CS102

C++ 算法

动态规划

二维DP

区间DP

区间做状态

利用小区间的最优解推算大区间的最优解

例题：变形金刚

有一排n个机器人，要合体成为一整个大型变形金刚。第i个机器人重量为w[i]，共要完成n-1次合并：每次只可以选相邻两个合并，费用为此次合并的总重量。请问合并完成时，最多可能的总花费是多少？
输入第一行为正整数n。第二行为n个正整数:w[1],w[2],...,w[n]

输入样例
3
1 2 3

输入样例
6
5 6 1 5 5 5

输出样例
11

输出样例
90

区间在哪里?

思考题
贪心算法是否正确?

变形金刚

错误的贪心算法：

每次都在相邻机器人里挑选合并重量最大的进行合并

输入样例
6
5 6 1 5 5 5

输出样例
90

错误的贪心算法：先合并5,6

11+12+17+22+27

正确区间DP的最优解：先合并5,5

10+15+16+22+27

思考题
贪心算法错误的根源是什么误区？

f[i][j]：i号到j号机器全合并完的最大费用

t[i][j]：i号到j号机器的重量总和(连续和)

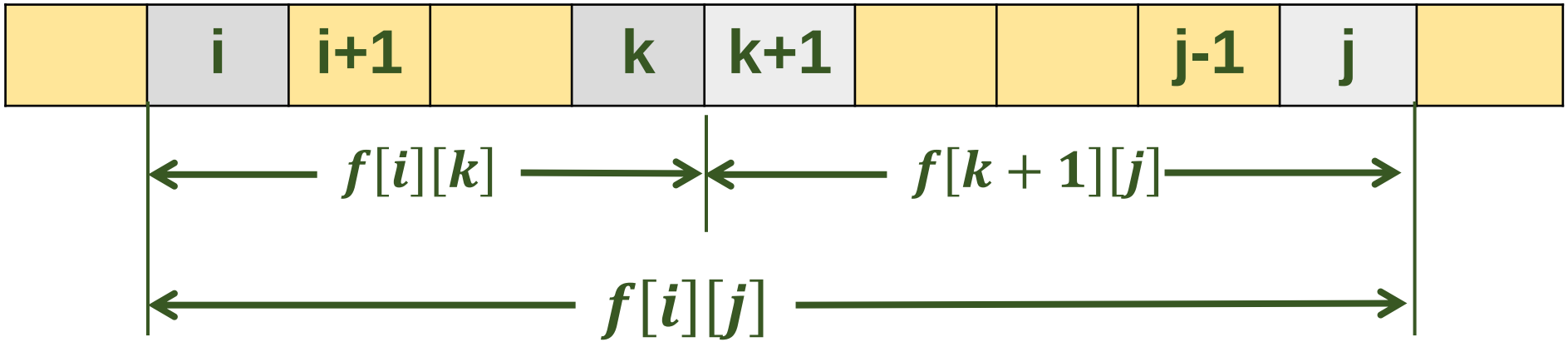
区间[i，j]

i==j	$f[i][j]=0$

边界条件

i<j	$f[i][j]=max_{k=i,i+1,\dots,j-1}\{f[i][k]+f[k+1][j]\}+t[i][j]$

状态转移方程



枚举区间[i，j]分割点k，分成左右两个区间

$ans=f[1][n]$

最终答案

f[i][j]：i号到j号机器全合并完的最大费用

t[i][j]：i号到j号机器的重量总和(连续和)

区间[i, j]

