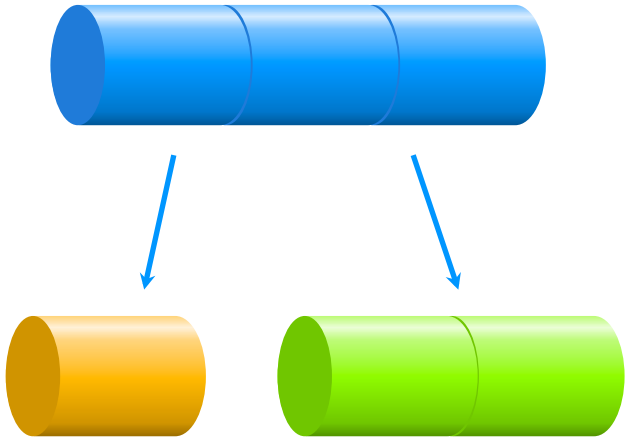
Rod Cutting

with dynamic programming



```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

We compute the maximum value of pieces cut from a rod of length 4.

length	0	1	2	3	4

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to
    q = -∞
    for i = 1 to
      if q < p[i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

p is the list of prices for a single rod of each length from 0 to 4.

length	0	1	2	3	4
p	0	1	5	8	9

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

r[i] is the maximum value we can get from cutting a rod of length i.

length	0	1	2	3	4
p	0	1	5	8	9
r	0				

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 1

We start by finding the max value for cutting rods of length 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0				

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i]
        q = p[i] +
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 1
q: -∞

q is the maximum value we have seen so far for cutting a rod of length j.

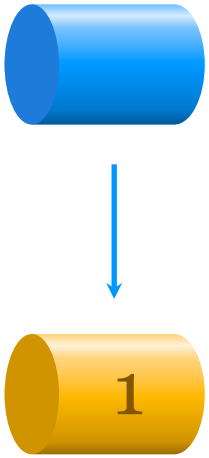
length	0	1	2	3	4
p	0	1	5	8	9
r	0				

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
  re
```

j: 1
q: -∞
i: 1

We will find the best value that can result when the first rod we cut has length 1.

length	0	1	2	3	4
p	0	1	5	8	9
r	0				

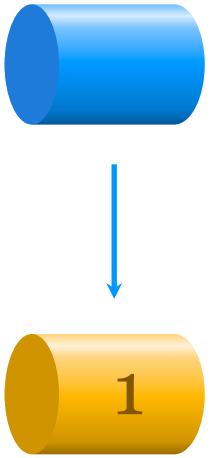


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
re
```

j: 1
q: -∞
i: 1

For the rod of length 1 this seems pointless but later on we'll see why it's important.

length	0	1	2	3	4
p	0	1	5	8	9
r	0				



```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 1
q: -∞
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0				
s					

"Is the value of a rod of length 1 plus the maximum value of cutting a rod of length 0 greater than q?"

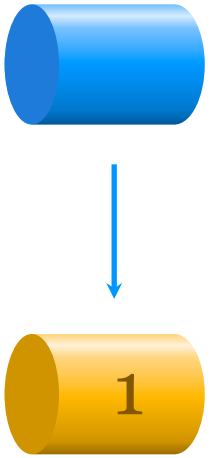
q is the previous max value.

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Update the previous max value.

j: 1
q: 1
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0				
s					

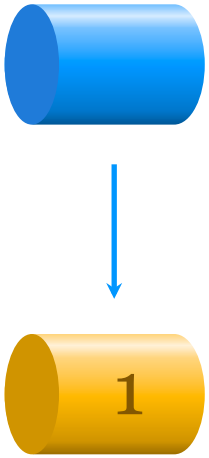


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to
    q = -∞
    for i = 1
      if q < p[
        q = p[i]  [j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

The best value for rods of length 1 comes when we start by cutting off a rod of length 1.

j: 1
q: 1
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0				
s		1			



```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 1
q: 1
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1			

The maximum value we can get from cutting a rod of length 1 is 1.

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] +
        q = p[i] +
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 2
q: 1
i: 1

Now we'll find the max value for cutting a rod of length 2.

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1			

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 2
q: -∞
i: 1

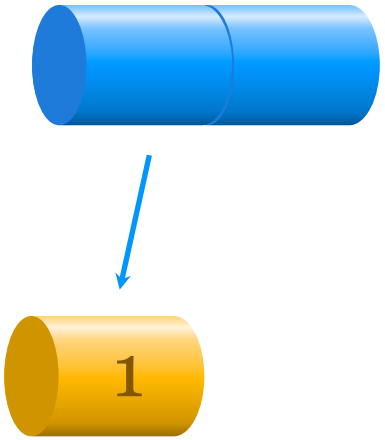
length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1			

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Start by cutting off a piece of length 1.

j: 2
q: -∞
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1			

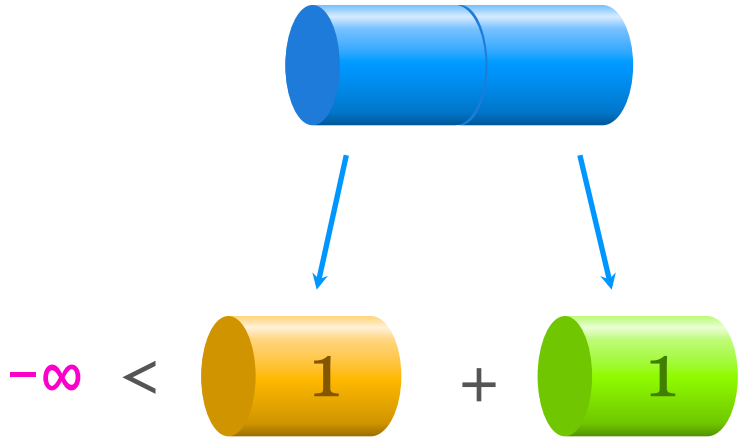


Is the value of a piece of length 1 plus the maximum value of the remaining portion greater than the previous max value?

j: 2
q: -∞
i: 1

