- Quy hoạch động: Quy hoạch động là kĩ thuật được dùng khi có một công thức hoặc một (một vài) trạng thái bắt đầu. Một bài toán được tính bởi các bài toán nhỏ hơn đã được tìm ra từ trước, và kết quả các bài toán sẽ được lưu lại để những lần tính toán tiếp theo nếu cần đến những kết quả đó thì không cần tốn thêm thời gian thực hiện lại những bài toán này nữa. Nói cách khác, quy hoạch động bắt đầu từ việc giải các bài toán nhỏ, để từ đó từng bước giải quyết bài toán lớn hơn , và cuối cùng là bài toán lớn nhất ( bài toán ban đầu).

**Bài toán cái túi bằng phương pháp quy hoạch động**

1. Nêu bài toán

Cho **n** món hàng (n ≤ 50). Món thứ **i** có khối lượng là **A[i]** (số nguyên). Cần chọn những món hàng nào để bỏ vào một ba lô sao tổng khối lượng của các món hàng đã chọn là lớn nhất nhưng không vượt quá khối lượng **W** cho trước. Mỗi món chỉ chọn **1 hoặc không chọn.**

1. Mô tả chi tiết thuật toán

Ta nhận thấy rằng: Giá trị của cái túi phụ thuộc vào 2 yếu tố: Có bao nhiêu vật đang được xét và trọng lượng còn lại cái túi có thể chứa được, do vậy chúng ta có 2 đại lượng biến thiên. Cho nên hàm mục tiêu sẽ phụ thuộc vào hai đại lượng biến thiên. Do vậy bảng phương án của chúng ta sẽ là bảng 2 chiều.

Gọi F[i,j] là tổng giá trị lớn nhất của cái túi khi xét từ vật 1 đến vật i và trọng của cái túi chưa vượt quá j. Với giới hạn j, việc chọn tối ưu trong số các vật {1,2,…,i-1,i} để có giá trị lớn nhất sẽ có hai khả năng:

Nếu không chọn vật thứ i thì F[i,j] là giá trị lớn nhất có thể chọn trong số các vật {1,2,…,i-1} với giới hạn trọng lượng là j, tức là:

**F[i,j]:=F[i-1,j].**

Nếu có chọn vật thứ i (phải thỏa điều kiện W[i] ≤ j) thì F[i,j] bằng giá trị vật thứ i là V[i] cộng với giá trị lớn nhất có thể có được bằng cách chọn trong số các vật {1,2,…,i-1} với giới hạn trọng lượng j-W[i] tức là về mặt giá trị thu được:

**F[i,j]:=V[i]+F[i-1,j-W[i]]**

Vậy chúng ta phải xem xét xem nếu chọn vật i hay không chọn vật i thì sẽ tốt hơn. Từ đó chúng ta có công thức truy hồi như sau.

|  |
| --- |
| F(i, j) = max{F(i − 1, j), vi + F(i − 1, j − wi)} if j − wi ≥ 0,  = F(i − 1, j) if j − wi < 0. |

F[0,j] = 0 (hiển nhiên) – Bài toán con nhỏ nhất.

F[i,j]= max(F[i-1,j], V[i]+F[i-1,j-W[i]]