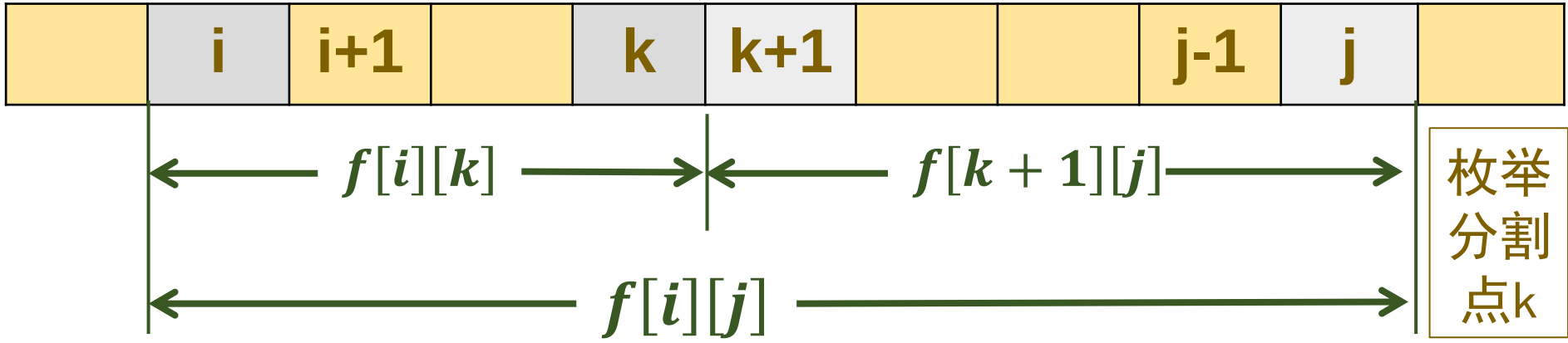
i==j	$f[i][j]=0$

边界条件

i<j	$f[i][j]=max_{k=i,i+1,\dots,j-1}\{f[i][k]+f[k+1][j]\}+s[j]-s[i-1]$

状态转移方程变形

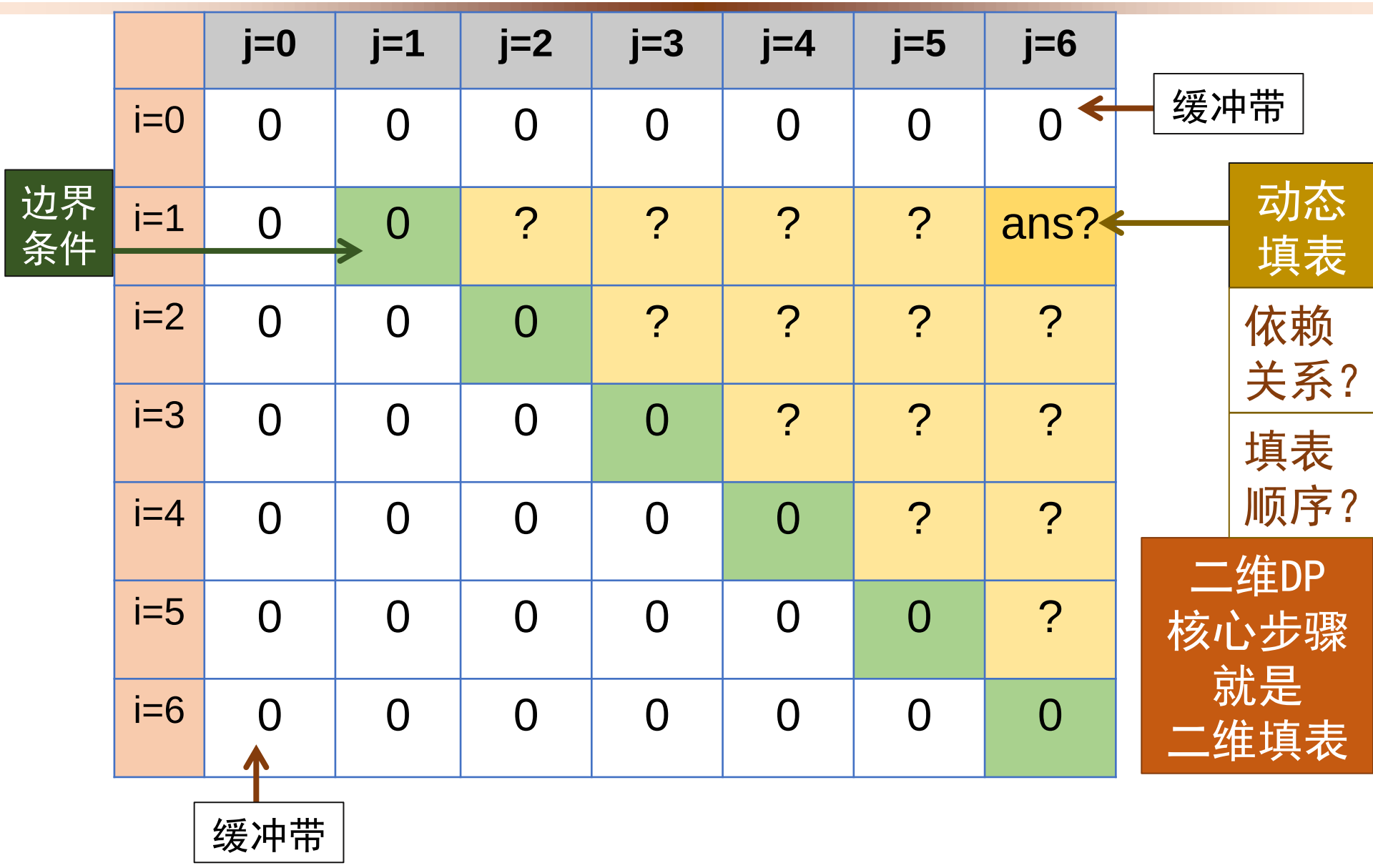
s[i]：1号到i号机器的重量总和(前缀和)



