商店购物 (shopping.c/cpp/pas)

有 n 家商店，每家商店可以消费非负整数元，其中前 m 家商店里第 i 家商店消费上限为 w_i 元，求总消费恰好为 k 元的方案数。

商店购物 (shopping.c/cpp/pas)

有 n 家商店，每家商店可以消费非负整数元，其中前 m 家商店里第 i 家商店消费上限为 w_i 元，求总消费恰好为 k 元的方案数。

- 对于 20% 的数据，$n \leq 100$，$w_i \leq 100$，$k \leq 1000$。

商店购物 (shopping.c/cpp/pas)

有 n 家商店，每家商店可以消费非负整数元，其中前 m 家商店里第 i 家商店消费上限为 w_i 元，求总消费恰好为 k 元的方案数。

- 对于 20% 的数据，$n \le 100$，$w_i \le 100$，$k \le 1000$。
- 对于 50% 的数据，$n \le 100$，$w_i \le 300$，$k \le 100000$。

商店购物 (shopping.c/cpp/pas)

有 n 家商店，每家商店可以消费非负整数元，其中前 m 家商店里第 i 家商店消费上限为 w_i 元，求总消费恰好为 k 元的方案数。

- 对于 20% 的数据，$n \le 100$，$w_i \le 100$，$k \le 1000$。
- 对于 50% 的数据，$n \le 100$，$w_i \le 300$，$k \le 100000$。
- 对于 70% 的数据，$n, k \le 5 \times 10^6$，$m \le 20$，$w_i \le 300$。

商店购物 (shopping.c/cpp/pas)

有 n 家商店，每家商店可以消费非负整数元，其中前 m 家商店里第 i 家商店消费上限为 w_i 元，求总消费恰好为 k 元的方案数。

- 对于 20% 的数据，$n \le 100$，$w_i \le 100$，$k \le 1000$。
- 对于 50% 的数据，$n \le 100$，$w_i \le 300$，$k \le 100000$。
- 对于 70% 的数据，$n, k \le 5 \times 10^6$，$m \le 20$，$w_i \le 300$。
- 对于 100% 的数据，$n, k \le 5 \times 10^6$，$m \le 300$，$w_i \le 300$。

30 分做法

- 对于 20% 的数据，$n \leq 100$，$w_i \leq 100$，$k \leq 1000$。

30 分做法

- 对于 20% 的数据，$n \leq 100$，$w_i \leq 100$，$k \leq 1000$。
- 考虑 DP，设 $f[i][j]$ 表示考虑了前 i 家商店，消费总和为 j 的方案数，暴力枚举每家商店消费了多少钱即可。

30 分做法

- 对于 20% 的数据，$n \le 100$，$w_i \le 100$，$k \le 1000$。
- 考虑 DP，设 $f[i][j]$ 表示考虑了前 i 家商店，消费总和为 j 的方案数，暴力枚举每家商店消费了多少钱即可。
- 时间复杂度 $O(nwk)$。

50 分做法

- 对于 50% 的数据，$n \le 100$，$w_i \le 300$，$k \le 100000$。

50 分做法

- 对于 50% 的数据，$n \leq 100$，$w_i \leq 300$，$k \leq 100000$。
- 考虑优化 DP 转移的复杂度，转移显然可以用前缀和优化到 $O(1)$。

50 分做法

- 对于 50% 的数据，$n \le 100$，$w_i \le 300$，$k \le 100000$。
- 考虑优化 DP 转移的复杂度，转移显然可以用前缀和优化到 $O(1)$。
- 时间复杂度 $O(nk)$。

70 分做法

- 对于 70% 的数据，$n, k \le 5 \times 10^6$，$m \le 20$，$w_i \le 300$。

70 分做法

- 对于 70% 的数据，$n, k \le 5 \times 10^6$，$m \le 20$，$w_i \le 300$。
- 假如没有上限，那么就是简单的排列组合问题，用隔板法即可算出方案数为 $C(k+n-1, n-1)$。

70 分做法

- 对于 70% 的数据，$n, k \leq 5 \times 10^6$，$m \leq 20$，$w_i \leq 300$。
- 假如没有上限，那么就是简单的排列组合问题，用隔板法即可算出方案数为 $C(k+n-1, n-1)$。
- 考虑容斥，暴力枚举哪些商店必然超过了限制，而不关心其它商店，那么把 k 减去这些商店的上限 $+1$ 之和，即可求出贡献。

