1. 课上测试完善：

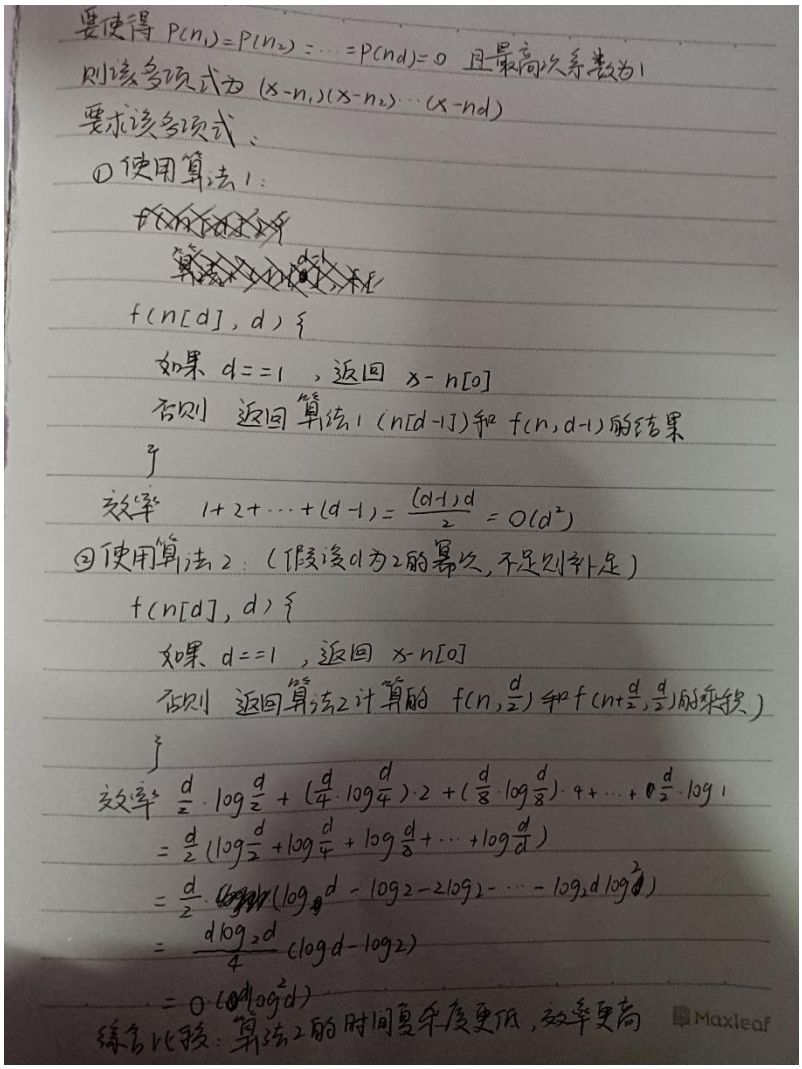
```python
def partition(a, p, r):  # 以一个确定的基准元素a[p]对子数组a[p:r]进行划分
    i = p
    j = r+1
    x = a[p]
    while 1:
        while 1:
            i += 1
            if a[i] < x and i < r:
                pass
            else:
                break
        while 1:
            j -= 1
            if a[j] > x:
                pass
            else:
                break
        if i >= j:
            break
        mid = a[i]
        a[i] = a[j]
        a[j] = mid
    a[p] = a[j]
    a[j] = x
    return j
```

```python
def randpartition(a, p, r):  # 随机化划分
    i = random.randrange(p, r)
    mid = a[i]
    a[i] = a[p]
    a[p] = mid
    return partition(a, p, r)
```

```python
def randselect(a, p, r, k):  # 选择
    if p == r:
        return a[p]
    i = randpartition(a, p, r)
    j = i - p + 1
    if k <= j:
        return randselect(a, p, i, k)
    else:
        return randselect(a, i + 1, r, k-j)
```

```python
a = [4, 5, 1, 6, 2, 7, 3, 8]
for i in range(1, 5):
    print(randselect(a, 0, 7, i))
```

2.



3.
代码如下：

```
#include <iostream>
#include <fstream>
using namespace std;

int factorial(int n)
{
	int C = 1;

	while (n >= 1)
	{
		C = C * n;
		--n;
	}

	return C;
}//计算 n 的阶乘

int dic(int arrange[], int n)
{
	int head, tail, count = 0;

	if (n == 1)
		return 1;
	for (int i = 0; i < n - 1; ++i)
	{
		if (arrange[n - 1] > arrange[i])
			++count;
	}
	head = factorial(n - 1) * count;
	tail = dic(arrange, n - 1);

	return tail + head;
}//递归计算字典序

void redic(int arrange[], int n, int order)
{
	int num, mid, * array, count;

	array = (int*)malloc(n * sizeof(int));
	if (!array)
		exit(-1);
	for (int i = 0; i < n; ++i)
```

```
            *(array + i) = 1;
        for (int i = 0; i < n; ++i)
        {
            num = factorial(n - i - 1);
            mid = order / num;
            order = order - mid * num;
            ++mid;
            count = 0;
            for (int k = 0; k < n; ++k)
            {
                if (*(array + k) == 1)
                    ++count;
                if (count == mid)
                {
                    arrange[i] = k + 1;
                    array[k] = 0;
                    break;
                }
            }
        }
}//循环计算下一个排列

int main()
{
        int n, * arrange, * mid, order;
        ifstream in;
        ofstream out;

        in.open("input.txt", ios::in);
        if (!in)
        {
            cout << "文件打开失败！" << endl;
            exit(-1);
        }
        in >> n;
        arrange = (int*)malloc(n * sizeof(int));
        if (!arrange)
            exit(-1);
        mid = (int*)malloc(n * sizeof(int));
        if (!mid)
            exit(-1);
        for (int i = 0; i < n; ++i)
            in >> *(mid + i);
        in.close();
```

```
    for (int i = 0; i < n; ++i)
        *(arrange + i) = *(mid + (n - i - 1));
    out.open("output.txt", ios::out);
    if (!out)
    {
        cout << "文件打开失败！" << endl;
        exit(-1);
    }
    order = dic(arrange, n) - 1;
    out << order << endl;
    redic(arrange, n, order + 1);
    for (int i = 0; i < n; ++i)
        out << *(arrange + i) << " ";

    return 0;
}
```

运行结果如下图：

```
8227
2 6 4 5 8 3 1 7
```