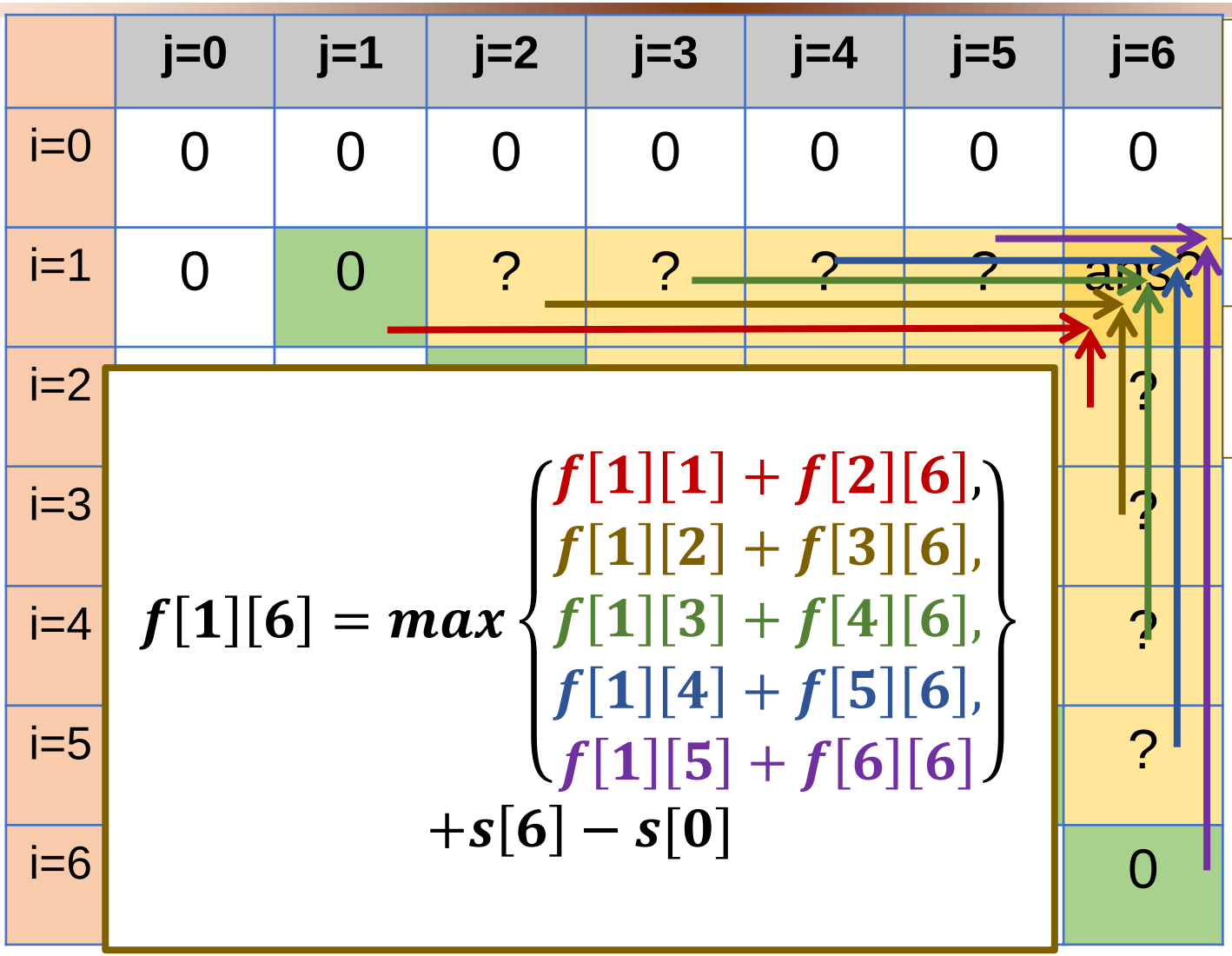
$ans=f[1][n]$

最终答案

f[i][j]：i号到j号机器全合并完的最大费用



f[i][j]：i号到j号机器全合并完的最大费用



f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：区间从小到大，依次枚举j-i=0,1,2,3,...,n-1

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	?	?	?	ans?
i=2	0	0	0	7	?	?	?
i=3	0	0	0	0	6	?	?
i=4	0	0	0	0	0	10	?
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：区间从小到大，依次枚举j-i=0,1,2,3,...,n-1

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	?	?	?
i=2	0	0	0	7	19	?	?
i=3	0	0	0	0	6	21	?
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：区间从小到大，依次枚举j-i=0,1,2,3,...,n-1

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	40	?	?
i=2	0	0	0	7	19	38	?
i=3	0	0	0	0	6	21	41
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：区间从小到大，依次枚举j-i=0,1,2,3,...,n-1

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	40	62	?
i=2	0	0	0	7	19	38	63
i=3	0	0	0	0	6	21	41
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：区间从小到大，依次枚举j-i=0,1,2,3,...,n-1