70 分做法

- 对于 70% 的数据，$n, k \le 5 \times 10^6$，$m \le 20$，$w_i \le 300$。
- 假如没有上限，那么就是简单的排列组合问题，用隔板法即可算出方案数为 $C(k+n-1, n-1)$。
- 考虑容斥，暴力枚举哪些商店必然超过了限制，而不关心其它商店，那么把 k 减去这些商店的上限 $+1$ 之和，即可求出贡献。
- 时间复杂度 $O(n+k+2^m)$。

100 分做法

- 注意到暴力枚举哪些商店必然超过了限制时，我们事实上只关心选的商店的上限之和以及选的商店个数的奇偶性。

100 分做法

- 注意到暴力枚举哪些商店必然超过了限制时，我们事实上只关心选的商店的上限之和以及选的商店个数的奇偶性。
- 考虑 DP，设 $f[i][j]$ 表示考虑了前 i 个商店，目前选的商店的上限 $+1$ 之和为 j 时的贡献。

100 分做法

- 注意到暴力枚举哪些商店必然超过了限制时，我们事实上只关心选的商店的上限之和以及选的商店个数的奇偶性。
- 考虑 DP，设 $f[i][j]$ 表示考虑了前 i 个商店，目前选的商店的上限 $+1$ 之和为 j 时的贡献。
- 如果这个商店不选，那么有 $f[i][j] += f[i-1][j]$，否则有 $f[i][j] -= f[i-1][j-w[i]-1]$。

100 分做法

- 注意到暴力枚举哪些商店必然超过了限制时，我们事实上只关心选的商店的上限之和以及选的商店个数的奇偶性。
- 考虑 DP，设 $f[i][j]$ 表示考虑了前 i 个商店，目前选的商店的上限 $+1$ 之和为 j 时的贡献。
- 如果这个商店不选，那么有 $f[i][j] += f[i-1][j]$，否则有 $f[i][j] -= f[i-1][j-w[i]-1]$。
- $ans = \sum_i f[n][i] C(k+n-1-i, n-1)$。

100 分做法

- 注意到暴力枚举哪些商店必然超过了限制时，我们事实上只关心选的商店的上限之和以及选的商店个数的奇偶性。
- 考虑 DP，设 $f[i][j]$ 表示考虑了前 i 个商店，目前选的商店的上限 $+1$ 之和为 j 时的贡献。
- 如果这个商店不选，那么有 $f[i][j]+=f[i-1][j]$，否则有 $f[i][j]-=f[i-1][j-w[i]-1]$。
- $ans=\sum_i f[n][i]C(k+n-1-i,n-1)$。
- 时间复杂度 $O(n+k+m^2w)$。

公路建设 (highway.c/cpp/pas)

给定一张 n 个点，m 条边的无向图。

公路建设 (highway.c/cpp/pas)

给定一张 n 个点，m 条边的无向图。

q 次询问，每次给定一个区间 $[l, r]$，问仅保留编号在这个区间内的边时，这个图的最小生成森林的边权和。

公路建设 (highway.c/cpp/pas)

给定一张 n 个点，m 条边的无向图。

q 次询问，每次给定一个区间 $[l, r]$，问仅保留编号在这个区间内的边时，这个图的最小生成森林的边权和。

- 对于 30% 的数据，$n \leq 100$，$m \leq 1000$，$q \leq 1000$。

公路建设 (highway.c/cpp/pas)

给定一张 n 个点，m 条边的无向图。

q 次询问，每次给定一个区间 $[l, r]$，问仅保留编号在这个区间内的边时，这个图的最小生成森林的边权和。

- 对于 30% 的数据，$n \le 100$，$m \le 1000$，$q \le 1000$。
- 对于 60% 的数据，$n \le 100$，$m \le 100000$，$q \le 15000$，边权随编号递增。

公路建设 (highway.c/cpp/pas)

给定一张 n 个点，m 条边的无向图。

q 次询问，每次给定一个区间 $[l, r]$，问仅保留编号在这个区间内的边时，这个图的最小生成森林的边权和。

- 对于 30% 的数据，$n \le 100$，$m \le 1000$，$q \le 1000$。
- 对于 60% 的数据，$n \le 100$，$m \le 100000$，$q \le 15000$，边权随编号递增。
- 对于 100% 的数据，$n \le 100$，$m \le 100000$，$q \le 15000$。

