

# Angka Tebak

Batas Waktu	1 detik
Batas Memori	32 MB

## Deskripsi

Semoga Anda masih ingat soal **Tebak Angka** di mana Anda diharuskan menebak angka yang dipikirkan Pak Dengklek (dari rentang 1 hingga N) dengan menebak maksimal Q kali dan Pak Dengklek akan menjawab TERLALU KECIL, TERLALU BESAR, atau SELAMAT apabila berhasil menebak angka yang dipikirkan.

Sekarang, mari kita bertukar posisi!

Anda diminta untuk memikirkan sebuah angka dari rentang 1 hingga N. Anda dinyatakan menang apabila Pak Dengklek tidak dapat menebak angka yang Anda pikirkan hingga akhir tebakan (tebakan ke-Q).

Sama seperti strategi Anda pada soal **Tebak Angka**, Pak Dengklek memakai prinsip *Divide & Conquer* untuk menebak angka yang Anda pikirkan. Apabila angka yang Anda pikirkan berada pada rentang A hingga B, maka Pak Dengklek akan menebak  $C = (A + B) / 2$  sebagai angka yang Anda pikirkan. Apabila C bukan berupa bilangan bulat, maka secara acak Pak Dengklek akan memilih pembulatan ke atas atau pembulatan ke bawah. Tentu saja apabila Anda menjawab TERLALU BESAR maka angka yang Anda pikirkan berada pada rentang A hingga C - 1 dan apabila Anda menjawab TERLALU KECIL maka angka yang Anda pikirkan berada pada rentang C + 1 hingga B.

Perlu diperhatikan bahwa Pak Dengklek dapat mengecek apakah jawaban-jawaban Anda konsisten sehingga kecurangan-kecurangan yang bisa saja Anda lakukan secara otomatis akan langsung terdeteksi. Salah satu contoh kecurangan yaitu Anda menjawab TERLALU BESAR ketika Pak Dengklek menebak angka 1.

## Format Interaksi

Pada awalnya, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.
- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, dan berisi '.' jika bukan.
- Untuk setiap nilai i di antara 1 hingga banyaknya subsoal, berlaku:
  - jika kasus uji tersebut memenuhi batasan subsoal ke-i, maka karakter ke-i berisi i, atau

- jika kasus uji tersebut tidak memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0 . . 3 4 5, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Selanjutnya, program Anda akan menerima input dua buah bilangan  $N$  dan  $Q$ .

Kemudian, Anda akan menerima sebuah bilangan bulat  $C$  yaitu tebakan Pak Dengklek untuk angka yang Anda pikirkan. Anda diminta untuk menjawab TERLALU KECIL, TERLALU BESAR, atau SELAMAT berdasarkan angka yang Anda pikirkan.

Apabila Anda menjawab SELAMAT, artinya Pak Dengklek berhasil menebak angka yang Anda pikirkan dan Anda dinyatakan kalah. Apabila Anda menjawab selain SELAMAT, maka Pak Dengklek akan menebak kembali angka yang Anda pikirkan.

Anda dinyatakan menang apabila hingga tebakan ke- $Q$  Pak Dengklek masih belum mendapatkan jawaban SELAMAT.

### Contoh Interaksi

Keluaran Program Peserta	Umpan Balik Grader
	0 . . . 4 .
	9 3
	5
TERLALU BESAR	
	2
TERLALU KECIL	
	4
TERLALU BESAR	
	(interaksi selesai)

### Penjelasan Contoh Interaksi

Pada kasus tersebut, nilai N dan Q secara berturut-turut adalah 9 dan 3. Dari angka 1 hingga 9, Anda memilih angka 3.

Untuk tebakan pertama, nilai A dan B adalah 1 dan 9 sehingga  $C = (1 + 9) / 2 = 5$ .

Karena 5 terlalu besar dibandingkan dengan 3, maka Anda menjawab **TERLALU BESAR**. Sehingga untuk tebakan kedua, nilai A dan B adalah 1 dan 4. Karena nilai  $C = (1 + 4) / 2 = 2,5$  bukan merupakan bilangan bulat, maka Pak Dengklek secara acak memilih untuk membulatkan ke bawah.