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	40	62	90
i=2	0	0	0	7	19	38	63
i=3	0	0	0	0	6	21	41
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

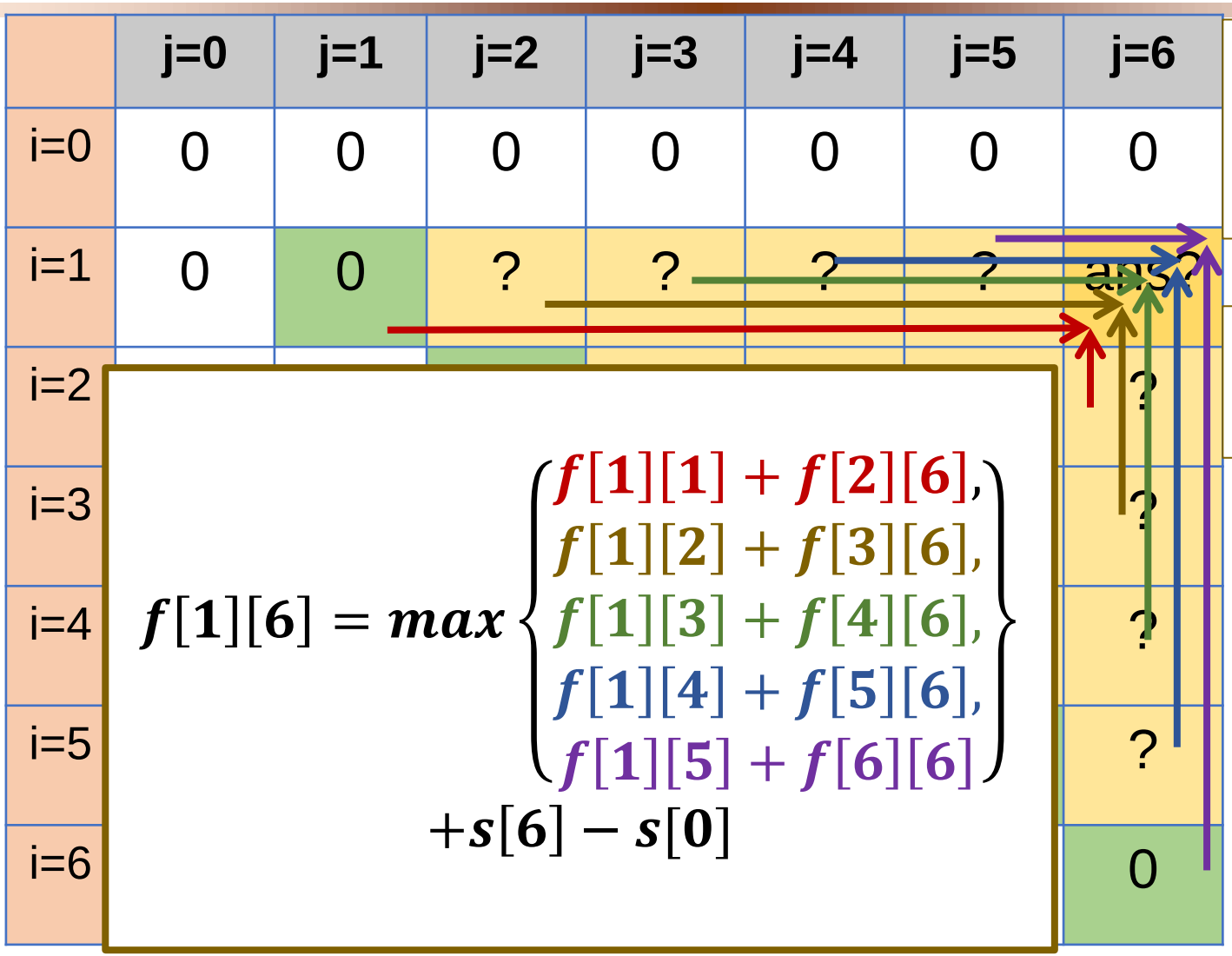
变形金刚代码1

填表顺序： 区间从小到大，依次枚举j-i=0,1,2,3,...,n-1

```
5  int n,f[N][N],w[N],s[N];
6  int main(){//f[i][j]把i到j号合并最大费用
7      cin>>n;
8      for(int i=1;i<=n;i++)cin>>w[i];
9      s[0]=0;
10     for(int i=1;i<=n;i++)s[i]=s[i-1]+w[i];
11     for(int len=1;len<=n-1;len++)
12         for(int i=1;i<=n-len;i++){
13             int j=i+len;
14             for(int k=i;k<=j-1;k++)
15                 f[i][j]=max(f[i][k]+f[k+1][j]+s[j]-s[i-1],f[i][j]);
16         }
17     cout<<f[1][n]<<endl;
```

易错点在哪里?