```
for i = 1 to j
    if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
  r[j] = q
return [r,s]
```

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1			

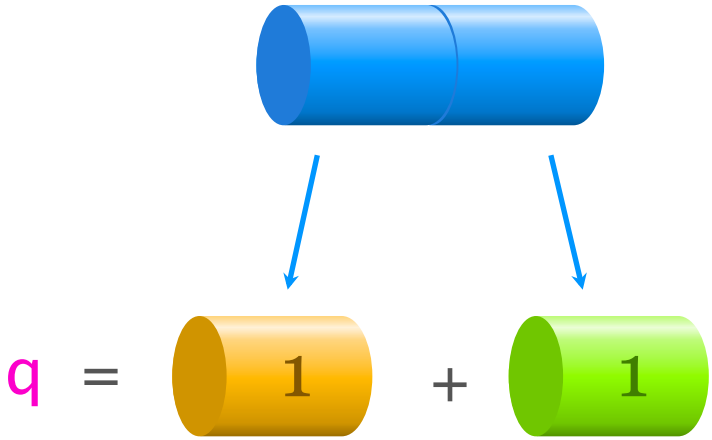


```
EBUcutRod(p
  r[0] = 0
  for j = 1
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Two is the best value we have seen *so far* for cutting a rod of length 2.

j: 2
q: 2
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1			

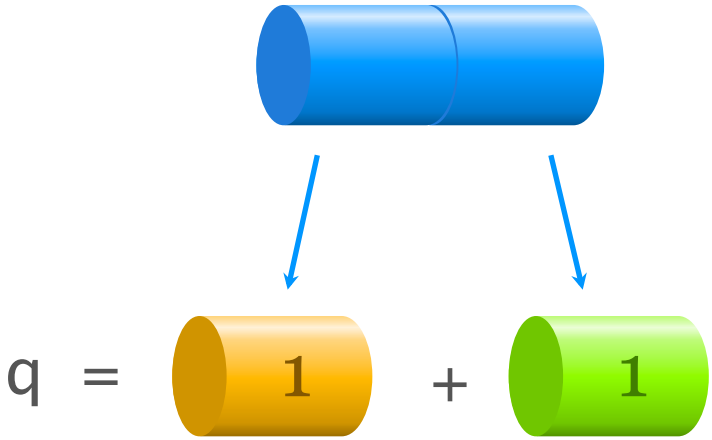


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to
    q = -∞
    for i = 1
      if q < p[
        q = p[i] +    [j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

The maximum value we have seen for cutting a rod of length 2 was achieved when we started by cutting off a section of length 1.

j: 2
q: 2
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1	1		

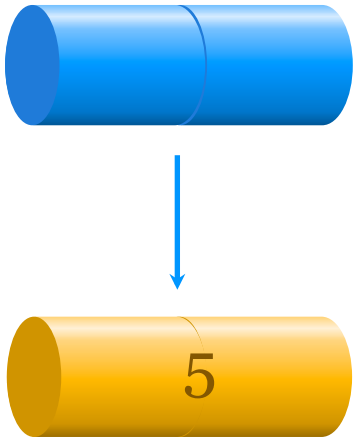


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Now we'll see what happens if we start by cutting a piece of length 2.

j: 2
q: 2
i: 2

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1	1		

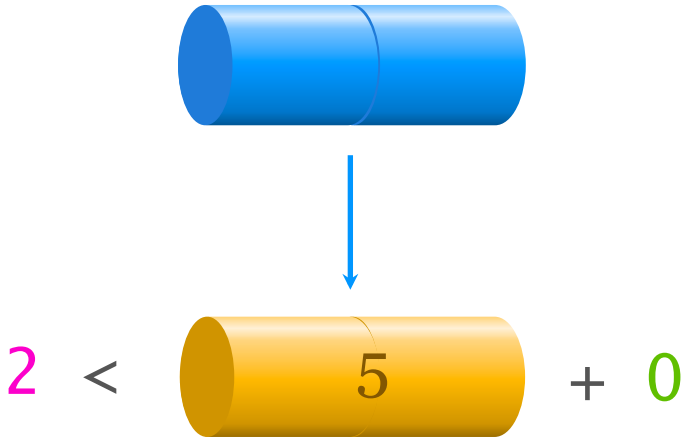


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 
    q = -∞
    for i = 1 to 
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Is this better than our previous max result?

