Идея решения

Для решения задачи можно просто перебрать все возможные варианты. Очевидно, что если в рейку вбито некоторое количество гвоздей, то количество промежутков между гвоздями (а следовательно и веревочек) будет на единицу меньше. Кроме длины веревочки для решения важно знание того факта, есть ли в данном промежутке веревочка или нет.

Пусть в рейку вбито M гвоздей, тогда максимально возможное количество привязанных веревочек – M-1. Поэтому рейку удобно представить в виде двумерного массива размерностью 2 x (M-1). Элементы первой строки этого массива содержат длины промежутков между гвоздями, а элементы второй строки показывают наличие или отсутствие веревочки в промежутке с соответствующим номером. Если веревочка в промежутке есть, то соответствующий этому промежутку элемент второй строки равен 1, если нет – 0.

Для решения задачи необходимо перебрать все варианты расположения веревочек. Для этого можно представить вторую строку как двоичное число (1 - веревочка есть, 0 – веревочки нет), и тогда задача сведется к перебору всех таких чисел длины (M-1). Перебор представляет собой циклический процесс прибавления к двоичному числу единицы по правилам двоичной арифметики до тех пор, пока все разряды двоичного числа не окажутся заполненными единицами.

Таким образом, алгоритм представляет собой цикл построения двоичных чисел, начиная с нулевого, на каждом шаге которого проверяется, подходит ли данное расположение веревочек к условию задачи. Для этого достаточно убедиться, что первая и последняя цифры двоичного числа не равны 0 (т.е. к первому и последнему гвоздям привязаны веревочки) и что в записи числа не содержатся два идущих подряд нуля (т.е. нет гвоздей, к которым ничего не привязано).

Если расположение веревочек удовлетворяет условию задачи, то остается только посчитать сумму длин тех промежутков, где есть веревочки и сравнить эту сумму с каким-либо значением (обозначим как MIN), для которого заведомо найдется меньшее или равное. Например, в качестве этого значения можно использовать сумму длин абсолютно всех промежутков между гвоздями. Если длина веревочек при рассматриваемой расстановке меньше MIN, то необходимо сохранить этот вариант положения веревочек в массив, а MIN приравнять к найденной сумме. Эти операции необходимо повторять для всех прочих случаев положения веревочек. Таким образом, в конце перебора MIN будет равно минимальной удовлетворяющей условию сумме длин веревочек, а в массиве B будет сохранено расположение веревочек, соответствующее этой сумме.

Вариант решения на языке C.

```
#include "stdafx.h"
```

```
#include <conio.h>
#include <iostream.h>
int condition(int a[2][19], int m) // Проверка двоичного числа на случай заполнения всех разрядов единицами
{
	for (int j=0; j<=m;j++)
	{
		if (a[1][j]!=1) return 0;
	}
	return 1;
}
int check(int a[2][19], int m)
{
	for (int j=0; j<=(m-1); j++)
	{
		if ((a[1][j]==0) && (a[1][j+1]==0))  // Проверка на наличие двух идущих подряд нулей
		{
			return 1;
		}
	}
	if ((a[1][0]==0)|| (a[1][m]==0)) return 1;//Проверка на наличие веревочек возле первого и последнего гвоздей
	return 0;
}
void output(int a[2][19], int m)
{
	for (int j=0; j<=m; j++)
	{
		cout<<a[1][j]<<" ";
	}
	cout<<"\n";
}
void main()
{
	int m,sum,b[2][19];
	int min=0;
	int a[2][19];
	cin>>m;
	for (int j=0; j<=(m-2);j++)
	{
		cin>>a[0][j];
		a[1][j]=0;
		min=min+a[0][j];  //Подсчет первоначального значения MIN
	}
```

```
	a[1][m-2]=-1;
	do	//Основной цикл
	{
		sum=0;
		a[1][m-2]++;
		for (int j=(m-2); j>=0;j--) //Построение двоичных чисел
		{
			if (a[1][j]==2)
			{
				a[1][j]=0;
				a[1][j-1]++;
			}
		}
		if (check(a,m-2) == 0) //Подсчет суммы длин веревочек в случае
выполнения условий задачи
		{
			for (int j=0; j<=(m-2); j++)
			{
				sum+=a[0][j]*a[1][j];
			}
			if (sum <= min) //Сохранение минимальной суммы и
соответствующего положения веревочек
			{
				min=sum;
				for (int j=0; j<=(m-2); j++)
				{
					b[1][j]=a[1][j];
					b[0][j]=a[0][j];
				}
			}
		}
	}
	while (condition(a,m-2)!=1);
	output(b,m-2);
	cout<<"Summa = " <<min<<"\n";
}
```

Вариант решения на языке Компонентный Паскаль

```
MODULE Zadacha3;
IMPORT In, StdLog;
TYPE
	mass = ARRAY 2 OF ARRAY 19 OF INTEGER;
```

```
PROCEDURE condition(a: mass; m: INTEGER): INTEGER;  (*Проверка двоичного числа на случай заполнения всех разрядов единицами*)
VAR
	j,flag: INTEGER;
BEGIN
	flag:=1;
	FOR j := 0 TO m DO
		IF a[1,j] # 1 THEN
			flag:=0;
		END;
	END;
	RETURN flag;
END condition;
PROCEDURE check(a:mass; m: INTEGER): INTEGER;
VAR
	j,flag: INTEGER;
BEGIN
	flag := 0;
	FOR j := 0 TO m-1 DO
		IF ((a[1,j] = 0) & (a[1,j+1]=0)) THEN		(*Проверка на наличие двух идущих подряд нулей*)
			flag := 1;
		END;
	END;
	IF (a[1,0]=0) OR (a[1,m]=0) THEN		(*Проверка на наличие веревочек возле первого и последнего гвоздей*)
		flag:=1;
	END;
	RETURN flag;
END check;

PROCEDURE output(a:mass;m: INTEGER);
VAR
	i,j: INTEGER;
BEGIN
		FOR j := 0 TO m DO
			StdLog.Int(a[1,j]);
			StdLog.String('  //  ');
		END;
END output;

PROCEDURE main*;
VAR
	a,b:mass;
	j,m,sum,min: INTEGER;
```

```
BEGIN
	In.Open;
	In.Int(m);
	FOR j := 0 TO m-2 DO
		In.Int(a[0,j]);
		a[1,j] :=0;
		min:=min+a[0,j];	(*Подсчет первоначального значения MIN*)
	END;
	a[1,(m-2)] := -1;
	REPEAT				(*Основной цикл*)
		sum:=0;
		INC(a[1,(m-2)]);
			FOR j := (m-2) TO 0 BY -1 DO (*Построение двоичных
чисел*)
				IF a[1,j] = 2 THEN
					a[1,j]:=0;
					INC(a[1,j-1]);
				END;
			END;
		IF check(a, m-2) = 0 THEN	(*Подсчет суммы длин веревочек в
случае выполнения условий задачи*)
			FOR j := 0 TO (m-2) DO
				sum:=sum+a[0,j] * a[1,j];
			END;
				IF sum <= min THEN		(*Сохранение
минимальной суммы и соответствующего положения веревочек*)
					min:=sum;
					FOR j := 0 TO (m-2) DO
						b[1,j] := a[1,j];
						b[0,j] := a[0,j];
					END;
				END;
		END;
	UNTIL condition(a, m-2) = 1;
	output(b, m-2);
	StdLog.String('summa');
	StdLog.Int(min);
END main;

END Zadacha3.
```