f[i][j]：i号到j号机器全合并完的最大费用



f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	?	?	?	?	?
i=2	0	0	0	?	?	?	?
i=3	0	0	0	0	?	?	?
i=4	0	0	0	0	0	?	?
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	?	?	?	?	?
i=2	0	0	0	?	?	?	?
i=3	0	0	0	0	?	?	?
i=4	0	0	0	0	0	10	?
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	?	?	?	?	?
i=2	0	0	0	?	?	?	?
i=3	0	0	0	0	?	?	?
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	?	?	?	?	?
i=2	0	0	0	?	?	?	?
i=3	0	0	0	0	6	?	?
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	?	?	?	?	?
i=2	0	0	0	?	?	?	?
i=3	0	0	0	0	6	21	?
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	?	?	?	?	?
i=2	0	0	0	?	?	?	?
i=3	0	0	0	0	6	21	41
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	?	?	?	?	?
i=2	0	0	0	7	?	?	?
i=3	0	0	0	0	6	21	41
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	?	?	?	?	?
i=2	0	0	0	7	19	?	?
i=3	0	0	0	0	6	21	41
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	?	?	?	?	?
i=2	0	0	0	7	19	38	?
i=3	0	0	0	0	6	21	41
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	?	?	?	?	?
i=2	0	0	0	7	19	38	63
i=3	0	0	0	0	6	21	41
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：行数从大到小，列数从小到大

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	40	62	90
i=2	0	0	0	7	19	38	63
i=3	0	0	0	0	6	21	41
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

