1)Cutting

```
def cut(size):
        if size==1:
                return 0
        return cut(int(size/2)+(size%2))+1 ==> T(n)=T(n/2)+1  ==>Θ(log(n))
```

Explanation: If the size of wire is 1 , no need for cut.
If the size of wire is greater then 1:
size is odd: new size= (size/2)+1
size is even: new size= (size/2)+0
When new size is calculated, that means wire cut so +1

2)Worst_best

```
def divide_best(rates):
        if len(rates)==1:
                return rates[0]
        mid=int(len(rates)/2)
        in1=divide_best(rates[0:mid])==>T(n/2)
        in2 = divide_best(rates[mid:len(rates)])==>    T(n/2)
        if in1>in2:
                return in1
        else:
                return in2
```

T(n)=2*T(n/2)+1==> T(n)=Θ(n) by Master Theorem

```
def divide_worst(rates):
        if len(rates) == 1:
                return rates[0]
        mid = int(len(rates)/2)
        in1 = divide_worst(rates[0:mid])==>      T(n/2)
        in2 = divide_worst(rates[mid:len(rates)])==>  T(n/2)
        if in1 < in2:
                return in1
        else:
                return in2
```

T(n)=2*T(n/2)+1==> T(n)=Θ(n) by Master Theorem

Explanation: Divide into two array ,call again separately.

3)Meaningful

```
def decrease(arr,k):
    min=arr[0]
    index=0
    for i in range(0,len(arr)):
        if min>arr[i]:
            min=arr[i]
            index=i
    if k==1:
        return min
    arr.pop(index)
    return decrease(arr,k-1)
```

==> $\sum_{i=0}^{n-1} 1 = n \Rightarrow \Theta(n)$

==> $T(k)=T(k-1)+\Theta(n)$

$T(n)=\Theta(n*k)$

Explanation: First, find the minimum number and remove it from array, then call the function again with decreasing n by 1.
k means nth smallest needed.

4)Find_rop

Explanation: I used a mergesort algorithm to find the number of reverse ordered pairs but every time when an element of the rigth array copied to main array ,I increased the counter by size of remaing left array .Meaning of that number is there are that number of reverse ordered pairs.

Complexity: I used a mergesort algorithm so;

$\Theta(n*\log(n))$

5)Exponent

```
def brute(a,n):
    res=1
    for i in range(0,n):
        res=res*a
    return res
```

==> $\sum_{i=0}^{n-1} 1 = \Theta(n)$

```
def divide(a,n):
    if (n==1):
        return a
    mid=int(n/2)
    return divide(a,mid)*divide(a,n-mid)
```

==>$T(n)=2*T(n/2)+1$

==>$T(n)=\Theta(n)$ by Master theorem

$a^n=a^{(n/2)}*a^{(n/2)}$