https://www.luogu.com.cn/problem/P1043

chenjiuri_shuziyouxi

- 给定一个圆盘，将其分为m组。
- 将其中每组的数字相加，求和。对10取模。
- 求出各部分相乘的最大值。

20*mins*

- 采用动态规划的思想
  - 探究解空间。暴力枚举当前的分界线所在的位置，就是做一个组合计算。复杂度相当的大。
- 像这一类关于模运算的最有问题，有如下经验
  - 2022桂林
    - 其中是发现解的本质不同的解有限性。
    - 当数据非常大的时候就考虑这一方面。
  - 这里数据比较小，所以考虑一个计数问题。
  - 把一部分当成整体时，讨论其他部分的情况。
  - 因为是相乘组合问题。考虑所有方案中的最大情况。
- 定义状态$f_{i,j,k}$为对应区间为$[i,\ j]$被分为k部分的最优结果。
  - 一个状态的解-» 满足问题的解结构的最优情况。
  - $f_{i,j,k} = max(f_{i,j,k}, f_{i,t,k-1} + f_{t+1,j,1})$;
  - 这样从小到大枚举就能枚举出所有情况。

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

void MAIN();
int main()
{
    ios::sync_with_stdio(false);
    cin.tie(nullptr), cout.tie(nullptr);
    MAIN();
}
typedef long long ll;
const int maxn = 200 + 10;
//------code-----٩(´ω`*)و ------靓仔代码-----٩(´ω`*)و ----talk is cheap ,
show me the code--------

int a[maxn];
int S[maxn][maxn][10], B[maxn][maxn][10]; // smallest,biggist;
```

```
int mod(int x)
{
    return (x % 10 + 10) % 10;
}

void MAIN()
{
    int n, m;
    cin >> n >> m;
    memset(S, 0x3f, sizeof S);
    //  memset(B, 0xf0, sizeof B);
    // cout << S[1][1][1] << '\n';
    // cout << B[1][1][1] << '\n';
    for (int i = 1; i <= n; i++)
    {
        cin >> a[i];
        a[i] += a[i - 1];
    }                                //前缀和的情况。
    for (int i = 1; i <= n; i++) //先进行断链成环。
        a[i + n] = a[i] + a[n];
    for (int i = 1; i <= 2 * n; i++)
        for (int j = i; j <= 2 * n; j++)
            B[i][j][1] = S[i][j][1] = mod(a[j] - a[i - 1]);
    for (int k = 2; k <= m; k++)
        for (int i = 1; i <= 2 * n; i++)
            for (int j = i + k - 1; j <= 2 * n; j++)
                for (int h = i + k - 2; h < j; h++)
                {
                    S[i][j][k] = min(S[i][j][k], (S[i][h][k - 1] * S[h +
1][j][1]));
                    B[i][j][k] = max(B[i][j][k], B[i][h][k - 1] * B[h +
1][j][1]);
                }
    int ans1 = 0;
    int ans2 = 1e9;
    for (int i = 1; i <= n; i++)
        ans1 = max(ans1, B[i][i + n - 1][m]), ans2 = min(ans2, S[i][i +
n - 1][m]);
    cout << ans2 << '\n'
         << ans1 << '\n';
}
```

生长思考：

- 枚举解的问题，分步乘法原理：
    - 这里对于子问题$f_{i,j}$的解：
        - 就是枚举第一个断点，然后剩下的几个机会机会在剩下的区间中产生。

        - 确保了相关小规模的子问题，剩下的子问题，于是就可以很好的解决。
        - 只要枚举完第二个断点的情况，然后结合子问题的成果就可以考虑完所有的点。
- 码风进步参考：
    - 00用来表示无穷。
- 上述代码中，枚举区间的时候：
    - 枚举$i,j,k,\ mid,\ i,j$时候，明确到，$[i,i+k-2]$没有解。j应该直接从更后边进行枚举。
    - 不要犯贱，从$j=i,mid=i$开始枚举
        - 小心出现无法估计的后果：比如说->溢出。
        - 比如说这个问题的背景下。
            - 由于计算的量是相乘，如果int类型下，初始化值过大，会出现溢出问题。
            - 因此通过优化枚举，来避免出现这种情况。
        - 当自己开到long,long之后，同时初始化为成一个合理的值，不会溢出。那么一样可以Acpect.