Karena 2 terlalu kecil dibandingkan dengan 3, maka Anda menjawab **TERLALU KECIL**. Sehingga untuk tebakan ketiga, nilai A dan B adalah 3 dan 4. Karena nilai  $C = (3 + 4) / 2 = 3,5$  bukan merupakan bilangan bulat, maka Pak Dengklek secara acak memilih untuk membulatkan ke atas.

Karena 4 terlalu besar dibandingkan dengan 3, maka Anda menjawab **TERLALU BESAR**. Karena setelah  $Q = 3$  tebakan Pak Dengklek tidak mendapatkan jawaban **SELAMAT**, maka Anda dinyatakan menang pada permainan ini.

## Subsoal

### Subsoal 1 (9 poin)

- $N = 31$
- $Q = 4$

### Subsoal 2 (9 poin)

- $N = 127$
- $Q = 6$

Khusus untuk subsoal 1 dan subsoal 2:

- Hanya terdapat sebuah kasus uji (satu subsoal dinyatakan oleh satu kasus uji), yang dapat dimainkan [di sini](#).
- Anda boleh memainkan permainan ini berulang kali tanpa mendapatkan penalti.
- Jika Anda sudah memenangkan permainan untuk subsoal tertentu, Anda dapat memilih pilihan pada permainan untuk mengeluarkan *source code* yang dapat langsung Anda kirimkan ke *grader* dan menjawab dengan benar pada subsoal yang telah Anda menangkan.
- Anda tidak diwajibkan memainkan permainan ini untuk mengerjakan kedua subsoal ini. Anda diperbolehkan untuk menulis kode Anda sendiri untuk mengerjakan kedua subsoal ini.

### Subsoal 3 (20 poin)

- $N = 4$
- $Q = 2$

### Subsoal 4 (27 poin)

- $N = 9$
- $Q = 3$

**Subsoal 5 (35 poin)**

- $N = 16$
- $Q = 4$

**Catatan**

Yang perlu diperhatikan adalah bahwa untuk tipe soal interaktif seperti ini, Anda harus selalu memberikan perintah `"fflush(stdout);"` (bagi pengguna C/C++) atau `"flush(output);"` (bagi pengguna PASCAL) setiap kali Anda mencetak keluaran (dengan kata lain, setiap kali ada perintah mencetak keluaran misalnya `write`, `writeln`, `printf`, `cout`, atau `puts`, tepat di bawahnya harus ada perintah `fflush/flush`).

Sebagai contoh, berikut adalah contoh *source code* dalam bahasa Pascal yang akan selalu memikirkan angka 42 tanpa mepedulikan nilai  $N$  dan  $Q$  yang diberikan.

```
var subsoal: string;
    N, Q, C, i: longint;

begin
  readln(subsoal);
  readln(N, Q);

  for i := 1 to Q do begin
    readln(C);
    if (C < 42) then begin
      writeln('TERLALU KECIL');
      flush(output);
    end else if (C > 42) then begin
      writeln('TERLALU BESAR');
      flush(output);
    end else begin
      writeln('SELAMAT');
      flush(output);
    end;
  end;
end.
```

Dan berikut adalah contoh *source code* dalam bahasa C++.

```
#include <cstdio>
#include <cstring>

char subsoal[100];
int N, Q, C, i;

int main() {
  scanf("%s", subsoal);
  scanf("%d %d", &N, &Q);

  for (i = 1; i <= Q; i++) {
    scanf("%d", &C);
    if (C < 42) {
```

```
        printf("TERLALU KECIL\n");
        fflush(stdout);
    } else if (C > 42) {
        printf("TERLALU BESAR\n");
        fflush(stdout);
    } else {
        printf("SELAMAT\n");
        fflush(stdout);
    }

    return 0;
}
```

## Peringatan

Apabila program Anda melakukan salah satu dari hal-hal di bawah ini:

- melakukan tindakan tidak sesuai format sehingga tidak dikenali oleh *grader*,
- melakukan kecurangan dengan tidak konsistennya jawaban Anda, atau
- gagal memenangkan permainan,

maka program Anda akan dihentikan secara otomatis dan Anda mendapatkan *verdict Wrong Answer* pada kasus uji yang bersangkutan.