1. Bộ dữ liệu
2. Bộ thứ nhất

|  |  |
| --- | --- |
| 6 19  3 7  4 10  5 20  7 19  6 13  9 40 | Cac phan tu duoc chon la: 2 3 6  Tong khoi luong la: 18  Tong gia tri la: 70 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | i/L | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| **c** | **w** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **7** | **3** | 1 | 0 | 0 | 0 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| **10** | **4** | 2 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **20** | **5** | 3 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **19** | **7** | 4 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **13** | **6** | 5 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **40** | **9** | 6 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | i/L | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| **c** | **w** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **7** | **3** | 1 | 0 | 0 | 0 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| **10** | **4** | 2 | 0 | 0 | 0 | 7 | 10 | 10 | 10 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 |
| **20** | **5** | 3 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **19** | **7** | 4 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **13** | **6** | 5 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **40** | **9** | 6 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | i/L | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| **c** | **w** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **7** | **3** | 1 | 0 | 0 | 0 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| **10** | **4** | 2 | 0 | 0 | 0 | 7 | 10 | 10 | 10 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 |
| **20** | **5** | 3 | 0 | 0 | 0 | 7 | 10 | 20 | 20 | 20 | 27 | 30 | 30 | 30 | 37 | 37 | 37 | 37 | 37 | 37 | 37 | 37 |
| **19** | **7** | 4 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **13** | **6** | 5 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **40** | **9** | 6 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | i/L | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| **c** | **w** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **7** | **3** | 1 | 0 | 0 | 0 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| **10** | **4** | 2 | 0 | 0 | 0 | 7 | 10 | 10 | 10 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 |
| **20** | **5** | 3 | 0 | 0 | 0 | 7 | 10 | 20 | 20 | 20 | 27 | 30 | 30 | 30 | 37 | 37 | 37 | 37 | 37 | 37 | 37 | 37 |
| **19** | **7** | 4 | 0 | 0 | 0 | 7 | 10 | 20 | 20 | 20 | 27 | 30 | 30 | 30 | 39 | 39 | 39 | 46 | 49 | 49 | 49 | 56 |
| **13** | **6** | 5 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **40** | **9** | 6 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | i/L | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| **c** | **w** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **7** | **3** | 1 | 0 | 0 | 0 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| **10** | **4** | 2 | 0 | 0 | 0 | 7 | 10 | 10 | 10 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 |
| **20** | **5** | 3 | 0 | 0 | 0 | 7 | 10 | 20 | 20 | 20 | 27 | 30 | 30 | 30 | 37 | 37 | 37 | 37 | 37 | 37 | 37 | 37 |
| **19** | **7** | 4 | 0 | 0 | 0 | 7 | 10 | 20 | 20 | 20 | 27 | 30 | 30 | 30 | 39 | 39 | 39 | 46 | 49 | 49 | 49 | 56 |
| **13** | **6** | 5 | 0 | 0 | 0 | 7 | 10 | 20 | 20 | 20 | 27 | 30 | 30 | 33 | 39 | 39 | 40 | 46 | 49 | 49 | 52 | 56 |
| **40** | **9** | 6 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

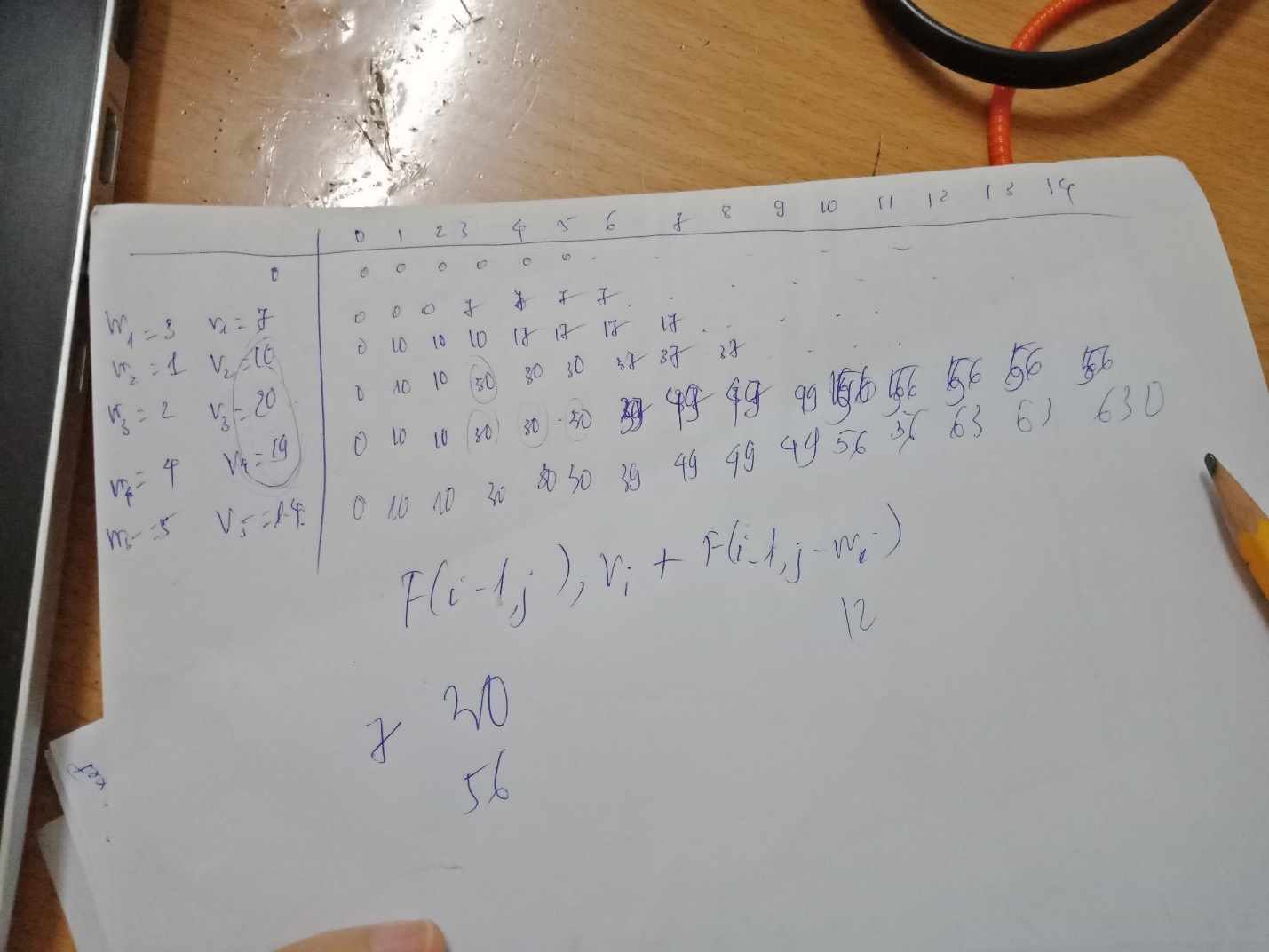
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | i/L | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| **c** | **w** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **7** | **3** | 1 | 0 | 0 | 0 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| **10** | **4** | 2 | 0 | 0 | 0 | 7 | 10 | 10 | 10 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 |
| **20** | **5** | 3 | 0 | 0 | 0 | 7 | 10 | 20 | 20 | 20 | 27 | 30 | 30 | 30 | 37 | 37 | 37 | 37 | 37 | 37 | 37 | 37 |
| **19** | **7** | 4 | 0 | 0 | 0 | 7 | 10 | 20 | 20 | 20 | 27 | 30 | 30 | 30 | 39 | 39 | 39 | 46 | 49 | 49 | 49 | 56 |
| **13** | **6** | 5 | 0 | 0 | 0 | 7 | 10 | 20 | 20 | 20 | 27 | 30 | 30 | 33 | 39 | 39 | 40 | 46 | 49 | 49 | 52 | 56 |
| **40** | **9** | 6 | 0 | 0 | 0 | 7 | 10 | 20 | 20 | 20 | 27 | 40 | 40 | 40 | 40 | 50 | 60 | 60 | 60 | 67 | 70 | 70 |