j: 2
q: 2
i: 2

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1	1		

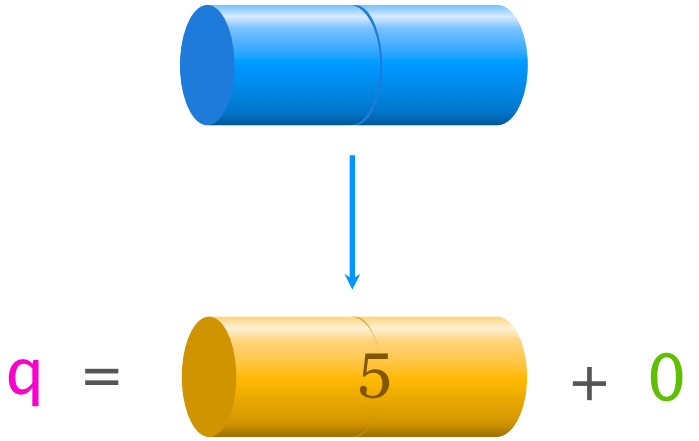


```
EBUcutRod(p
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

It is better so we update q.

j: 2
q: 5
i: 2

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1	1		



```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 2
q: 5
i: 2

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1			
s		1	2		

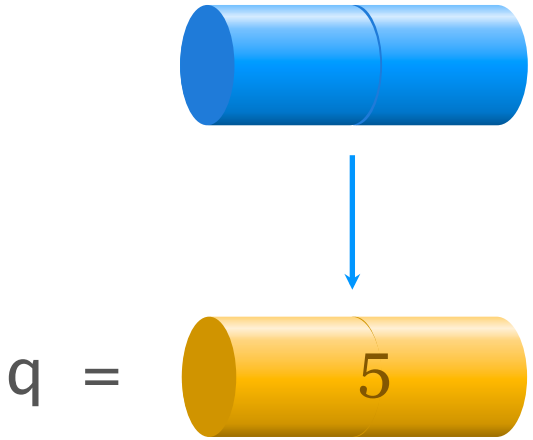
The maximum value we have seen for cutting a rod of length 2 was achieved when we started by cutting off a section of length 2.

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q <
        q =
        s[j] =
    r[j] = q
  return [r,s]
```

j: 2
q: 5
i: 2

The maximum value of cutting a rod of length 2 is 5.

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2		



```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1
      if q <
        q = p[·] · ·[j ·]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 3
q: 5
i: 2

Now we'll find the max value for cutting a rod of length 3.

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2		

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 3
q: -∞
i: 2

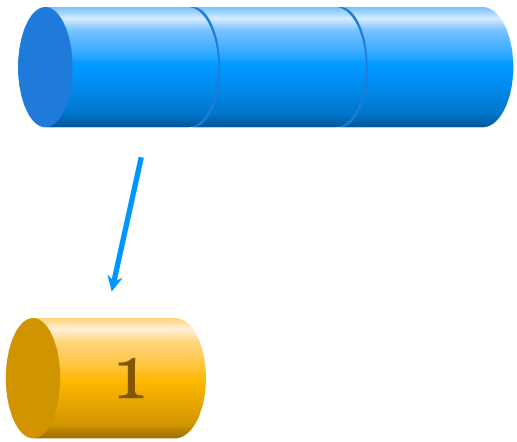
length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2		

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Begin by cutting off a section of length 1.

j: 3
q: -∞
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2		

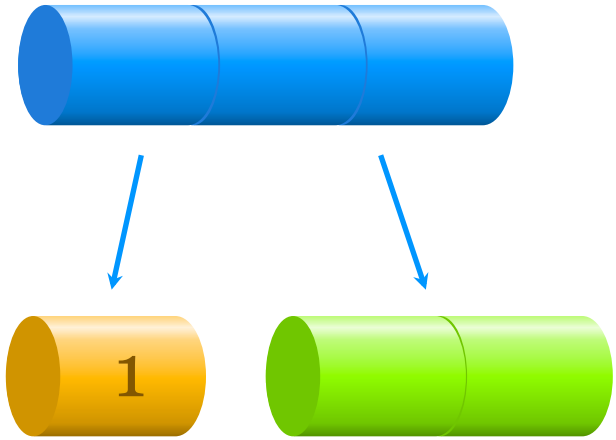


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

After we cut off a section of length 1, the remaining piece has length 2.

j: 3
q: -∞
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2		

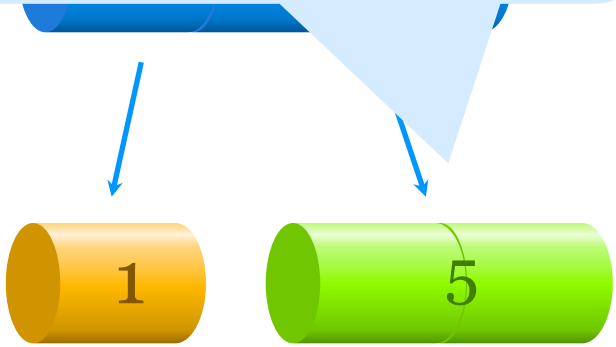


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 3
q: -∞
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2		

We already know that the max value for the remaining piece is 5.

