二进制枚举

二进制枚举是解决子集枚举一类问题的方法，对于一个包含 n 个元素的集合，其子集的数量为 2^n 种。通过二进制枚举，我们可以高效地遍历所有可能的子集。

蛋糕的美味值：https://www.lanqiao.cn/problems/8664/learning/

递归实现指数型枚举：https://www.lanqiao.cn/problems/19685/learning

五子棋对弈：https://www.lanqiao.cn/problems/19694/learning

基本原理

- 每个子集可以用一个长度为 n 的二进制数表示。
- 二进制数的每一位表示集合中对应元素是否被选中：
  - `1` 表示选中该元素。
  - `0` 表示不选中该元素。
- 例如，集合 $\{a, b, c\}$ 的子集可以用 3 位二进制数表示：
  - `000`：空集。
  - `001`：$\{c\}$。
  - `010`：$\{b\}$。
  - `011`：$\{b, c\}$。
  - `100`：$\{a\}$。
  - `101`：$\{a, c\}$。
  - `110`：$\{a, b\}$。
  - `111`：$\{a, b, c\}$。

模版

```cpp
#include <bits/stdc++.h>
using namespace std;

int main() {
    int n;
    cin >> n; // 输入集合的大小 n

    // 遍历所有可能的子集（2^n 种）
    for (int i = 0; i < (1 << n); i++) {
        // 检查每个元素是否被选中
        for (int j = 0; j < n; j++) {
            if ((i >> j & 1) == 1) { // 如果 i 的第 j 位为 1
                // 这里可以处理选中的元素
                // 例如：输出选中的元素
                cout << j << " ";
            }
        }
        cout << "\n"; // 换行，表示一个子集结束
    }

    return 0;
}
```

```java
import java.util.Scanner;

public class BinaryEnumeration {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt(); // 输入集合的大小 n

        // 遍历所有可能的子集（2^n 种）
        for (int i = 0; i < (1 << n); i++) {
            // 检查每个元素是否被选中
            for (int j = 0; j < n; j++) {
                if ((i >> j & 1) == 1) { // 如果 i 的第 j 位为 1
                    // 这里可以处理选中的元素
                    // 例如：输出选中的元素
                    System.out.print(j + " ");
                }
            }
            System.out.println(); // 换行，表示一个子集结束
        }
    }
}
```

```python
n = int(input())  # 输入集合的大小 n

# 遍历所有可能的子集（2^n 种）
for i in range(1 << n):
    # 检查每个元素是否被选中
    for j in range(n):
        if (i >> j) & 1 == 1:  # 如果 i 的第 j 位为 1
            # 这里可以处理选中的元素
            # 例如：输出选中的元素
            print(j, end=" ")
    print()  # 换行，表示一个子集结束
```

五子棋对弈

```cpp
#include<bits/stdc++.h>
using namespace std;

int b[5][5];  // 修正为5x5的棋盘
int a[25];    // 修正为25个元素

bool check() {
    int pos = 0;
    for (int i = 0; i < 5; i++) {
        for (int j = 0; j < 5; j++) {
            b[i][j] = a[pos++];
        }
    }

    // 检查行和列
    for (int i = 0; i < 5; i++) {
        int rowSum = 0, colSum = 0;
        for (int j = 0; j < 5; j++) {
            rowSum += b[i][j];
            colSum += b[j][i];
```

```cpp
        }
        if (rowSum == 0 || rowSum == 5 || colSum == 0 || colSum == 5) {
            return false;
        }
    }

    // 检查对角线
    int diag1 = 0, diag2 = 0;
    for (int i = 0; i < 5; i++) {
        diag1 += b[i][i];
        diag2 += b[i][4 - i];
    }
    if (diag1 == 0 || diag1 == 5 || diag2 == 0 || diag2 == 5) {
        return false;
    }

    return true;
}

int main() {
    int ans = 0;
    // 遍历所有可能的25位二进制数
    for (int i = 0; i < (1 << 25); i++) {
        int s = 0;
        for (int j = 0; j < 25; j++) {
            a[j] = (i >> j) & 1;
            if (a[j]) {
                s++;
            }
        }
        // 只有当棋子数量为13时才进行检查
        if (s != 13) continue;
        if (check()) {
            ans++;
        }
    }
    cout << ans << endl;
    return 0;
}
```

```java
public class Main {
    static int[][] b = new int[5][5]; // 5x5 的棋盘
    static int[] a = new int[25];     // 25 个元素

    // 检查棋盘是否满足条件
    static boolean check() {
        int pos = 0;
        for (int i = 0; i < 5; i++) {
            for (int j = 0; j < 5; j++) {
                b[i][j] = a[pos++];
            }
        }

        // 检查行和列
        for (int i = 0; i < 5; i++) {
            int rowSum = 0, colSum = 0;
            for (int j = 0; j < 5; j++) {
```

```
                rowSum += b[i][j];
                colSum += b[j][i];
            }
            if (rowSum == 0 || rowSum == 5 || colSum == 0 || colSum == 5) {
                return false;
            }
        }

        // 检查对角线
        int diag1 = 0, diag2 = 0;
        for (int i = 0; i < 5; i++) {
            diag1 += b[i][i];
            diag2 += b[i][4 - i];
        }
        if (diag1 == 0 || diag1 == 5 || diag2 == 0 || diag2 == 5) {
            return false;
        }

        return true;
    }

    public static void main(String[] args) {
        int ans = 0;
        // 遍历所有可能的 25 位二进制数
        for (int i = 0; i < (1 << 25); i++) {
            int s = 0;
            for (int j = 0; j < 25; j++) {
                a[j] = (i >> j) & 1;
                if (a[j] == 1) {
                    s++;
                }
            }
            // 只有当棋子数量为 13 时才进行检查
            if (s != 13) continue;
            if (check()) {
                ans++;
            }
        }
        System.out.println(ans);
    }
}
```

```
def check():
    global a, b
    pos = 0
    for i in range(5):
        for j in range(5):
            b[i][j] = a[pos]
            pos += 1

    # 检查行和列
    for i in range(5):
        row_sum = sum(b[i][j] for j in range(5))
        col_sum = sum(b[j][i] for j in range(5))
        if row_sum == 0 or row_sum == 5 or col_sum == 0 or col_sum == 5:
            return False

```

```python
    # 检查对角线
    diag1 = sum(b[i][i] for i in range(5))
    diag2 = sum(b[i][4 - i] for i in range(5))
    if diag1 == 0 or diag1 == 5 or diag2 == 0 or diag2 == 5:
        return False

    return True

def main():
    global a, b
    b = [[0] * 5 for _ in range(5)]  # 5x5 的棋盘
    a = [0] * 25                     # 25 个元素
    ans = 0

    # 遍历所有可能的 25 位二进制数
    for i in range(1 << 25):
        s = 0
        for j in range(25):
            a[j] = (i >> j) & 1
            if a[j] == 1:
                s += 1
        # 只有当棋子数量为 13 时才进行检查
        if s != 13:
            continue
        if check():
            ans += 1

    print(ans)

if __name__ == "__main__":
    main()
```