所以按顺序把这些位置的数字换出来不会产生重复。对于第 1 位和第 n-1 位的数，由于中间调换了偶数次，所以第一次和最后一次恰好可以把这两个数调换到第 n 位，所以从 1->n 的循环恰好没有重复的把 1->n 交换到 A[n]，所以生成的排列正确。
2. 当 n 为奇数时，按照该算法每次生成前 n-1 数的排列时，最后一个排列是把最初排列的数循环左移一位。因此每次生成前 n-1 个数的排列后，第一个数都不一样，把他们换到 A[n]，就可以得到 n 个数的全部排列。

综上，该算法是正确的。

可以推出，时间复杂度递推公式为：

$$C(n)=\begin{cases}1, & n=1\\ n(C(n-1)+2), & n\geq 2\end{cases}$$

计算可得，时间复杂度为$O(n!)+O(n^{n-1})$

四、伪代码如下：

```
find_min(a, left, right):
    If left==right:
```

return a[left], left

min_left, idx_left = find_min($a, left, left + \left\lfloor \frac{right-left+1}{2} \right\rfloor - 1$)

min_right, idx_right = find_min($a, left + \lfloor \frac{right-left+1}{2} \rfloor, right$)

if min_left < min_right:

return min_left, idx_left

else

return min_right, idx_right

时间复杂度递推公式为：

$$C(n) = \begin{cases} 1, & n = 1 \\ 2 * C\left(\frac{n}{2}\right) + 1, & n \geq 2 \end{cases}$$

计算可得，时间复杂度为：O(2n-1)，可见其效率低于蛮力算法。

五、首先得到递推公式：

$$J(n) = \begin{cases} 2J\left(\frac{n-1}{2}\right) + 1, & if\ n\ is\ odd \\ 2 * J\left(\frac{n}{2}\right) - 1, & if\ n\ is\ even \end{cases}$$

列举出前面一些项：

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
J(n)	1	1	3	1	3	5	7	1	3	5	7	9	11	13	15

观察可知，当 n=2^k 时，J(n)=1，否则以加 2 的规律递增

所以猜测：$J(n) = 2\left(n - 2^{\lfloor \log_2 n \rfloor}\right) + 1$

下面证明：

1. 当 n=1 时，显然成立
2. 假设 n=k 时成立，下面证明 n=k+1 时成立。

a) 若 k+1 为奇数，则$\mathrm{J(k+1)} = 2J\left(\frac{k}{2}\right) + 1 = 2 * \left(2\left(\frac{k}{2} - 2^{\lfloor \log_2(\frac{k}{2}) \rfloor}\right) + 1\right) + 1 = 2\left(k + 1 - 2^{\lfloor \log_2(k) - 1 \rfloor + 1}\right) + 1$

所以只需要证明$2^{\lfloor \log_2(k) - 1 \rfloor + 1} = 2^{\lfloor \log_2(k+1) \rfloor}$，即$\lfloor \log_2(k) \rfloor = \lfloor \log_2(k+1) \rfloor$

因为 k 为偶数，所以该式成立。

所以当 n=k+1 为奇数时，公式成立。

b) 若 k+1 为偶数，则$\mathrm{J(k+1)} = 2J\left(\frac{k+1}{2}\right) - 1 = 2 * \left(2\left(\frac{k+1}{2} - 2^{\lfloor \log_2(\frac{k+1}{2}) \rfloor}\right) + 1\right) - 1 = 2\left(k + 1 - 2^{\lfloor \log_2(k+1) - 1 \rfloor + 1}\right) + 1 = 2\left(k + 1 - 2^{\lfloor \log_2(k+1) \rfloor}\right) + 1$

所以当 n=k+1 为偶数时，公式成立。

综上，对于所有的 n，有该公式成立。