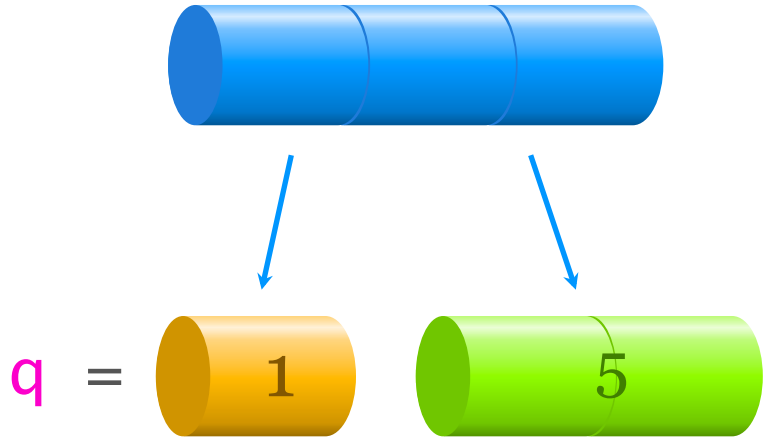


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Update our previous max value.

j: 3
q: 6
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2		

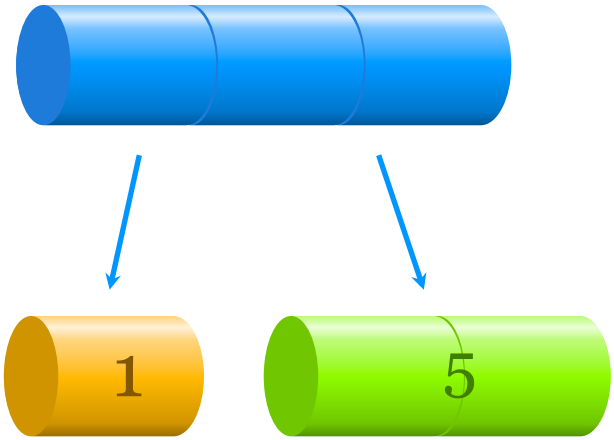


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to
    q = -∞
    for i = 1
      if q < p
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 3
q: 6
i: 1

The highest value we have seen for cutting rods of length 3 came when we started by cutting off a section of length 1.

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2	1	

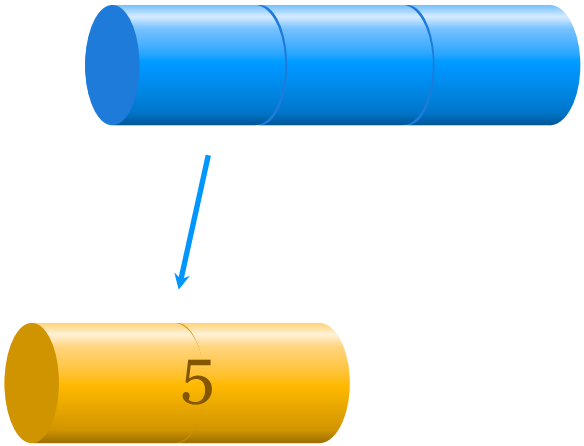


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Now, we try starting by cutting a piece of length 2.

j: 3
q: 6
i: 2

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2	1	

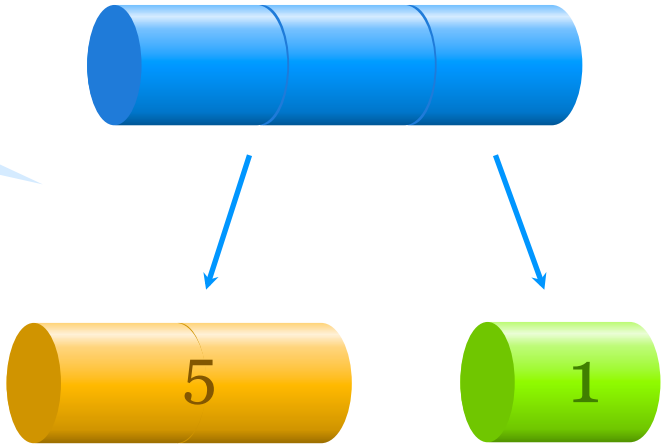


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
  r
```

j: 3
q: 6
i: 2

This is not better than what we got when we started with the rod of length 1.

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2	1	

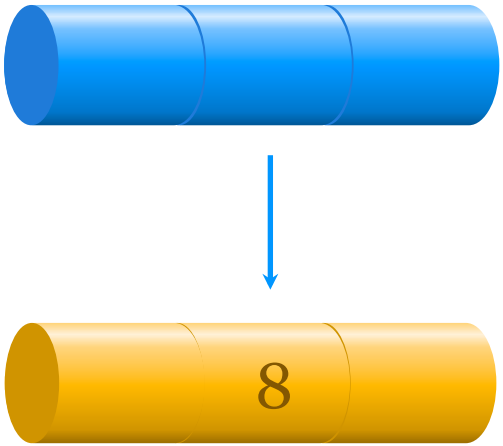