# Tebak Pokémon

Batas Waktu	4 detik
Batas Memori	256 MB

## Deskripsi



Pokédex adalah sebuah komputer saku yang mutlak dimiliki oleh pelatih Pokémon untuk mengidentifikasi spesies Pokémon yang ditemui. Dengan mengarahkan Pokédex menghadap Pokémon, pengguna akan mendapatkan data lengkap mengenai Pokémon tersebut (seperti morfologi, informasi evolusi, dan kekuatan khusus) hampir secara instan. Namun apakah Anda tahu bagaimana proses yang terjadi di dalam Pokédex sehingga bisa menampilkan data yang sesuai dengan sangat cepat?

Pada Pokédex tertanam sebuah kamera mini khusus yang berfungsi merekam ciri-ciri dari Pokémon yang hendak diidentifikasi. Ciri-ciri ini akan direkam ketika pengguna mengarahkan Pokédex menghadap Pokémon. Dalam sekali proses identifikasi, kamera mini mampu merekam sebanyak  $N$  ciri-ciri. Ciri-ciri tersebut kemudian diterjemahkan ke dalam bahasa manusia (dalam kasus ini bahasa Inggris), lalu dilakukan pencocokan dengan *database* di Pokémon Center agar didapat nama beserta data lengkap Pokémon yang diidentifikasi. Hal yang menarik adalah proses penerjemahan sangat canggih sedemikian sehingga bahasa manusia yang didapat selalu merupakan *substring* dari data lengkap Pokémon tersebut di *database*.

Dalam pengembangan Pokédex generasi berikutnya, Anda direkrut oleh Professor Dengklek, seorang peneliti Pokémon ternama yang juga merupakan rekan sejawat Professor Oak, untuk mengimplementasikan algoritma pencocokan antara ciri-ciri yang direkam kamera mini khusus dengan *database* di Pokémon Center.

Pada pengembangan iterasi pertama, Anda hanya perlu mencocokkan data dari Pokémon generasi pertama. Professor Dengklek tidak memedulikan algoritma yang Anda gunakan, asalkan hasilnya dapat sangat akurat. Dapatkah Anda menyelesaikan tugas Anda dengan baik?

## Format Masukan

Pada awalnya, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.

- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, dan berisi '.' jika bukan.
- Untuk setiap nilai  $i$  di antara 1 hingga banyaknya subsoal, berlaku:
  - jika kasus uji tersebut memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi  $i$ , atau
  - jika kasus uji tersebut tidak memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0 . . 345, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Baris kedua berisi sebuah bilangan bulat  $N$  yang merupakan banyaknya ciri-ciri Pokémon yang direkam oleh kamera mini khusus pada Pokédex generasi terbaru.

Berikutnya terdapat  $N$  baris. Pada tiap baris berisi *string*  $S$  yang menyatakan hasil terjemahan ciri-ciri Pokémon ke dalam bahasa manusia. Seperti yang disinggung di soal, **hasil terjemahan selalu merupakan substring dari data lengkap Pokémon tersebut di database.**

Sebagai referensi, *database* Pokémon yang lengkap dapat diunduh [di sini](#). Semua data yang terdapat dalam berkas tersebut diambil dari situs [Bulbapedia](#).

## Format Keluaran

Untuk ciri-ciri Pokémon yang diberikan, cetak keluaran dalam 1 baris berisi nama Pokémon yang dimaksud dari hasil analisis ciri-ciri Pokémon tersebut.

## Contoh Masukan

```
0.....6
6
The flame that burns at the tip of its tail is an indication of its
emotions
a bipedal, reptilian Pokemon with an orange body
A fire burns at the tip of this Pokemon's slender tail
one of three starter Pokemon of Kanto available at the beginning of Pokemon
Red, Green, Blue, FireRed, and LeafGreen
The flame at the tip of its tail makes a sound as it burns
Obviously prefers hot places
```

## Contoh Keluaran

Charmander

## Penjelasan Contoh Kasus Uji

Pada beberapa *browser*/peramban atau versi cetak, tampilan contoh masukan di atas mungkin akan terpecah menjadi beberapa baris untuk baris keempat karena terlalu panjang. Namun pada berkas masukan sebenarnya semua ciri-ciri terpisah berada di baris terpisah.

Ciri-ciri Pokémon yang diberikan diambil dari data lengkap Charmander. Bandingkan contoh masukan di atas dengan data lengkap Charmander (*substring* sebagai contoh masukan di-*highlight* kuning. Sama seperti penjelasan di contoh masukan, tampilan ciri-ciri di bawah mungkin terpecah menjadi beberapa baris).

