




# ACM-ICPC Thailand Southern Area Programming Contest 2010

## PSU Phuket Campus

# Practice Session

**12 July 2010**

- There are **4 problems** (A-D) to solve within 100 minutes.
- Solve as many problems as you can, in an order of your choice.
- Use C or C++ or Java to program at your convenience for any problems.
- Input and output of each program are **standard input** and **output**.

**Problem A: Eleven-Divisible**
**Problem B: Polygon Area**
**Problem C: Aircraft Maintenance: Lowest Cost**
**Problem D: Aircraft Maintenance: Shortest time**

***Organized by Department of Computer Engineering, PSU Phuket Campus***

# Problem A
# Eleven-Divisible

A test of divisibility by 11 can be done in an interesting and simple way. First, we number all digits (either from left to right or right to left) by 1, 2, 3, ... Then, we calculate two sums: the odd numbered digits and the even numbered digits, and then subtract one sum from the other, and see if the result (ignore the negative sign) is divisible by 11. By the way, if we end up with zero, then the original number is divisible by 11. We can also repeat that process until we reach 11 or zero. Take a look at these examples:

| Number | 132 | 23640258 | 19809031 |
|---|---|---|---|
| Sum of odd numbered digits | 1+2= 3 | 2+6+0+5= 13 | 1+8+9+3= 21 |
| Sum of even numbered digits | 3 | 3+4+2+8= 17 | 9+0+0+1= 10 |
| Result | 3-3= 0 | 13-17= -4 | 21-10=11 |
| Eleven-divisible | **Yes** | **No** | **Yes** |

We can, of course, do the summing in different orders. In fact we can just go from left to right adding and subtracting alternate digits, e.g. 1-9+8-0+9-3+1=11 (divisible by 11).

Given a group of positive integers, your task is to write a program for searching the maximum value of 11-divisible number in the group.

**Input**
The input contains a positive integer $g$ ($1 \le g \le 100$) in the first line which determines the number of groups. Next comes each group described as follows. The first line includes a positive integer $n$ and followed by $n$ positive integers in a line or multiple lines. The range of input numbers can be very large. They can contain up to 100 digits ($1 - 10^{100}$).

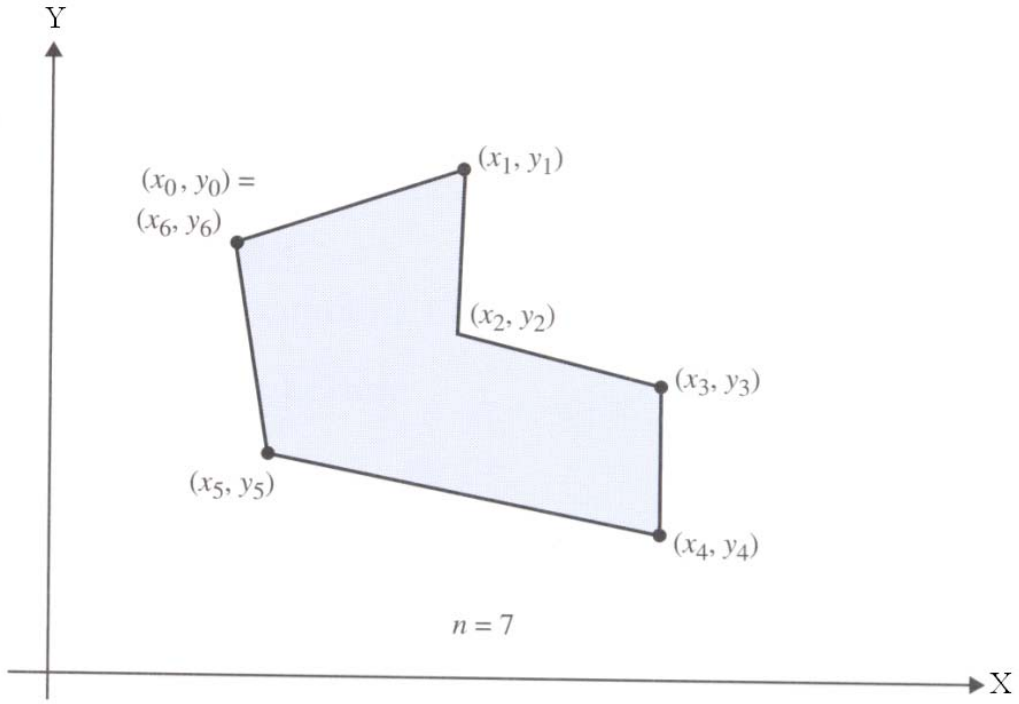
**Output**
The output has $g$ lines. Each line contains a maximum 11-divisible integer of each group.

| **Sample input** | **Sample output** |
|---|---|
| 2<br>9<br>35 202 1180 132 222 1025<br>8415 9250 8990<br>10<br>1235 132 126 8419 3201 2310 4752 1321 35900 990 | 8415<br>4752 |

# Problem B
# Polygon Area

If *n* points are connected to form a closed polygon as shown below, the area *A* of the polygon can be calculated as

$$A = \frac{1}{2}\left|\sum_{i=0}^{n-2}(x_{i+1} + x_i)(y_{i+1} - y_i)\right|$$




Notice that although the illustrated polygon has only six distinct corners, *n* for this polygon is 7 because the algorithm expects that the last point, $(x_6, y_6)$, will be a repeat of the initial point, $(x_0, y_0)$.