```
EBUcutRod(p
  r[0] = 0
  for j = 1
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Now we'll try using the entire rod of length 3.

j: 3
q: 6
i: 3

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2	1	

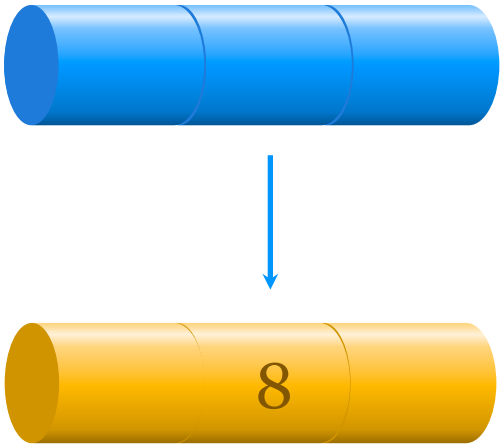


```
EBUcutRod(p
  r[0] = 0
  for j = 1
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Is one rod of length 3 better than a rod of length 1 and a rod of length 2?

j: 3
q: 6
i: 3

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2	1	

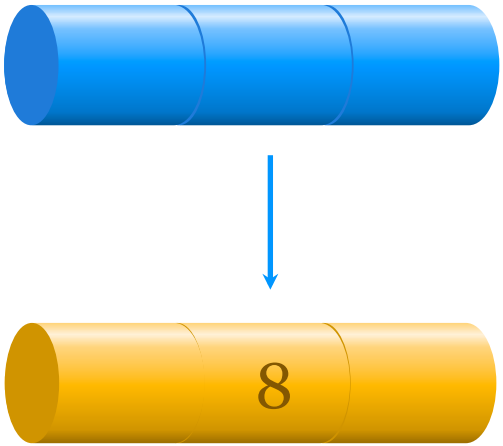


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

It is better, so we update our max value for rods of length 3.

j: 3
q: 8
i: 3

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2	1	



```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 3
q: 8
i: 3

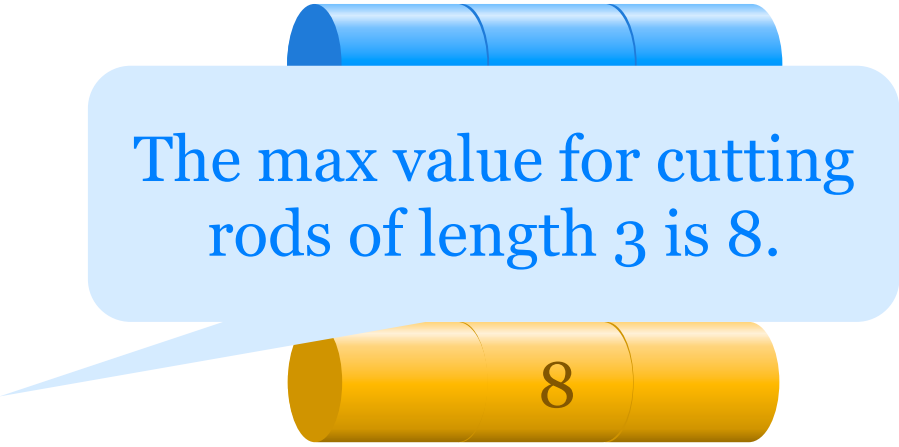
length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5		
s		1	2	3	

The max value for rods of length 3 comes when we start by cutting a rod of length 3.

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 3
q: 8
i: 3

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	



```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q <
        q = p
        s[j]
    r[j] = q
  return [r,s]
```

j: 4
q: 8
i: 3

Now we consider rods of length 4.

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

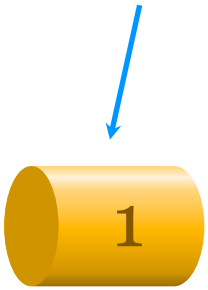
j: 4
q: -∞
i: 3

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 4
q: -∞
i: 1

Start with a rod of length 1.