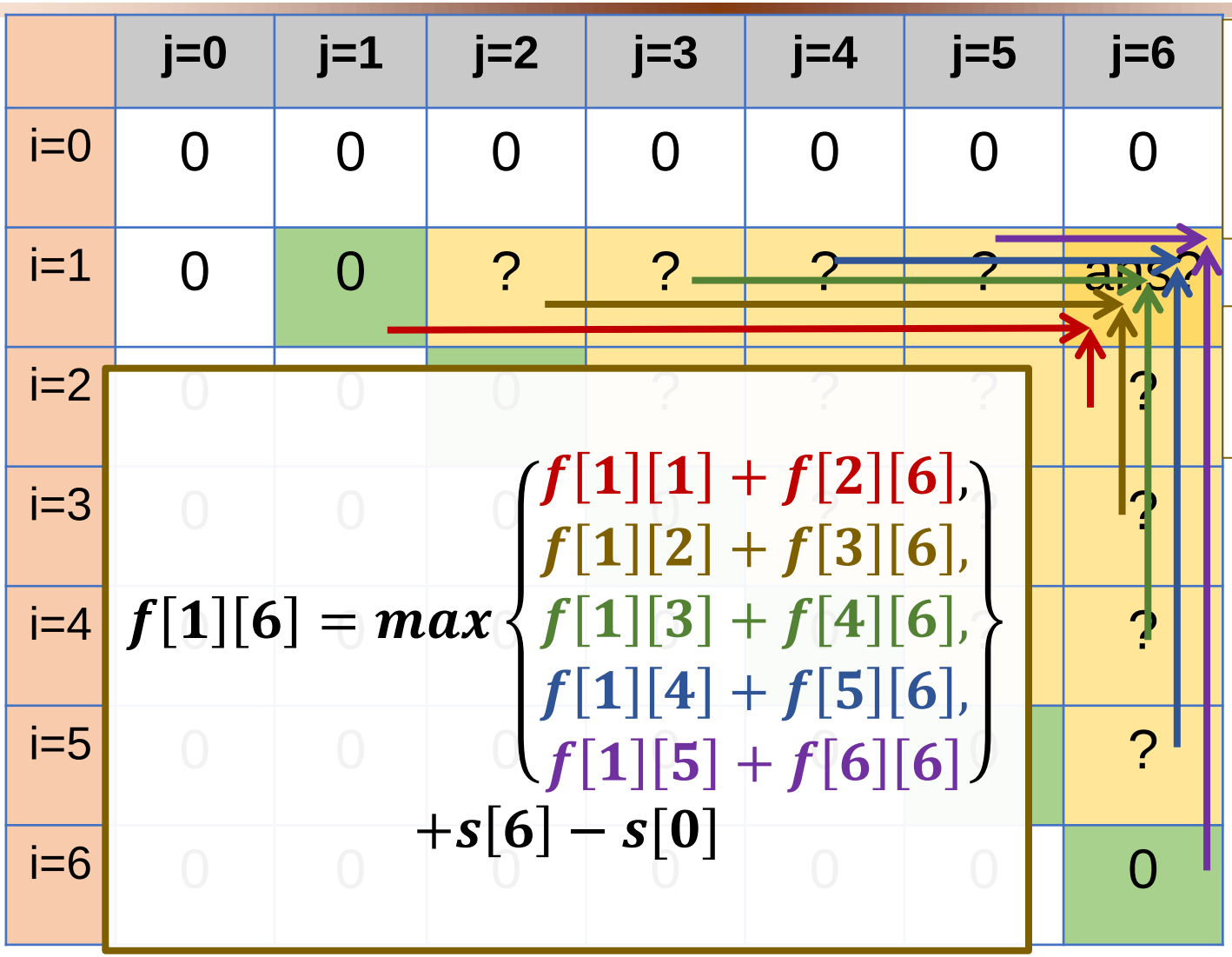
变形金刚代码2

填表顺序：行数从大到小，列数从小到大

```
5  int n,f[N][N],w[N],s[N];
6  int main(){//f[i][j]把i到j号合并最大费用
7      cin>>n;
8      for(int i=1;i<=n;i++)cin>>w[i];
9      s[0]=0;
10     for(int i=1;i<=n;i++)s[i]=s[i-1]+w[i];
11     for(int i=n-1;i>=1;i--)
12         for(int j=i+1;j<=n;j++)
13             for(int k=i;k<=j-1;k++)
14                 f[i][j]=max(f[i][k]+f[k+1][j]+s[j]-s[i-1],f[i][j]);
15     cout<<f[1][n]<<endl;
16     return 0;
17 }
```

易错点在哪里?

f[i][j]：i号到j号机器全合并完的最大费用



填每一格时只依赖左方和下方内容

填表顺序如何选择？

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：列数从小到大，行数从大到小

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	?	?	?	?
i=2	0	0	0	?	?	?	?
i=3	0	0	0	0	?	?	?
i=4	0	0	0	0	0	?	?
i=5	0	0	0	0	0	0	?
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：列数从小到大，行数从大到小

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	?	?	?	?
i=2	0	0	0	7	?	?	?
i=3	0	0	0	0	?	?	?
i=4	0	0	0	0	0	?	?
i=5	0	0	0	0	0	0	?
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：列数从小到大，行数从大到小

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	?	?	?
i=2	0	0	0	7	?	?	?
i=3	0	0	0	0	?	?	?
i=4	0	0	0	0	0	?	?
i=5	0	0	0	0	0	0	?
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：列数从小到大，行数从大到小

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	?	?	?
i=2	0	0	0	7	?	?	?
i=3	0	0	0	0	6	?	?
i=4	0	0	0	0	0	?	?
i=5	0	0	0	0	0	0	?
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：列数从小到大，行数从大到小

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	?	?	?
i=2	0	0	0	7	19	?	?
i=3	0	0	0	0	6	?	?
i=4	0	0	0	0	0	?	?
i=5	0	0	0	0	0	0	?
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：列数从小到大，行数从大到小

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	40	?	?
i=2	0	0	0	7	19	?	?
i=3	0	0	0	0	6	?	?
i=4	0	0	0	0	0	?	?
i=5	0	0	0	0	0	0	?
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：列数从小到大，行数从大到小

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	40	62	?
i=2	0	0	0	7	19	38	?
i=3	0	0	0	0	6	21	?
i=4	0	0	0	0	0	10	?
i=5	0	0	0	0	0	0	?
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

f[i][j]：i号到j号机器全合并完的最大费用

填表顺序：列数从小到大，行数从大到小

	j=0	j=1	j=2	j=3	j=4	j=5	j=6
i=0	0	0	0	0	0	0	0
i=1	0	0	11	23	40	62	90
i=2	0	0	0	7	19	38	63
i=3	0	0	0	0	6	21	41
i=4	0	0	0	0	0	10	25
i=5	0	0	0	0	0	0	10
i=6	0	0	0	0	0	0	0

i	w[i]	s[i]
1	5	5
2	6	11
3	1	12
4	5	17
5	5	22
6	5	27

变形金刚代码3

填表顺序： 列数从小到大，行数从大到小

```
5  int n,f[N][N],w[N],s[N];
6  int main(){//f[i][j]把i到j号合并最大费用
7      cin>>n;
8      for(int i=1;i<=n;i++)cin>>w[i];
9      s[0]=0;
10     for(int i=1;i<=n;i++)s[i]=s[i-1]+w[i];
11     for(int j=1;j<=n;j++)
12         for(int i=j-1;i>=1;i--)
13             for(int k=i;k<=j-1;k++)
14                 f[i][j]=max(f[i][k]+f[k+1][j]+s[j]-s[i-1],f[i][j]);
15     cout<<f[1][n]<<endl;
```