30 分做法

- 对于 30% 的数据，$n \leq 100$，$m \leq 1000$，$q \leq 1000$。

30 分做法

- 对于 30% 的数据，$n \le 100$，$m \le 1000$，$q \le 1000$。
- 暴力用 Kruskal 算法求出最小生成森林即可。

30 分做法

- 对于 30% 的数据，$n \leq 100$，$m \leq 1000$，$q \leq 1000$。
- 暴力用 Kruskal 算法求出最小生成森林即可。
- 时间复杂度 $O(qm(\log m + \alpha(n)))$。

60 分做法

- 对于 60% 的数据，边权随编号递增。

60 分做法

- 对于 60% 的数据，边权随编号递增。
- 注意到边权随编号递增，因此从大到小依次加入每条边。

60 分做法

- 对于 60% 的数据，边权随编号递增。
- 注意到边权随编号递增，因此从大到小依次加入每条边。
- 每加入一条边时，如果没有成环，那么它就是树边，否则要切掉环上最大的那条边，再连上这条边。

60 分做法

- 对于 60% 的数据，边权随编号递增。
- 注意到边权随编号递增，因此从大到小依次加入每条边。
- 每加入一条边时，如果没有成环，那么它就是树边，否则要切掉环上最大的那条边，再连上这条边。
- 那么询问 $[l, r]$ 的答案就是加入了所有 $[l, m]$ 的边后，编号不超过 r 的树边的权值之和。

60 分做法

- 对于 60% 的数据，边权随编号递增。
- 注意到边权随编号递增，因此从大到小依次加入每条边。
- 每加入一条边时，如果没有成环，那么它就是树边，否则要切掉环上最大的那条边，再连上这条边。
- 那么询问 $[l, r]$ 的答案就是加入了所有 $[l, m]$ 的边后，编号不超过 r 的树边的权值之和。
- 对于树结构的维护可以暴力，对于权值和的询问可以用树状数组维护。

60 分做法

- 对于 60% 的数据，边权随编号递增。
- 注意到边权随编号递增，因此从大到小依次加入每条边。
- 每加入一条边时，如果没有成环，那么它就是树边，否则要切掉环上最大的那条边，再连上这条边。
- 那么询问 $[l, r]$ 的答案就是加入了所有 $[l, m]$ 的边后，编号不超过 r 的树边的权值之和。
- 对于树结构的维护可以暴力，对于权值和的询问可以用树状数组维护。
- 时间复杂度 $O(mn + (m + q)\log m)$。

100 分做法

- 考虑用线段树直接维护每个区间的答案。

100 分做法

- 考虑用线段树直接维护每个区间的答案。
- 注意到一个区间最多只有 $n-1$ 条树边有用，所以线段树每个节点按权值从小到大保存区间内用到的树边即可。

100 分做法

- 考虑用线段树直接维护每个区间的答案。
- 注意到一个区间最多只有 $n-1$ 条树边有用，所以线段树每个节点按权值从小到大保存区间内用到的树边即可。
- 合并两个区间的信息时，只需要将树边归并，然后做 Kruskal 算法。

100 分做法

- 考虑用线段树直接维护每个区间的答案。
- 注意到一个区间最多只有 $n-1$ 条树边有用，所以线段树每个节点按权值从小到大保存区间内用到的树边即可。
- 合并两个区间的信息时，只需要将树边归并，然后做 Kruskal 算法。
- 时间复杂度 $O((m+q\log m)n\alpha(n))$。

航海舰队 (sailing.c/cpp/pas)

给定一个 $n \times m$ 的网格图，有些点是障碍点。

航海舰队 (sailing.c/cpp/pas)

给定一个 $n \times m$ 的网格图，有些点是障碍点。

地图上还有一支固定阵形的船队，问船队能扫过的格子数。

航海舰队 (sailing.c/cpp/pas)

给定一个 $n \times m$ 的网格图，有些点是障碍点。
地图上还有一支固定阵形的船队，问船队能扫过的格子数。

- 对于 30% 的数据，$n, m \le 50$。

航海舰队 (sailing.c/cpp/pas)

给定一个 $n \times m$ 的网格图，有些点是障碍点。
地图上还有一支固定阵形的船队，问船队能扫过的格子数。

- 对于 30% 的数据，$n, m \leq 50$。
- 对于 60% 的数据，$n, m \leq 500$，阵形为一个满的矩形。

航海舰队 (sailing.c/cpp/pas)