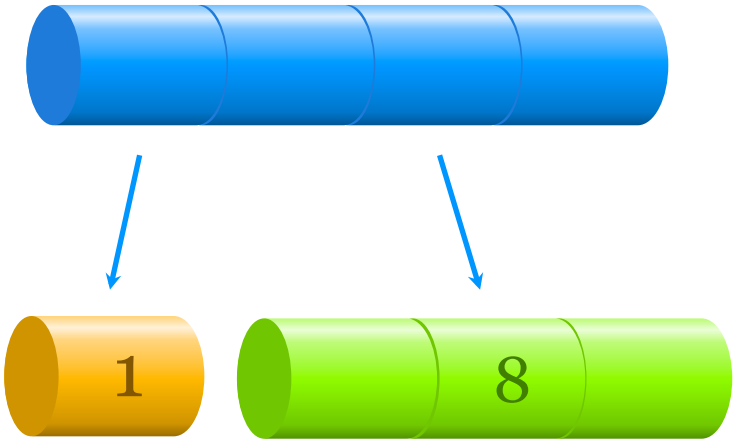


length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 4
q: -∞
i: 1

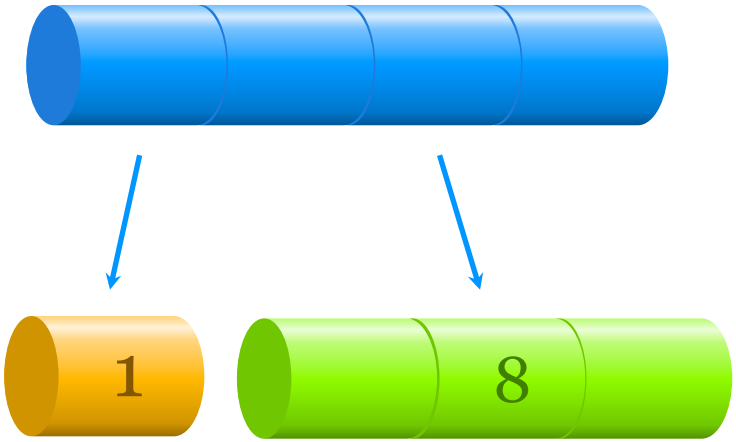
length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	



```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 4
q: 9
i: 1

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	



```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 4
q: 9
i: 1

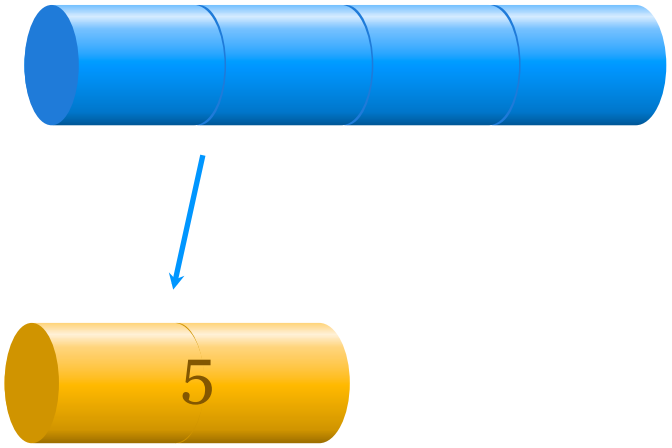
length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	1

The best value we have seen for cutting rods of length 4 came when we started with a rod of length 1.

```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 4
q: 9
i: 2

length	0	1	2	3	4	5
p	0	1	5	8	9	10
r	0	1	5	8		
s		1	2	3	1	

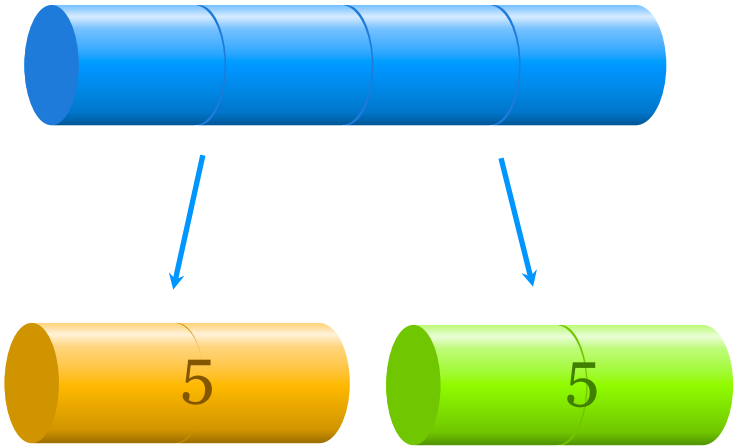


```
EBUcutRod(p,4
  r[0] = 0
  for j = 1 t
    q = -∞
    for i = 1 t  j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Are two rods of length 2 better than a rod of length 1 and another of length 3?

j: 4
q: 9
i: 2

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	1

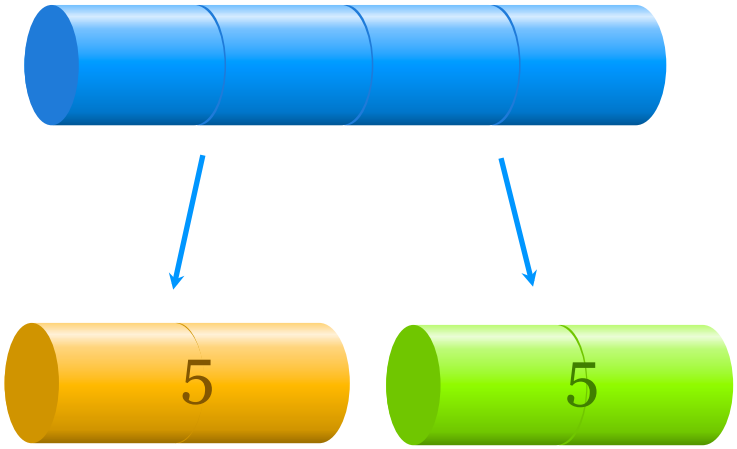


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 t
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

They are better so we update q.

j: 4
q: 10
i: 2

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	1

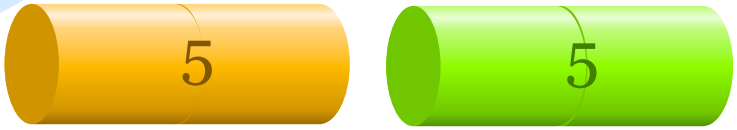


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 4
q: 10
i: 2

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	2

The best value we have seen for cutting rods of length 4 came when we started with a rod of length 2.

