

DSAP Homework3

B05703100 財金三 郭仲嘉

第一題

(a)

```
#include <iostream>
using namespace std;
```

```
long long cat(int num)
{
    if (num == 0 or num == 1) return 1;
    else
    {
        long long result = 0;
        for (int k = 0; k < num; k++)
            result += ( cat(k) * cat(num - 1 - k) );
        return result;
    }
}
```

```
int main()
{
    int num;
    cin >> num;
    cout << cat(num);

    return 0;
}
```

(b)

令 $S(I, J)$ 為前 I 個物品中，在包包負重上限為 J 的情況下，使總價值最高的組合的總價值

$$S(I, J) = \begin{cases} 0, & \text{when } I=0 \text{ or } J \leq 0 \\ \max \begin{cases} S(I-1, J-w_I) + V_I \\ S(I-1, J) \end{cases}, & \text{otherwise} \end{cases}$$

\downarrow \downarrow
 不選第 I 個物品 選第 I 個物品

令 $T(I, J)$ 為前 I 個物品中，在包包負重為 J 的情況下，使總價值最高的組合

$$T(I, J) = \begin{cases} \text{空集合}, & \text{when } I=0 \text{ or } J \leq 0 \\ T(I-1, J-w_I) \cup \{\text{物品 } I\}, & \text{選物品 } I \\ T(I-1, J), & \text{不選物品 } I \end{cases}$$

for $i=1$ to n

for $j=1$ to B

$$S(i, j) = \max \begin{cases} S(i-1, j-w_i) + v_i \\ S(i-1, j) \end{cases}$$

if $S(i-1, j-w_i) + v_i > S(i-1, j)$

$$T(i, j) = T(i-1, j-w_i) \cup \{\text{物品 } I\}$$

else

$$T(i, j) = T(i-1, j)$$

if $S(n, B) \geq V$ return $T(n, B)$

else return impossible

(c)

Queue<Path> q

Node s, t

for all nextPath of s

if capacity(nextPath == 0) continue

else q.enqueue(nextPath)

while (!q.empty())

```

for all nextPath of q.front()
    if nextPath reaches t, return nextPath
    if capacity(nextPath) == 0, continue
    if nextPath makes loop, continue
    capacity(nextPath) = min(capacity(nextPath), capacity(q.front()))
    q.enqueue(nextPath)
q.dequeue()
return Path not Found

```

(d)

(i)

從 v_1 開始用原演算法，但搜尋到 t_1 或 t_2 都可視為找到 augmenting path。

v_1 搜尋完畢後再以同樣規則搜尋完 v_2 、 v_3 。

(ii)

紀錄每個地點的 U_i 。搜尋到 augmenting path 時，紀錄增加此 path 每個地點會減少或增加(反方向流經)的流量。而搜尋時 path 的 maximum flow 現為：原 Path 的流量、下一條邊的剩餘流量、及下一個地點的剩餘流量三者的最小值。

原演算法中下一條邊的剩餘流量為 0 時，或加入下一個地點即形成圈圈時就不排入搜尋的 Queue 中。另加入一條件，下一個地點的剩餘流量為 0 時就不排入搜尋的 Queue。