Algorithm HW5_report

A. 程式說明

此題為 rod-cut problem，已知(輸入)單位長度 1~10 的對應價格，輸入 rod 的總長度，要求出如何切割才有辦法得到最高和最低的價格，此題有兩種動態規劃的方法，一種是 bottom-up，另一種是 top-down。Top-down 的方法是將原問題分解為更小的問題，用遞迴的方式將原本的問題分解成更小的問題，直到出現最小的子問題，而為了避免重複計算相同的子問題，我們須開一個 array 將以算過的子問題存起來，若遇到以計算過的子問題便直接得到答案。而 Bottom-up 的方法則是先從較小的子問題解決，利用迴圈慢慢往上解決更大的問題。

B. 程式結果

```
please input total length: 4
please input the price corresponding to length 1~10 :
1 5 8 9 10 17 17 20 24 30
-----Top Down:-----
total length:4
maximum price:10
2 2
number of pieces: 2

mininum price:4
1 1 1 1
number of pieces: 4

-----Bottom up:-----
total length:4
maxinum price:10
2 2
number of pieces: 2

mininum price:4
1 1 1 1
number of pieces: 4
```

```
please input total length: 30
please input the price corresponding to length 1~10 :
1 5 8 9 10 17 17 20 24 30
-----Top Down:-----
total length:30
maximum price:90
10 10 10
number of pieces: 3

mininum price:30
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
number of pieces: 30

-----Bottom up:-----
total length:30
maxinum price:90
10 10 10
number of pieces: 3

mininum price:30
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
number of pieces: 30
```

分別使用 Top-Down 和 Bottom-up 的方法跑出來的結果皆相同。

C. 探討與課本不同的地方

1.裁切後每根的最大上限為 10： 課本的方法裁切後的每一根都沒有固定長度，而此題的最大長度只有 10 個單位長，所以在裁切的時候不會出現有單一根的長度大於 10 的情況，所以在 for 迴圈裡面將條件限制於 10。

```
for(int j=i;j>=1;j--){
    if(j<=10){
        if(choice==0 && q<price[j]+r[i-j]){
            q=price[j]+r[i-j];
            s[i]=j;
        }
        else if(choice==1 && q>price[j]+r[i-j]){
            q=price[j]+r[i-j];
            s[i]=j;
        }
    }
```

2.需增加價格最小的情況 ： 與價格最大的情況相同，只需在判斷式改成 q<price[j]+r[i-j]，同時 q 的初始值要從負無限大改成正無限大，如此一來，在剛進迴圈時，不用額外判斷邊界條件。

3.若最高價值或最低價值相同，要輸出裁切數最少的 ： 在裁切時，若先從長度長的開始考慮，如上圖，會圈從 j=i 開始，慢慢往下遞減，如此會先考慮裁切數少的情況，而當裁切數較多但價值相同時，並不會進到判斷式裡面，因為判斷式為大於或小於，都沒有加等號。