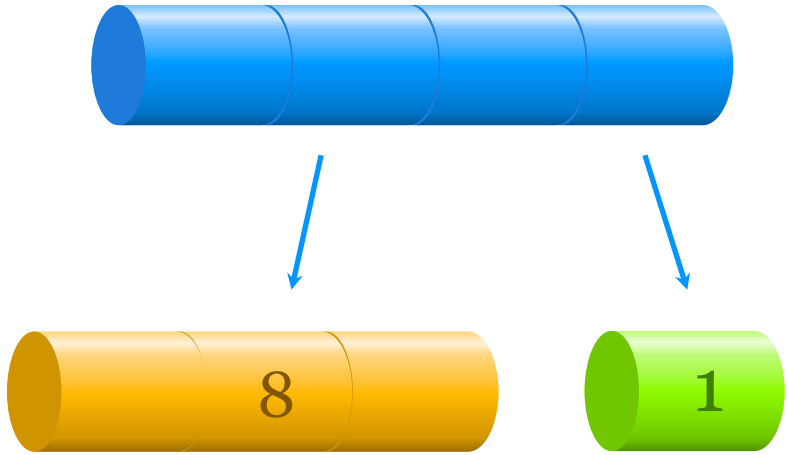


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

This is not better than our previous best result.

j: 4
q: 10
i: 3

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	2

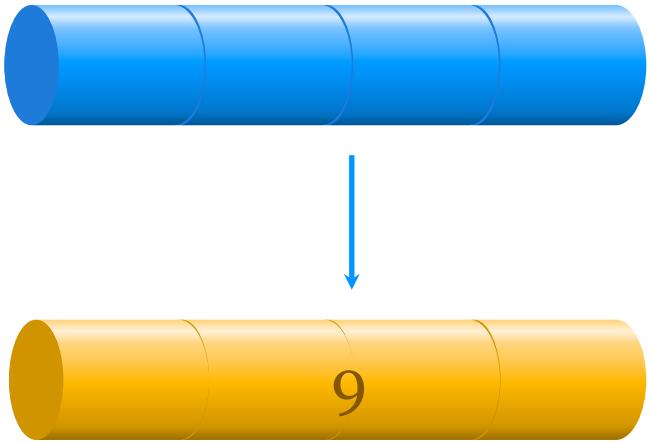


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

Finally, we try keeping the whole rod of length 4.

j: 4
q: 10
i: 4

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	2

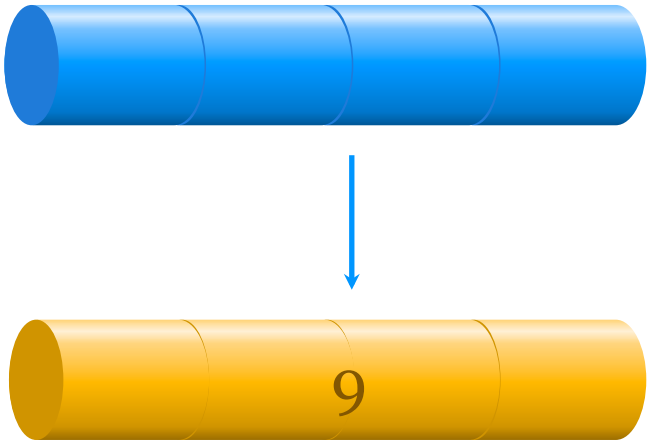


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 t
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

This is also not better than our previous best result.

j: 4
q: 10
i: 4

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	
s		1	2	3	2

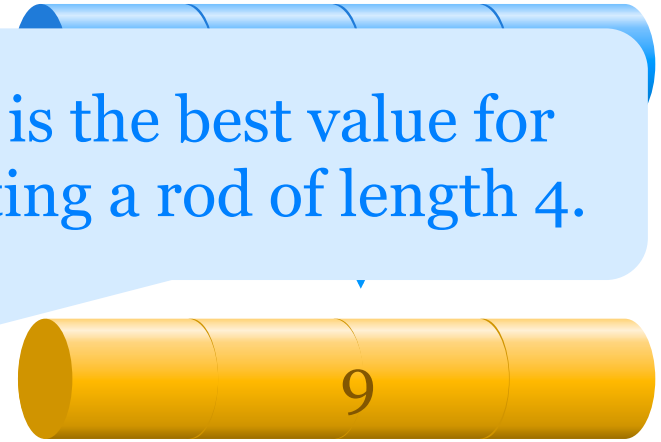


```
EBUcutRod(p,4)
  r[0] = 0
  for j = 1 to n
    q = -∞
    for i = 1 to j
      if q < p[i] + r[j-i]
        q = p[i] + r[j-i]
        s[j] = i
    r[j] = q
  return [r,s]
```

j: 4
q: 10
i: 4

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	10
s		1	2	3	2

10 is the best value for cutting a rod of length 4.



length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	10
s	0	1	2	3	2

How can we use the results in this table?

