ETH Eidgenössische Technische Hochschule Zürich	Ecole polytechnique fédérale de Zurich Politecnico federale di Zurigo Federal Institute of Technology at Zurich

Institut für Theoretische Informatik
Peter Widmayer
Thomas Tschager
Antonis Thomas

6th April 2016

Datenstrukturen & Algorithmen — Exercise Sheet 6 — FS 16

Remarks about the description of a dynamic program: A complete description of a dynamic program **always** considers the following aspects (interesting also for the exam!):

1) *Definition of the DP Table*: What are the dimensions of the table? What is the meaning of each entry?

2) *Computation of an entry*: How can an entry be computed from the values of other entries? Which entries do not depend on others?

3) *Calculation order*: In which order can entries be computed so that values needed for each entry have been determined in previous steps?

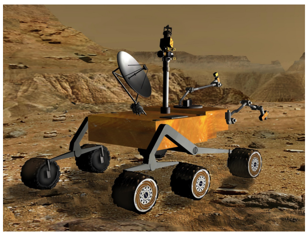
4) *Extracting the solution*: How can the final solution be extracted once the table has been filled?

The running time of a dynamic program is usually easy to calculate by multiplying the size of the table with the time required to compute each entry. Sometimes, however, the time to extract the solution dominates the time to compute the entries.

Exercise 6.1 *Mars mission.*

The rover *Curiosity* landed on Mars and is located at a starting position S. The goal is to move to a target position Z, and to collect rock samples that are as valuable as possible. To not use too much energy, the rover is only allowed to take a step to the east (right) and to the south (down). The value of each rock sample is stored in an $(m \times n)$ matrix, e.g.

S	9	2	5	11	8
17	21	32	5	15	3
2	2	3	8	1	5
8	2	8	11	15	9
0	5	3	10	4	**Z**



In the matrix above, an example of a south-east path from S to Z is shown where the value of the collected rock samples is maximum. This path can be described by enumerating the corresponding matrix positions: $(1,1) \to (2,1) \to (2,2) \to (2,3) \to (2,4) \to \cdots \to (5,6)$.

Provide a *dynamic programming* algorithm that takes an $(m \times n)$ matrix A with $A[1,1] = A[m,n] = 0$, and that computes a south-east path from $S = (1,1)$ to $Z = (m,n)$ where the value of the collected rock samples is maximal. Notice that we search for the path itself, and not just for the maximum value. Provide also the running time of your solution in dependency of m and n.

Please turn over.

Exercise 6.2 *Self-Organizing Linear Lists.*

Consider the following self-organizing list. Provide the number of key comparisons when the elements 'R', 'A', 'I', 'S', 'T', 'A', 'A', 'G' are accessed in this order using the Move-to-Front rule.

$$A \longrightarrow L \longrightarrow G \longrightarrow O \longrightarrow R \longrightarrow I \longrightarrow T \longrightarrow H \longrightarrow M \longrightarrow U \longrightarrow S$$

Exercise 6.3 *Trading (**Programming Exercise**).*

There are n different item types at your disposal. You can exchange a set of three items of pairwise different types into 1 Franc. What is the maximum amount of Francs you can make if you are given a_i many items of type i?

Input The first line of the input contains only the number t of test instances. After that, we are given t testcases in the following way: the first integer $n \leq 5000$ indicates the amount of different item types. After that, we are given n integers $a_i \leq 2000$ for $1 \leq i \leq n$ indicating the number of items of type i.

Output For each testcase, output the largest amount of Francs that you can make with the given items.

Example

Input:

```
3
2 5 10
3 2 3 4
4 1 2 3 4
```

Output:

```
0
2
3
```

Testsets There are three categories of testsets for a total of 100 points:

- **Small**: You can assume $n \leq 20$. Worths 40 points.
- **Medium**: You can assume $n \leq 1000$. Worths 40 points.
- **Large**: No further assumptions. Worths 20 points.

Hand-in: Wednesday, 13th April 2016 in your exercise group.

Hints: It is faster to solve this with the use of an array `Bucket[]` such that `Bucket[k]` corresponds to the number of different item types that have k items.