B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Incorrect solution **for partitioning all the chocolate**

1. Compute the number of bars of each bar size (there are 21 sizes).

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Incorrect solution **for partitioning all the chocolate**

1. Compute the number of bars of each bar size (there are 21 sizes).
2. Always greedily pair up bars so that you never have 2 or more of any size.

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Incorrect solution **for partitioning all the chocolate**

1. Compute the number of bars of each bar size (there are 21 sizes).
2. Always greedily pair up bars so that you never have 2 or more of any size.
3. Find minimum number of breaks needed for remaining bars.

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Incorrect solution **for partitioning all the chocolate**

1. Compute the number of bars of each bar size (there are 21 sizes).
2. Always greedily pair up bars so that you never have 2 or more of any size.
3. Find minimum number of breaks needed for remaining bars.
4. Can be computed efficiently by dynamic programming over 2^{21} states.

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Incorrect solution **for partitioning all the chocolate**

1. Compute the number of bars of each bar size (there are 21 sizes).
2. Always greedily pair up bars so that you never have 2 or more of any size.
3. Find minimum number of breaks needed for remaining bars.
4. Can be computed efficiently by dynamic programming over 2^{21} states.
5. Does not always give the optimum number of breaks. Example:

   `3x2 3x3 1x5 2x5 3x5 3x5`

   Split one 3x5 as `3x2+3x3` and the other as `1x5+2x5` to get away with two splits.

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Incorrect solution **for partitioning all the chocolate**

1. Compute the number of bars of each bar size (there are 21 sizes).
2. Always greedily pair up bars so that you never have 2 or more of any size.
3. Find minimum number of breaks needed for remaining bars.
4. Can be computed efficiently by dynamic programming over 2^{21} states.
5. Does not always give the optimum number of breaks. Example:

   3x2 3x3 1x5 2x5 3x5 3x5

   Split one 3x5 as 3x2+3x3 and the other as 1x5+2x5 to get away with two splits.
6. But this *does* give upper bound on number of breaks that may be needed (it is 9).

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Actual Solution

1. Recursively search for the best way to break the bars, going from larger bars to smaller ones.

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Actual Solution

1. Recursively search for the best way to break the bars, going from larger bars to smaller ones.
2. Keep track of number of breaks made and how many squares can be partitioned among the larger bars already considered.

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Actual Solution

1. Recursively search for the best way to break the bars, going from larger bars to smaller ones.
2. Keep track of number of breaks made and how many squares can be partitioned among the larger bars already considered.
3. Avoid recursive calls that will produce more breaks than best solution found.

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Actual Solution

1. Recursively search for the best way to break the bars, going from larger bars to smaller ones.
2. Keep track of number of breaks made and how many squares can be partitioned among the larger bars already considered.
3. Avoid recursive calls that will produce more breaks than best solution found.
4. Since there is no need for more than 9 breaks and any bar be broken in at most 5 different ways, this turns out to be fast enough.

B — Breaking Bars

Problem

Given list of rectangular chocolate bars, make as few breaks as possible to produce two equal collections of bars, each collection having at least t squares.

Actual Solution

1. Recursively search for the best way to break the bars, going from larger bars to smaller ones.
2. Keep track of number of breaks made and how many squares can be partitioned among the larger bars already considered.
3. Avoid recursive calls that will produce more breaks than best solution found.
4. Since there is no need for more than 9 breaks and any bar be broken in at most 5 different ways, this turns out to be fast enough.

Statistics at 4-hour mark: 17 submissions, 1 accepted, first after 01:13