Write a program for calculating the area of a polygon from a set of coordinates of polygon corners.

**Input**

The number of input polygons (*np*) is given in the first line (*1≤ np ≤ 100*). The following lines contain information of polygons. Each polygon starts with the number *n* (*1≤ n ≤ 100*) in the first line and the next *n lines* contain coordinates of polygon corners (values of *x* and *y* separated by a space).

**Output**

The output includes *np* lines containing the area of input polygons in the same order as input. The output must be real numbers with 2-digit decimal fraction.

| Sample input | Sample output |
|---|---|
| 2<br>4<br>1 1<br>3 1<br>3 3<br>1 1<br>7<br>4 0<br>4 7.5<br>7 7.5<br>7 3<br>9 0<br>7 0<br>4 0 | 2.00<br>25.50 |

# Problem C

# Aircraft Maintenance: Lowest Cost

Many operational engineering systems require complex scheduling of people, machines and supplies to provide a service or produce a product. To schedule a system, one needs to know three things: the resources available to the system, the resources required to provide the desired service, and any constraints on the resources. Many sophisticated algorithms are developed to minimize the cost or time required to provide a service.

Here you are dealing with scheduling of aircraft maintenance of an airlines company. Suppose you have three crews with different qualifications as follows:

| Crew No. | Skill Level | Cost of crew per hour |
|---|---|---|
| 0 | 1 | 200 |
| 1 | 2 | 300 |
| 2 | 3 | 400 |

A crew member having higher skill level can also perform the work that requires a lower skill level. For example, Crew no.2 can do the task of level 3, 2 and 1 but Crew no. 0 can only do the level-1 task.

The list of aircraft maintenance tasks is described in the following table. You need to schedule the tasks by assigning them to available crew members. Note that each task must be done in its entirety by one crew and the crews are paid by their working hours on the task(s).

| Task ID | Level of maintenance (required skill level) | Number of hours |
|---|---|---|
| A101 | 1 | 5 |
| B102 | 1 | 10 |
| C103 | 1 | 6 |
| B201 | 2 | 4 |
| B202 | 2 | 7 |
| C203 | 2 | 3 |
| A301 | 3 | 2 |
| A302 | 3 | 6 |

Given a list of crew members with their skill level and cost and a list of tasks, required levels and hours, you have to write a program to schedule these tasks in order to minimize the cost of the maintenance work.

**The Input**

The first *n* lines (*1*≤ *n* ≤ *100*) contain information of the crew members: crew number (from *0 to n-1*), skill level, and cost per hour (all numbers are integers). All crew members have different levels. The following lines contain task ID (always starting with a letter a-z or A-Z), level of maintenance and the required hours of each task. The line beginning with 0 indicates the end of input.

**The Output**

The output consists of the lowest total cost of the maintenance work.

| Sample input | Sample output |
|---|---|
| `0 1 200`<br>`1 2 300`<br>`2 3 400`<br>`A101 1 5`<br>`B102 1 10`<br>`C103 1 6`<br>`B201 2 4`<br>`B202 2 7`<br>`C203 2 3`<br>`A301 3 2`<br>`A302 3 6`<br>`0` | `11600` |

# Problem D

# Aircraft Maintenance: Shortest time

In this problem we deal with the same scenario as described in Problem C but this time the objective of the scheduling is to minimize the job hours (no matter how much they cost). You have to write a program to arrange the tasks in order to get all tasks done in the shortest time. Note that all crew members can work at the same time and all task can be done independently i.e. there is no dependency between tasks.

### The Input

The first *n* lines ($1 \le n \le 100$) contain information of the crew members: crew number (from *0 to n-1*), skill level, and cost per hour (all numbers are integers). All crew members have different levels. The following lines contain task ID (always starting with a letter a-z or A-Z), level of maintenance and the required hours of each task. The line beginning with 0 indicates the end of input.

### The Output

The output consists of the minimum number of hours required by the maintenance work.

| Sample input | Sample output |
|---|---|
| 0 1 200<br>1 2 300<br>2 3 400<br>A101 1 5<br>B102 1 10<br>C103 1 6<br>B201 2 4<br>B202 2 7<br>C203 2 3<br>A301 3 2<br>A302 3 6<br>0 | 15 |

### Explanation

Since all crew can work in parallel, the shortest time required for the maintenance work is the longest hours among crew members. In the Sample input, in order to achieve the minimum working hours, the tasks must be distributed to the crews as follows:

- Crew No.2 gets tasks A302, A301 and C103 and has 6+2+6 = 14 working hours.
- Crew No.1 gets tasks B201, B202 and C203 and has 4+7+3 = 14 working hours.
- Crew No.0 gets tasks A101 and A102 and has 5+10 = 15 working hours.

Therefore the resulting minimum number of hours is 15.