Test Session for the Northwestern Europe Regional Contest 2025

NWERC 2025 Test Session

November 29, 2025

Problems

This page is intentionally left blank.

NWERC 2025 Test Session

Problem A
Atlantic Email
Time limit: 1 second

CC BY-SA 4.0 by Reinhard Ferdinand on Wikimedia Commons

The year is 1984. Your boss, Professor Werner Zorn, has been pushing for establishing connections to international networks. His newest project is for Karlsruhe University[1] to receive the first email from the US Computer Science Network (CSNet).

The email will be sent by Laura Breeden (Massachusetts Institute of Technology) to the Karlsruhe University computers. For simplicity, the email only consists of n lowercase letters. It should be encoded so it can be sent across 5 transatlantic binary channels.

You want the communication to be done efficiently, so you require that across each channel, no more than $n + 10$ bits are sent. The channels are working flawlessly and losslessly, so for each of the 5 channels, the bits will arrive intact in Karlsruhe.

Everything was almost ready, when you noticed that the wires coming into the computer in Karlsruhe are a mess, and so you do not know which of the 5 channels is which. Thus, as the receiver, you will receive the binary strings of the 5 channels in arbitrary order. Instead of untangling the mess, you think it is a better idea to just fix it in software. Design a communication protocol that can transfer the email correctly and efficiently.

Input

The input consists of:

- One line with the action that your program needs to perform: either the string "`send`" or "`receive`", which denotes whether you are at the sending or receiving end.
- If the action is "`send`":
  - One line with one integer n ($1 \leq n \leq 10\,000$), the length of the email.
  - One line with a string s of length n, the email to send. The email only consists of English lowercase letters (`a-z`).
- If the action is "`receive`":
  - Five lines, each with a string b ($1 \leq |b| \leq n + 10$), only consisting of digits 0 and 1, the five binary strings you used to encode the email, in an arbitrary order.

Note that when receiving, you have to infer n from the received binary strings, as this is not known to you up front.

This is a multi-pass problem. Your program will be invoked multiple times, possibly more than twice. It is guaranteed that the first pass is a "`send`" action, and that each subsequent pass is a "`receive`" action.

For testing purposes, the number and input of subsequent passes will depend on the output of your submission.

A testing tool is provided to help you develop your solution.

[1] Former university in Karlsruhe that later merged to become Karlsruhe Institute of Technology.

Output

If the action is "`send`", output 5 bit strings $b_1, \dots, b_5$ ($1 \leq |b_i| \leq n + 10$ for each i), only consisting of digits 0 and 1, the bit strings to send.

If the action is "`receive`", output the original email s, a string only consisting of English lowercase letters (`a-z`).

Sample Case 1

Sample Input **Pass 1** **Sample Output**

```
send
17
willkommenimcsnet
```

```
0
01
111111
1001
1
```

Sample Input **Pass 2** **Sample Output**

```
receive
1
01
1001
0
111111
```

```
willkommenimcsnet
```

Sample Case 2

Sample Input **Pass 1** **Sample Output**

```
send
3
kit
```

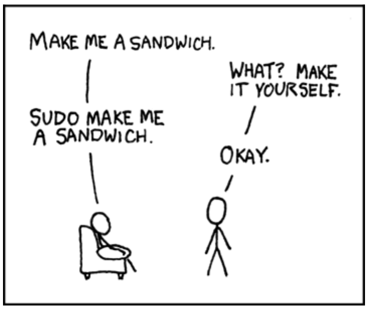
```
000000000
111111111
000000000
111111111
000000000
```

Sample Input **Pass 2** **Sample Output**

```
receive
111111111
000000000
000000000
000000000
111111111
```

```
kit
```

Problem B
Build Dependencies
Time limit: 5 seconds



xkcd comic 149.
CC BY-NC 2.5 by Randall Munroe on xkcd.com

A Makefile is a file that specifies dependencies between different source code files. When one source code file changes, this file needs to be recompiled, and when one or more dependencies of another file are recompiled, that file needs to be recompiled as well. Given the Makefile and a changed file, output the set of files that need to be recompiled, in an order that satisfies the dependencies (i.e., when a file X and its dependency Y both need to be recompiled, Y should come before X in the list).

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10^5$), the number of Makefile rules.
- n lines, each with a Makefile rule. Such a rule starts with "f**:**" where f is a filename, and is then followed by a list of the filenames of the dependencies of f. Each file has at most 5 dependencies.
- One line with one string c, the filename of the changed file.

Filenames are strings consisting of between 1 and 10 English lowercase letters (`a-z`). Exactly n different filenames appear in the input file, each appearing exactly once as f in a Makefile rule. The rules are such that no two files depend (directly or indirectly) on each other.

Output

Output the list of files that need to be recompiled, in an order such that all dependencies are satisfied.

If there are multiple valid solutions, you may output any one of them.

Sample Input 1

```
6
gmp:
solution: set map queue
base:
set: base gmp
map: base gmp
queue: base
gmp
```

Sample Output 1

```
gmp
map
set
solution
```

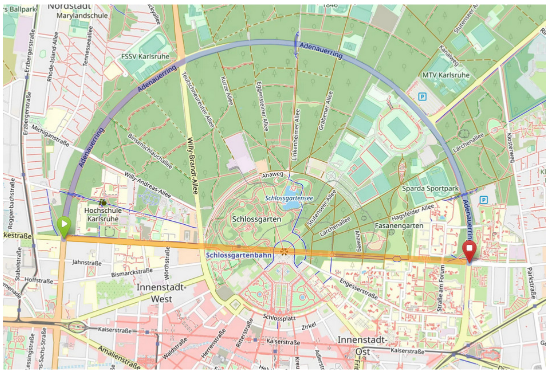
This page is intentionally left blank.

NWERC 2025 Test Session

Problem C
Circular Shortcut
Time limit: 1 second

After the practice session, Katrin and her team join the guided city tour of Karlsruhe, where they learn all sorts of random fun facts about Karlsruhe. It is well-known that the first email in Germany was received here (sent by Laura Breeden, MIT) and that the bicycle was invented in Karlsruhe, but did you know that the palace has exactly 365 windows – one for each day of the year? Or that there may (or may not) be an umbrella buried beneath the great pyramid in the city centre?

Later that evening, Katrin and her team find themselves in the western part of Karlsruhe, and they have to hurry to the Dining Hall to make it in time for dinner! They could take the direct route through the palace garden, which is a straight line. But on such a pleasant evening, the palace garden is crowded with pedestrians and cyclists who might slow them down. The alternative is to follow the *Adenauerring*, a street forming a perfect half-circle between their current location and the intersection close to the Dining Hall, and nothing will slow them down there.



Map data from OpenStreetMap

Figure C.1: The two possible paths from their current location to the intersection close to the Dining Hall. The blue semi-circular road is the Adenauerring, and the orange straight path in the middle is the route through the palace garden.

Katrin and her team have arranged to meet up with their coach on the intersection close to the Dining Hall, as shown in Figure C.1. They would like to know: how much longer is the route along the Adenauerring?

Input

The input consists of:

- One line with an integer d ($1 \leq d \leq 10^9$), the length of the straight way through the palace garden, in metres.

Output

Output how much longer the route along the Adenauerring is, compared to going straight through the palace garden, in metres.

Your answer should have an absolute or relative error of at most 10^{-9}.

Sample Input 1

```
2025
```

Sample Output 1

```
1155.86256175967
```

Sample Input 2

```
123456789
```

Sample Output 2

```
70468681.6790926
```