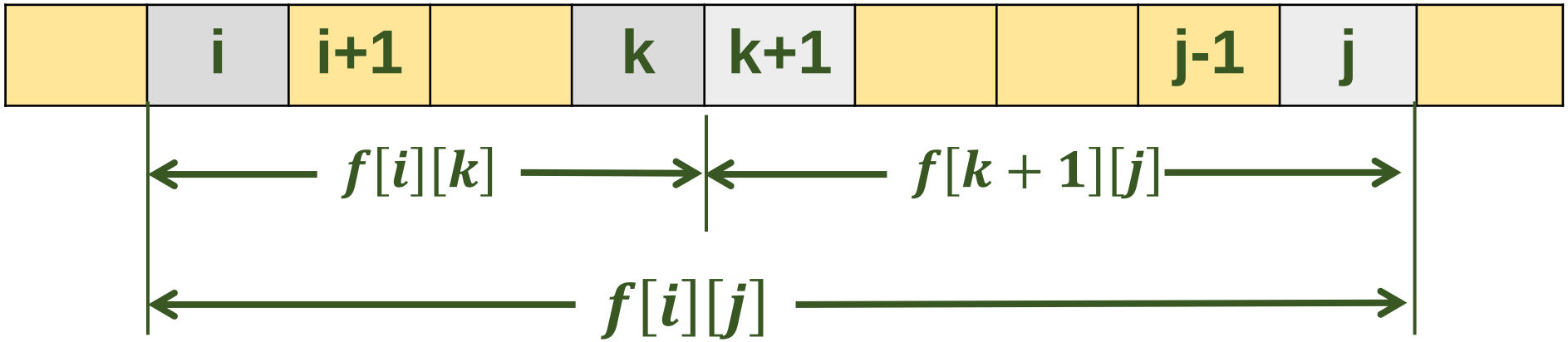
易错点在哪里?

区间DP小结

区间做状态

利用小区间的最优解推算大区间的最优解

i<j	$f[i][j] = max_{k=i,i+1,\dots,j-1}\{f[i][k] + f[k+1][j]\} + t[i][j]$	状态转移方程



例题：串珠

你家祖传有一串珠宝，串上依次共n颗珠宝。你穷困潦倒准备每年卖1颗，每次珠宝只能从头部或尾部拿出来卖。第i颗珠宝的品质为v[i]。
随着年份增长，售价也会增长：如果第a年卖出第i颗，那么售价为v[i]*a。请问卖完这串珠宝最多赚多少钱？

输入样例
4
12 1 11 10

输出样例
88

输入第一行为正整数n
输入第二行为n个正整数:v[1],v[2],...,v[n]

区间在哪里?

思考：贪心算法是否正确?

例题：串珠

错误的贪心算法：

每次都从头部和尾部的两个珠宝里挑较便宜的去卖

输入样例
4
12 1 11 10

输出样例
88

错误的贪心算法：先卖10

10*1+11*2+1*3+12*4=83

正确区间DP的最优解：先卖12

12*1+1*2+10*3+11*4=88

思考题
贪心算法错误的根源是什么误区？

f[i][j]：i号到j号珠宝最大总售价

v[i]：i号珠宝的品质

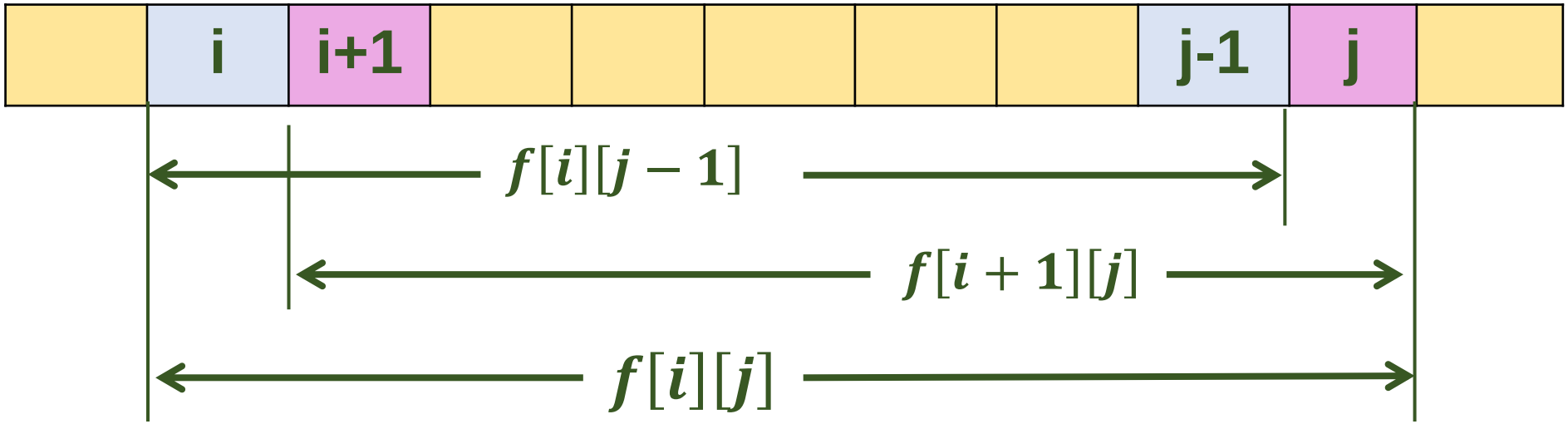
区间[i, j]

i==j	$f[i][j] = v[i] \times n$

边界条件

i<j	$f[i][j] = max\begin{Bmatrix} f[i+1][j] + v[i] \times (n-j+i), \\ f[i][j-1] + v[j] \times (n-j+i) \end{Bmatrix}$