Charmander 16

Charmander is a Fire-type Pokemon.

It evolves into Charmeleon starting at level 16, which evolves into Charizard starting at level 36.

Along with Bulbasaur and Squirtle, Charmander is one of three starter Pokemon of Kanto available at the beginning of Pokemon Red, Green, Blue, FireRed, and LeafGreen.

Charmander is a bipedal, reptilian Pokemon with an orange body, though its underside and soles are cream-colored. It has two small fangs visible in its upper and lower jaws and blue eyes. Its arms and legs are short with four fingers and three clawed toes. A fire burns at the tip of this Pokemon's slender tail, and has blazed there since Charmander's birth. The flame can be used as an indication of Charmander's health and mood, burning brightly when the Pokemon is strong, weakly when it is exhausted, wavering when it is happy, and blazing when it is enraged. It is said that Charmander dies if its flame goes out.

Charmander can be found in hot, mountainous areas. However, it is found far more often in the ownership of Trainers. As shown in Pokemon Snap, Charmander exhibits pack behavior, calling others of its species if it finds food.

Obviously prefers hot places. When it rains, steam is said to spout from the tip of its tail.

The flame at the tip of its tail makes a sound as it burns. You can only hear it in quiet places.

Even the newborns have flaming tails. Unfamiliar with fire, babies are said to accidentally burn themselves.

The flame on its tail shows the strength of its life force. If it is weak, the flame also burns weakly.

The flame on its tail indicates Charmander's life force. If it is healthy, the flame burns brightly.

If it's healthy, the flame on the tip of its tail will burn vigorously, even if it gets a bit wet.

The flame that burns at the tip of its tail is an indication of its emotions. The flame wavers when Charmander is enjoying itself. If the Pokemon becomes enraged, the flame burns fiercely.

The flame that burns at the tip of its tail is an indication of its emotions. The flame wavers when Charmander is happy, and blazes when it is enraged.

From the time it is born, a flame burns at the tip of its tail. Its life would end if the flame were to go out.

It has a preference for hot things. When it rains, steam is said to spout from the tip of its tail.

The fire on the tip of its tail is a measure of its life. If healthy, its tail burns intensely.

Berikut adalah gambar Charmander sebagai ilustrasi visual.



## Panduan Membaca *Database* Lengkap Pokémon

Seperti yang disebutkan di bagian format masukan, *database* Pokémon yang lengkap dapat diunduh [di sini](#). Berkas ini dapat Anda gunakan sebagai panduan untuk menentukan nama Pokémon berdasarkan ciri-ciri yang diberikan. Semua data yang terdapat dalam berkas tersebut diambil dari situs [Bulbapedia](#).

Berikut adalah format dari *database* Pokémon. Baris pertama berisi sebuah bilangan bulat  $P$  yang menyatakan banyaknya Pokémon yang diberikan data lengkapnya. Berikutnya terdapat  $P$  set data yang dijelaskan sebagai berikut. Tiap set data diawali oleh nama Pokémon tersebut dan banyaknya data lengkap dari Pokémon tersebut yang diberikan dalam satu baris.

Berikutnya diikuti oleh data lengkap dari Pokémon tersebut, dipisah per baris. Sebagai ilustrasi berikut ada struktur dari berkas *database* Pokémon.

```
P
<nama pokémon ke-1><spasi><D1>
<data pokémon 1 ke-1>
<data pokémon 1 ke-2>
...
<data pokémon 1 ke-D1>
<nama pokémon ke-2><spasi><D2>
<data pokémon 2 ke-1>
<data pokémon 2 ke-2>
...
<data pokémon 2 ke-D2>
...
<nama pokémon ke-P><spasi><DP>
<data pokémon P ke-1>
<data pokémon P ke-2>
...
<data pokémon P ke-DP>
```

## Pembagian Subsoal

Pada semua subsoal berlaku:

- $4 \leq N \leq 10$
- $1 \leq |S| \leq 150$
- Dijamin nama Pokémon tersebut tidak akan muncul pada ciri-ciri Pokémon yang diberikan.

### Subsoal 1 (3 poin)

- Hanya terdapat sebuah kasus uji untuk subsoal ini, yang dapat diunduh [di sini](#).

