

The background of the entire image is a dark blue field filled with a pattern of red dots. These dots are arranged in a way that they form a large, faint, stylized circular shape, reminiscent of a DNA helix or a molecular structure, with the density of the dots varying to create a sense of depth and movement.

HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.



ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

Applied Algorithm Lab

Warehouse

ONE LOVE. ONE FUTURE.

- N stations located on a straight line
 - Station i has amount of goods a_i , located at i , pick-up time t_i
- **Objective:** Find a route for a truck to pick-up goods at stations
 - two consecutive stations: has distance $\leq D$
 - total time to pick up $\leq T$
 - maximize the total amount of goods
- Output: the total amount of goods
- Example:

Input:

6 6 2
6 8 5 10 11 6
1 2 2 3 3 2

Output:

24
ex = 2
pick: 6-8-10

- Idea to solve: dynamic programming
 - Let $f[i][k]$ be the maximal amounts of good if we choose some stations in $1 \rightarrow i - 1$, and **choose station i** and the total time is $\leq k$.
 - Formula:
 - If $k < t[i] : f[i][k] = 0;$
 - If $k \geq t[i] : f[i][k] = \max(f[j][k - t[i]] + a[i]), j = i - D \rightarrow i - 1;$
 - return:
 $\max(f[i][k]), i = 1 \rightarrow n, k = 1 \rightarrow T;$
 - Complexity:
 $O(n * T * D).$

Warehouse - Implementation

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int N, T, D;
4  const int maxN = 1001, maxT = 101, maxD = 11;
5  int result, S[maxN][maxT], a[maxN], t[maxN];
6  // S[i][k] = lượng hàng lớn nhất lấy từ 1->i, lấy kho a[i] và tổng tgian <= k
7
8  void input() {
9      cin >> N >> T >> D;
10     for (int i=1; i<=N; i++)
11         cin >> a[i];
12
13     for (int i=1; i<=N; i++)
14         cin >> t[i];
15 }
16
```

Warehouse - Implementation

```
17  int main() {
18      input();
19      int ans = a[1]; S[1][t[1]] = a[1]; // trạng thái cơ sở
20
21      for(int i=1; i<=N; i++) { // quy hoạch động bottom-up
22          for (int k=1; k<=T; k++) {
23              if (k<=t[i]) { // nếu k<t[i] thì không thể lấy được
24                  for (int j=1; j<=D && i-j>=0; j++)
25                      S[i][k] = max(S[i][k], S[i-j][k-t[i]]+a[i]);
26                  ans = max(ans, S[i][k]);
27              } else S[i][k] = 0;
28          }
29      }
30      cout << ans;
31      return 0;
32  }
```





HUST

THANK YOU !