四、离散与组合数学

【大纲内容】

【2】集合

【2】加法原理

【2】乘法原理

【4】排列

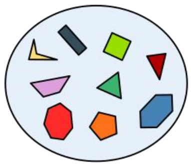
【4】组合

【4】杨辉三角

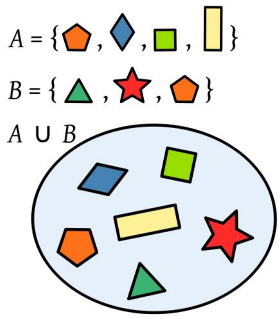
（一）集合（Set）

集合是一些**互不相同**的对象（元素）的整体，通常用大括号 {} 表示。

- 元素可以是数字、字母或其他东西。
- 集合里的元素没有顺序，也没有重复。



- 集合就像一个装东西的盒子，里面放的是不同的玩具，没有顺序，也不能有重复的玩具。

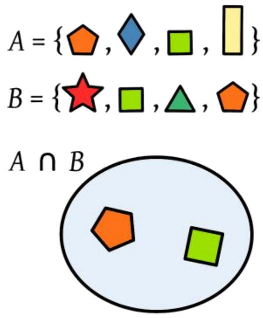


举例：

- A={1,2,3} 是包含数字 1、2、3 的集合。
- B={a,b,c} 是包含字母 a、b、c 的集合。

相关操作：

- **并集（Union）**：把两个集合的所有元素**合起来**，不重复。
- **交集（Intersection）**：两个集合中**共有的**元素。



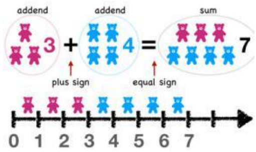
（二）加法原理（Addition Principle）

加法原理是指：

如果完成一个任务有 m 种方法，完成另一个**互不相干**任务有 n 种方法，那么完成这两个任务中**任意一个任务**共有 m+n 种方法。

举例说明

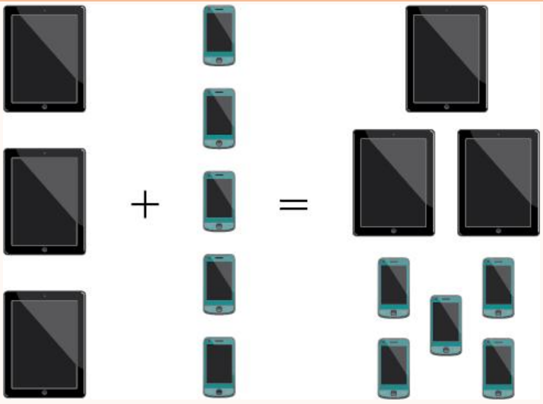
如果你可以选红色球有 3 种方法，选蓝色球有 4 种方法，那么选红色或蓝色球一共有 3 + 4 = 7 种方法。



举例：

选择穿衣服，有 3 件红衬衫和 4 件蓝衬衫，选择一件衬衫有多少种？答案是 3 + 4 = 7 种。

C++示例：使用集合（Set）



```
#include <iostream>
#include <set>
using namespace std;
int main() {
    set<int> A = {1, 2, 3};
    set<int> B = {3, 4, 5};
    // 并集
    set<int> unionSet = A;
    for (int x : B) {
        unionSet.insert(x);
    }
    cout << "集合 A 和 B 的并集是：";
    for (int x : unionSet) {
        cout << x << " ";
    }
    return 0;
}
```

考试注意事项 & 易错点

易错点	正确说明

集合元素顺序混淆	集合元素无顺序
重复元素计数错误	集合不允许重复元素
加法原理只适用于互不相干事件	任务必须互不重叠，不能重复计算

（三）乘法原理（Multiplication Principle）

乘法原理说明：

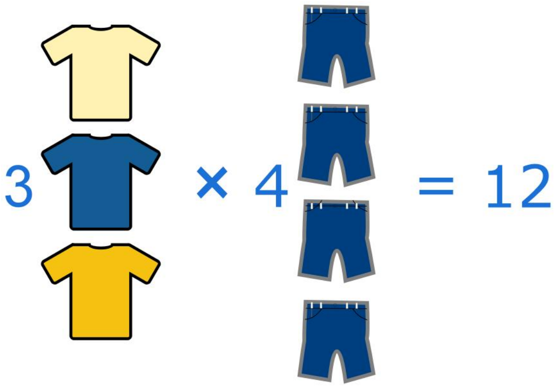
如果做第一件事有 m 种方法，做第二件事有 n 种方法，那么**做这两件事的所有可能**情况数是 m×n。