给定一个 $n \times m$ 的网格图，有些点是障碍点。
地图上还有一支固定阵形的船队，问船队能扫过的格子数。

- 对于 30% 的数据，$n, m \le 50$。
- 对于 60% 的数据，$n, m \le 500$，阵形为一个满的矩形。
- 对于 100% 的数据，$n, m \le 700$。

30 分做法

- 对于 30% 的数据，$n, m \leq 50$。

30 分做法

- 对于 30% 的数据，$n, m \leq 50$。
- 暴力 BFS 即可。

30 分做法

- 对于 30% 的数据，$n, m \le 50$。
- 暴力 BFS 即可。
- 时间复杂度 $O(n^2m^2)$。

60 分做法

- 对于 60% 的数据，$n, m \leq 500$，阵形为一个满的矩形。

60 分做法

- 对于 60% 的数据，$n, m \leq 500$，阵形为一个满的矩形。
- 对障碍物预处理出二维前缀和，那么就可以 $O(1)$ 判断一个矩形内是否有障碍物。

60 分做法

- 对于 60% 的数据，$n, m \leq 500$，阵形为一个满的矩形。
- 对障碍物预处理出二维前缀和，那么就可以 $O(1)$ 判断一个矩形内是否有障碍物。
- 同理也可以用差分二维前缀和对答案数组进行打标记。

60 分做法

- 对于 60% 的数据，$n, m \leq 500$，阵形为一个满的矩形。
- 对障碍物预处理出二维前缀和，那么就可以 $O(1)$ 判断一个矩形内是否有障碍物。
- 同理也可以用差分二维前缀和对答案数组进行打标记。
- 然后直接 BFS 即可，判断与打标记都是 $O(1)$ 的。

60 分做法

- 对于 60% 的数据，$n, m \leq 500$，阵形为一个满的矩形。
- 对障碍物预处理出二维前缀和，那么就可以 $O(1)$ 判断一个矩形内是否有障碍物。
- 同理也可以用差分二维前缀和对答案数组进行打标记。
- 然后直接 BFS 即可，判断与打标记都是 $O(1)$ 的。
- 时间复杂度 $O(nm)$。

100 分做法

- 首先抠出包围了阵形的最小矩形。

100 分做法

- 首先抠出包围了阵形的最小矩形。
- 将地图拉伸成一条链，即将第一行、第二行、第三行按顺序连接。阵形也可以用同样的方法处理。

100 分做法

- 首先抠出包围了阵形的最小矩形。
- 将地图拉伸成一条链，即将第一行、第二行、第三行按顺序连接。阵形也可以用同样的方法处理。
- 那么问题转化为，给定两个 01 串 S 和 T，问每个 S 中长度为 $|T|$ 的子串是否存在一个点，两个串对应字符都是 1。

100 分做法

- 首先抠出包围了阵形的最小矩形。
- 将地图拉伸成一条链，即将第一行、第二行、第三行按顺序连接。阵形也可以用同样的方法处理。
- 那么问题转化为，给定两个 01 串 S 和 T，问每个 S 中长度为 $|T|$ 的子串是否存在一个点，两个串对应字符都是 1。
- 将 T 串翻转，那么就变成了卷积的形式，FFT 计算即可。

100 分做法

- 首先抠出包围了阵形的最小矩形。
- 将地图拉伸成一条链，即将第一行、第二行、第三行按顺序连接。阵形也可以用同样的方法处理。
- 那么问题转化为，给定两个 01 串 S 和 T，问每个 S 中长度为 $|T|$ 的子串是否存在一个点，两个串对应字符都是 1。
- 将 T 串翻转，那么就变成了卷积的形式，FFT 计算即可。
- 在 BFS 求出所有可行的位置之后，对于答案的计算，也是卷积的形式，用 FFT 加速即可。

100 分做法

- 首先抠出包围了阵形的最小矩形。
- 将地图拉伸成一条链，即将第一行、第二行、第三行按顺序连接。阵形也可以用同样的方法处理。
- 那么问题转化为，给定两个 01 串 S 和 T，问每个 S 中长度为 $|T|$ 的子串是否存在一个点，两个串对应字符都是 1。
- 将 T 串翻转，那么就变成了卷积的形式，FFT 计算即可。
- 在 BFS 求出所有可行的位置之后，对于答案的计算，也是卷积的形式，用 FFT 加速即可。
- 时间复杂度 $O(nm\log(nm))$。

Thank you!