1. Bộ thứ hai

|  |  |
| --- | --- |
| 10 65  7 9  3 12  4 6  9 17  1 19  13 8  19 19  21 15  11 15  16 21 | Cac phan tu duoc chon la: 2 3 4 5 7 9 10  Tong khoi luong la: 63  Tong gia tri la: 109 |

1. Bộ thứ ba

|  |  |
| --- | --- |
| 5 14  3 7  1 10  2 20  4 19  5 14 | Cac phan tu duoc chon la: 2 3 4 5  Tong khoi luong la: 12  Tong gia tri la: 63 |



1. Chương trình Dev C++

|  |
| --- |
| #include<iostream>  #include<stdio.h>  #include <fstream>  using namespace std;  int MaxV[100][100];  int n;  int c[100]; //mang gia tri  int w[100]; //mang khoi luong  int L;  int select[100] = { 0 };  int main() {  fstream infile;  infile.open("inpt5.txt");  int t = 0;  infile >> n >> L;  for (int i = 1; i <= n; i++)  {  infile >> w[i] >> c[i];  }  infile.close();  //DP  for (int i = 0; i < L; i++) { MaxV[i][0] = 0; MaxV[0][i] = 0; }  for (int i = 1; i <= n; i++)  for (int b = 1; b <= L; b++)  {  MaxV[i][b] = MaxV[i - 1][b];  //F(i, j) = max{F(i − 1, j), vi + F(i − 1, j − wi)} if j − wi ≥ 0,  if (b >= w[i] && (MaxV[i - 1][b - w[i]] + c[i] > MaxV[i - 1][b]))  {  MaxV[i][b] = MaxV[i - 1][b - w[i]] + c[i];  }  }  int i = n, j = L;  while (i > 0) {  //neu chon vat thu i  if (j >= w[i] && (MaxV[i - 1][j - w[i]] + c[i] > MaxV[i - 1][j])) {  select[i] = 1; //danh dau la duoc chon  j = j - w[i]; //khoi luong con lai  }  i--;  }  int sum = 0;  cout << "Cac phan tu duoc chon la: ";  for (int i = 1; i <= n; i++) {  if (select[i] == 1) {  cout << i << " ";  sum += w[i];  }  }  cout << endl << "Tong khoi luong la: " << sum;  cout << endl << "Tong gia tri la: " << MaxV[n][L];  return 0;  } |

1. Đánh giá độ phức tạp

**Time Complexity:** O(n\*b) . vs n là số phần tử và b là khối lượng do duyệt các phần tử từ 1 đến n, vòng lặp bên trong duyệt khối lượng từ 1 -> b => độ phức tạp

**Auxiliary Space:** O(N\*W).