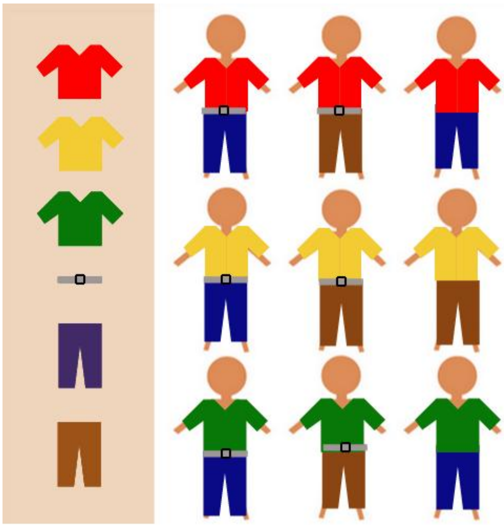
举例说明

- 想象你有 3 件不同颜色的衬衫和 2 条裤子，你穿衬衫和裤子的搭配共有多少种？
- 答案是 3×2=6 种。

举例：

有 3 顶帽子，4 双鞋子，选一顶帽子和一双鞋子的方法有多少？答案是 3×4=12 种。





（四）排列（Permutation）

排列是指从一组不同元素中，按照**顺序**选出若干元素组成一个序列。

- 排列数计算公式**从 n 个元素中选出 k 个排列数**是 P(n,k)= n! / (n−k)!
- n!（读作“n 的阶乘”）是 1×2×3×...×n。

举例说明

排列就是把物品按顺序排好，比如给 4 个不同的玩具选 3 个排队，不同顺序算不同排列。

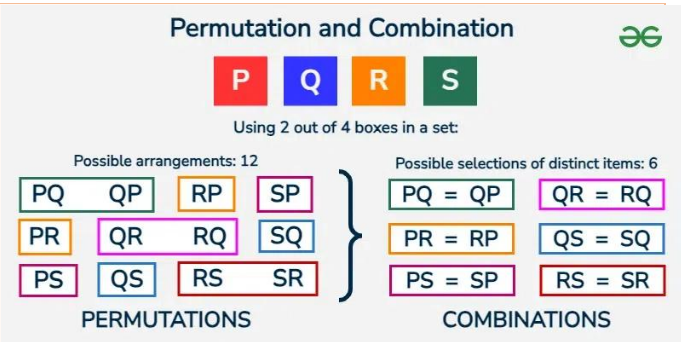
举例： $$P(n,r) = \frac{5!}{(5-4)!} = \frac{5x4x3x2x1}{1} = 120$$

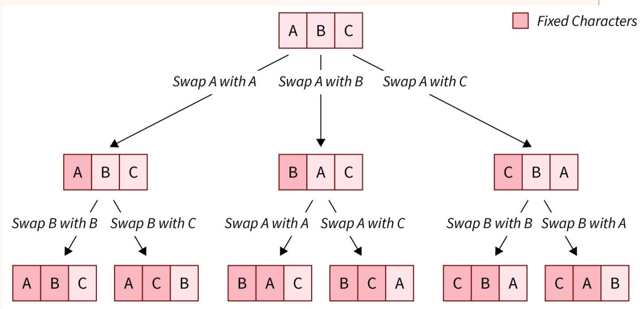
从 4 个数字（1，2，3，4）中选 3 个排成一排，有多少种方法?

计算：P(4,3)=4! / (4−3)! = 24 / 1=24 种。

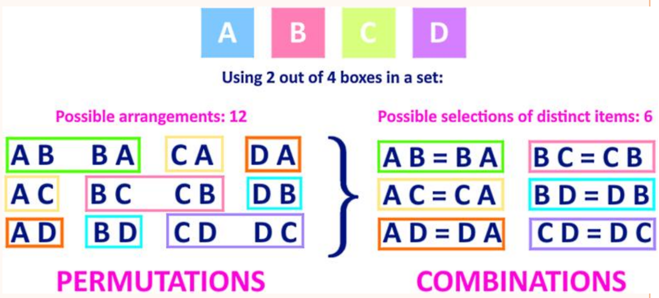
C++示例：计算排列数和乘法原理

```
#include <iostream>
using namespace std;
// 计算阶乘
int factorial(int n) {
    int res = 1;
    for (int i = 1; i <= n; i++)
        res *= i;
    return res;
}
// 计算排列数 P(n, k)
int permutation(int n, int k) {
    return factorial(n) / factorial(n - k);
}
int main() {
    int hats = 3, pants = 2;
```