### Subsoal 2 (4 poin)

- Hanya terdapat sebuah kasus uji untuk subsoal ini, yang dapat diunduh [di sini](#).

### Subsoal 3 (5 poin)

- Hanya terdapat sebuah kasus uji untuk subsoal ini, yang dapat diunduh [di sini](#).

### Subsoal 4 (4 poin)

- Hanya terdapat sebuah kasus uji untuk subsoal ini, yang dapat diunduh [di sini](#).

### Subsoal 5 (3 poin)

- Hanya terdapat sebuah kasus uji untuk subsoal ini, yang dapat diunduh [di sini](#).

### Subsoal 6 (hingga 81 poin)

- Untuk subsoal ini, terdapat 45 kasus uji. Berbeda dengan model subsoal umum, pada subsoal ini semua kasus uji akan dinilai secara independen (Anda akan tetap mendapat nilai untuk kasus uji yang benar walaupun ada beberapa kasus uji yang salah).
- Bobot tiap kasus uji adalah 1,8 poin.

### Catatan

Untuk menghindari terjadinya *error* pada eksekusi program, semua karakter "é" pada berkas masukan maupun berkas *database* Pokémon telah diganti menjadi karakter "e" biasa.

Semua Pokémon dengan karakter khusus pada namanya telah dihilangkan dari berkas *database* Pokémon sehingga tidak akan diujikan lewat kasus uji. Secara lengkap, berikut adalah daftar nama Pokémon yang dihilangkan walaupun berasal dari generasi pertama.

- Farfetch'd
- Nidoran♂
- Nidoran♀
- Mr. Mime

sehingga dijamin setiap nama Pokémon hanya terdiri dari alfabet huruf kapital dan huruf kecil.

Pokémon dan nama/gambar setiap karakter adalah merek dagang dan hak cipta dari pemiliknya masing-masing. Kami tidak berafiliasi dengan **Nintendo**, **Pokémon Company**, **Creatures Inc**, **Game Freak**, atau **Bulbapedia**. Tidak ada maksud untuk melanggar hak cipta atau merek dagang pelanggaran.

# Stupa Borobudur

Batas Waktu	1 detik
Batas Memori	32 MB

## Deskripsi

Pak Dengklek mengunjungi kawasan wisata Candi Borobudur. Kawasan ini menarik karena banyak terdapat stupa yang indah. Sang penjaga berkata bahwa ada keajaiban di balik bilangan bulat yang terjadi pada kawasan tersebut. Untuk setiap bilangan bulat positif  $X$ , terdapat tepat 0 atau  $X$  buah stupa setinggi  $X$  meter.

Pak Dengklek, yang tidak sempat berkeliling ke seluruh kawasan, hanya sempat melihat  $N$  stupa. Tinggi dari stupa ke- $i$  yang dilihatnya adalah  $A_i$  meter. Pak Dengklek pun penasaran mengenai banyaknya stupa di kawasan tersebut.



Tentukan banyaknya stupa minimum di kawasan wisata tersebut!

## Format Masukan

Pada awalnya, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.
- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, dan berisi '.' jika bukan.
- Untuk setiap nilai  $i$  di antara 1 hingga banyaknya subsoal, berlaku:
  - jika kasus uji tersebut memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi  $i$ , atau
  - jika kasus uji tersebut tidak memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0 . . 345, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Baris kedua berisi sebuah bilangan bulat  $N$ . Baris berikutnya berisi  $N$  buah bilangan bulat  $A_1, A_2, \dots, A_N$ .

## Format Keluaran

Sebuah baris berisi banyaknya stupa minimum. Apabila masukan tidak sesuai dengan perkataan sang penjaga (mungkin saja Pak Dengklek salah lihat!), keluarkan sebuah baris berisi -1.

### Contoh Masukan 1

```
0..34567
5
2 1 4 2 3
```

### Contoh Keluaran 1

```
10
```

### Contoh Masukan 2

```
0..34567
4
2 9 2 2
```

### Contoh Keluaran 2

```
-1
```

## Penjelasan

Untuk contoh masukan pertama, di kawasan tersebut terdapat:

- 1 buah stupa setinggi 1 meter
- 2 buah stupa setinggi 2 meter
- 3 buah stupa setinggi 3 meter
- 4 buah stupa setinggi 4 meter

sehingga terdapat total 10 buah stupa.