状态转移方程



$ans = f[1][n]$

最终答案

串珠代码

```
6  int n,f[N][N],v[N];
7  int main(){//f[i][j]拿i到j位最多几元
8      cin>>n;
9      for(int i=1;i<=n;i++)cin>>v[i];
10     for(int i=1;i<=n;i++)f[i][i]=v[i]*n;
11     for(int i=n-1;i>=1;i--)
12         for(int j=i+1;j<=n;j++)
13             f[i][j]=max(f[i][j-1]+v[j]*(n-j+i),f[i+1][j]+v[i]*(n-j+i));
14     cout<<f[1][n]<<endl;
```

易错点在哪里?

例题：添加括号

你的C++程序编译报错了，提示圆括号和方括号没有正确配对。按顺序，你依次找到所有圆括号和方括号，请问至少需要添加几个括号才能正确配对。总长度<=100
注意：此题括号没有等级区别，例如([])和[()]均合法

输入样例 [	输入样例)][(	输入样例 ([(]	输入样例 ([)]	输入样例 []()
输出样例 1	输出样例 4	输出样例 2	输出样例 2	输出样例 0

输入样例 ()()

输出样例 0

区间在哪里?

f[i][j]：第i到j号字符配对需要添加几个括号

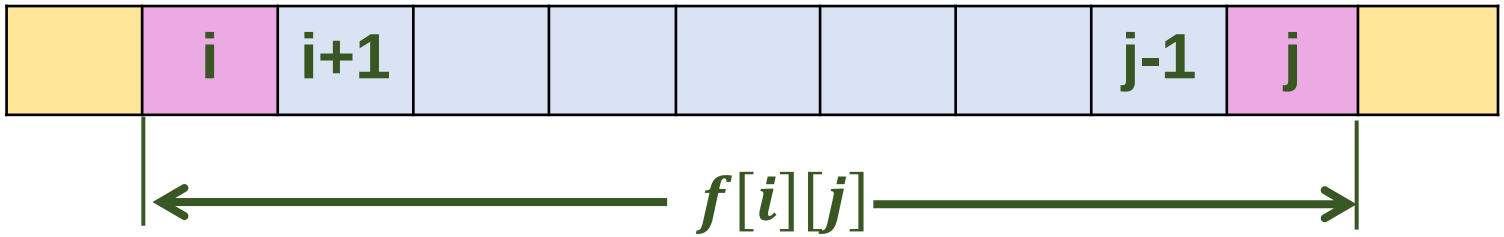
区间[i, j]

s[i]：第i号字符

i==j	$f[i][j]=1$
j-i==1	$f[i][j]=$取决于$s[i]$和$s[j]$是否配对

边界条件

j-i>=2, $s[i]$和$s[j]$不配对	$f[i][j]=min_{k=i,i+1,..,j-1}\{f[i][k]+f[k+1][j]\}$
j-i>=2, $s[i]$和$s[j]$配对	$f[i][j]=min\left\{\begin{matrix} f[i+1][j-1], \\ min_{k=i,i+1,..,j-1}\{f[i][k]+f[k+1][j]\} \end{matrix}\right\}$



状态转移方程

$ans=f[0][n-1]$

最终答案

添加括号代码

```
6  string s;
7  int n,f[N][N];
8  bool match(char a,char b){
9      return a=='('&&b==')'||a=='['&&b==']';
10 }
11 int main(){//f[i][j]将i到j位配对至少插入几个
12     cin>>s;
13     n=s.size();
14     fill(f[0],f[0]+N*N,INF);
15     for(int i=0;i<n;i++)f[i][i]=1;
16     for(int i=0;i<n-1;i++)
17         if(match(s[i],s[i+1]))f[i][i+1]=0;
18         else f[i][i+1]=2;
19     for(int i=n-3;i>=0;i--)
20         for(int j=i+2;j<=n-1;j++){
21             for(int k=i;k<=j-1;k++)
22                 f[i][j]=min(f[i][k]+f[k+1][j],f[i][j]);
23             if(match(s[i],s[j]))
24                 f[i][j]=min(f[i+1][j-1],f[i][j]);
25         }
26     cout<<f[0][n-1]<<endl;
```

易错点在哪里?

课件下载链接：
链接：https://pan.baidu.com/s/1ei7f7w
密码：q66i

作业网站：
http://120.132.18.213:8080/thrall-web/main#home