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	10
s	0	1	2	3	2

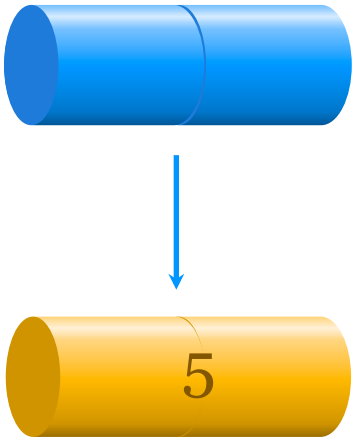
r[i] is the max value for cutting a rod of length i.

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	10
s	0	1	2	3	2

s[i] is the length of the *first piece* we should cut from a rod of length i.

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	10
s	0	1	2	3	2

For a rod of length 2, we get the max value of 5 when we start by cutting a piece of length 2.

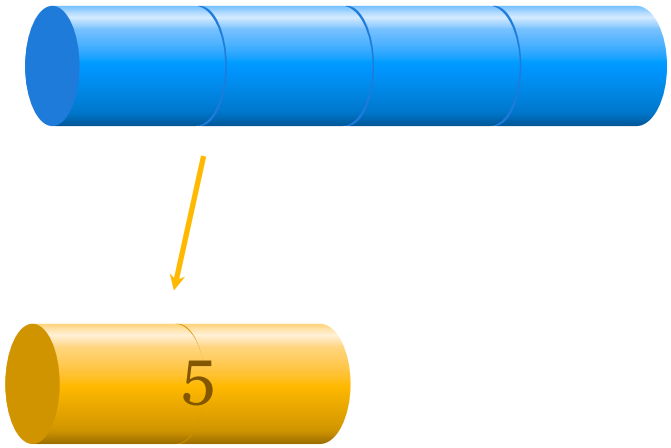


length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	10
s	0	1	2	3	2

How should we cut a rod of length 4?

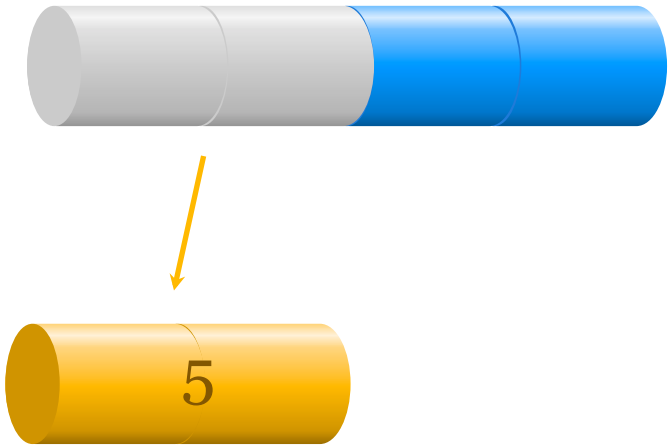
length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	10
s	0	1	2	3	2

For a rod of length 4 we should start by cutting off a piece of length 2.



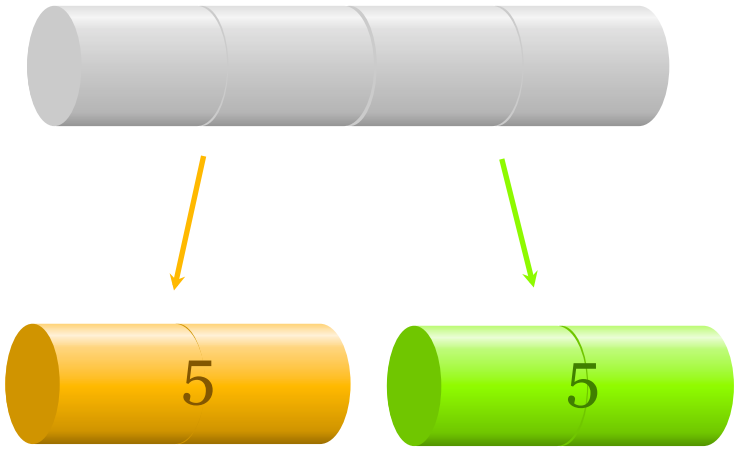
length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	10
s	0	1	2	3	2

After cutting the piece of length 2, the remaining portion has length 2.



length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	10
s	0	1	2	3	2

To get the maximum value from the remaining piece, we should cut it as one piece of length 2.



We have completely cut all pieces of the rod into sections that maximize the value so this is the maximum solution for rods of length 4.

length	0	1	2	3	4
p	0	1	5	8	9
r	0	1	5	8	10
s	0	1	2	3	2