生成排列/组合

Generating permutations and combinations

按字典序生成下一个排列

- **Algorithm**：从后往前寻找第一个比后一项小的项 a_j（即 $a_j < a_{j+1}$ 且 $a_{j+1} > a_{j+2} > \cdots > a_n$），将 $a_{j+1}, \cdots, a_n$ 中大于 a_j 的最小项 a_k 放在第 j 个位置上，然后从小到大列出 $a_j, \cdots, a_n$ 中其他元素，放在第 $j+1$ 到第 n 个位置上.
- **Complexity**：$O(n)$
- **Code**：

```cpp
inline bool nextPermutation(int a[], int n){
// if return false, then all the permutations have been generated
    int j = n - 1, k = n;
    while(a[j] >= a[j+1])   j--;
    if(!j)  return false;
    while(a[j] >= a[k]) k--;
    swap(a[k], a[j]);
    int l = j + 1, r = n;
    while(r > l)    swap(a[l++], a[r--]);
    return true;
}
```

按字典序生成下一个位串

- **Algorithm**：模拟二进制数加一，即从后往前找到第一个 0，将其变成 1 并将其右边所有 1 变成 0.
- **Complexity**：$O(n)$
- **Code**：

```
inline void nextBitString(char s[], int n){
//s[n-1]s[n-2]...s[0] is a bit string which is not 11...1
    int k = 0;
    while(s[k] == '1'){
        s[k] = '0';
        k++;
    }
    s[k] = '1';
}
```

按字典序生成下一个 r 组合

- **Algorithm**：从后往前找到第一个 $a_i \neq n-r+i$ 的 a_i，用 a_i+1 代替 a_i，且对于 $j=i+1, i+2, \cdots, r$，用 $a_j+j-i+1$ 代替 a_j.
- **Complexity**：$O(n)$
- **Code**：

```
inline bool nextCombination(int a[], int n, int r){ // n elements, r-combination
    int i = r;
    while(a[i] == n - r + i)    i--;
    if(!i)  return false;
    a[i] = a[i] + 1;
    for(int j = i + 1; j <= r; j++)
        a[j] = a[i] + j - i;
    return true;
}
```