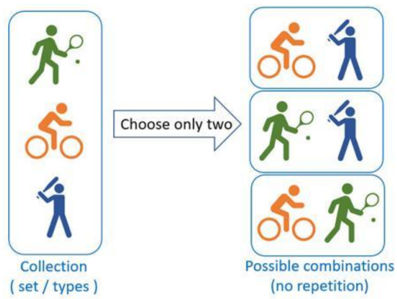
```
        cout << "穿搭组合数
        int n = 4, k = 3;
        cout << " 排 列 数
permutation(n, k) << endl;
        return 0;
    }
```



（五）组合（Combination）

组合是指从一组不同元素中，**不考虑顺序**地选出若干元素的方式数。

组合数计算公式：



C(n,k)= n! / k!(n−k)!

表示从 n 个元素中选出 k 个，不考虑顺序的不同选法。

举例说明

组合就是从玩具堆里选出几样玩具，不管选的顺序，只看选了哪些。

举例：

有 5 个不同的水果，从中选 3 个，有多少种选法？

计算：C(5,3)= 5! / (3!×2!)= 120 / (6×2)=10 种。

（六）杨辉三角（Pascal’ s Triangle）

杨辉三角是一种排列数字的三角形结构，其中每个数字是上一行左右两个数字之和。

- 它的**第 n 行第 k 个数**就是组合数 **C(n,k)**。
- 就像一座数字金字塔，底下的数字是上面两个数字加起来的。

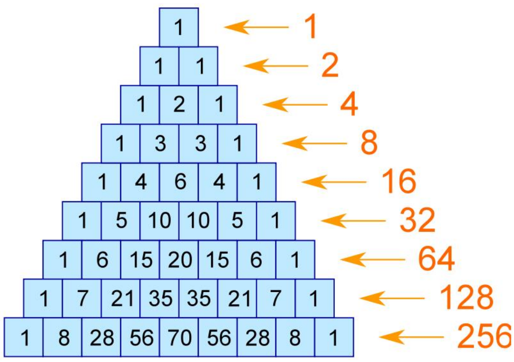
杨辉三角的前几行：

1

1 1

1 2 1

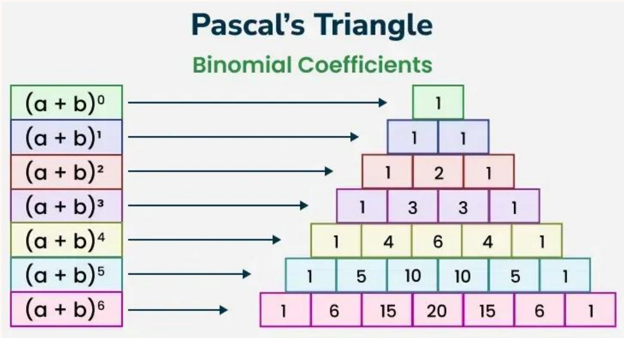
1 3 3 1



1 4 6 4 1

例如，第 4 行第 2 个数字是 3，就是从 4 个元素中选 2 个的组合数 C(4,2)=6。

C++示例：计算组合数和生成杨辉三角




```cpp
#include <iostream>
#include <vector>
using namespace std;
// 计算阶乘
long long factorial(int n) {
    long long res = 1;
    for (int i = 2; i <= n; i++) res *= i;
    return res;
}
// 计算组合数 C(n, k)
long long combination(int n, int k) {
    return factorial(n) / (factorial(k) * factorial(n - k));
}
// 生成杨辉三角前 n 行
void generatePascalsTriangle(int n) {
    vector<vector<long long>> triangle(n);
    for (int i = 0; i < n; i++) {
        triangle[i].resize(i + 1);
        triangle[i][0] = triangle[i][i] = 1;
        for (int j = 1; j < i; j++) {
            triangle[i][j] = triangle[i - 1][j - 1] + triangle[i - 1][j];
        }
    }
```

```
        for (auto &row : triangle) {
            for (auto num : row) cout << num << " ";
            cout << endl;
        }
    }
    int main() {
        int n = 5, k = 3;
        cout << "C(" << n << ", " << k << ") = " << combination(n, k) << endl;
        cout << "杨辉三角前 " << n << " 行: " << endl;
        generatePascalsTriangle(n);
        return 0;
    }
```