Untuk contoh masukan kedua, Pak Dengklek melihat 3 buah stupa setinggi 2 meter. Hal ini tidak sesuai dengan perkataan sang penjaga, oleh karena itu Pak Dengklek pasti salah lihat.

## Subsoal

### Subsoal 1 (7 poin)

- $N = 10$
- $A_i = i$
- Kasus uji dapat diunduh [di sini](#).

### Subsoal 2 (11 poin)

- $N = 8$

- $1 \leq A_i \leq 10$
- Kasus uji dapat diunduh [di sini](#).

### Subsoal 3 (7 poin)

- $N = 1$
- $1 \leq A_1 \leq 1.000$

### Subsoal 4 (11 poin)

- $1 \leq N \leq 1.000$
- $1 \leq A_1 = A_2 = \dots = A_N \leq 1.000$

### Subsoal 5 (17 poin)

- $1 \leq N \leq 1.000$
- $1 \leq A_i \leq 1.000$

### Subsoal 6 (20 poin)

- $1 \leq N \leq 100.000$
- $1 \leq A_i \leq 100.000$

### Subsoal 7 (27 poin)

- $1 \leq N \leq 100.000$
- $1 \leq A_i \leq 1.000.000.000$

# Cat Rumah

Batas Waktu	1 detik
Batas Memori	256 MB

## Deskripsi

Hari ini Pak Dengklek ingin mengecat rumahnya dengan warna favorit bebek-bebeknya. Karena Pak Dengklek hanya memiliki kaleng-kaleng cat dengan warna dasar (merah, kuning, dan biru), maka Pak Dengklek menyiapkan sebuah ember dengan kapasitas maksimal 1.000 cc untuk mencampurkan ketiga warna dasar tersebut menjadi warna favorit bebek-bebeknya.

Pada suatu waktu, Pak Dengklek dapat menambahkan suatu warna dasar sebanyak sejumlah cc ke dalam embernya kemudian mencampurkannya. Perlu diperhatikan bahwa campuran warna tersebut tidak boleh melebihi kapasitas ember. Karena cat yang dipakai Pak Dengklek mudah mengering, maka Pak Dengklek hanya dapat melakukan proses menambahkan dan mencampurkan ini maksimal sebanyak 100 kali. Setelah melakukan proses menambahkan dan mencampurkan ini, campuran warna di ember akan dipakai untuk mengecat rumah.

Warna favorit bebek-bebek Pak Dengklek didapat dengan mencampurkan  $M$  cc warna Merah,  $K$  cc warna Kuning, dan  $B$  cc warna Biru dengan ketentuan  $M + K + B = 1.000$ . Perlu diperhatikan bahwa nilai  $M$ ,  $K$ , dan  $B$  tidak dijamin berupa bilangan bulat.

Sayangnya, bebek-bebek Pak Dengklek tidak mengetahui nama dari warna favorit mereka sehingga mereka hanya dapat memberi tahu nilai kemiripan antara warna favorit mereka dengan warna yang berada pada ember Pak Dengklek. Nilai kemiripan tersebut dihitung dengan ketentuan berikut:

- Dimisalkan campuran warna di ember Pak Dengklek saat ini yaitu  $X$  cc warna merah,  $Y$  cc warna Kuning, dan  $Z$  cc warna Biru.
- Jarak warna antara warna di ember Pak Dengklek dengan warna favorit Bebek ( $W$ ) dihitung dengan meminimalkan nilai  $(|kX - M| + |kY - K| + |kZ - B|)$  untuk suatu nilai riil  $k$ .
- Sebagai catatan, pasti setidaknya ada satu dari ketiga nilai  $|kX - M|$ ,  $|kY - K|$ , atau  $|kZ - B|$  yang sama dengan nol untuk mendapatkan nilai  $W$ .
- Nilai kemiripan ( $F$ ) dihitung dengan formula:

Sebagai contoh, apabila nilai  $M$ ,  $K$ , dan  $B$  berturut-turut bernilai 200, 600, dan 200; serta nilai  $X$ ,  $Y$ , dan  $Z$  berturut-turut bernilai 125, 200, dan 50; maka:

- Untuk mendapatkan nilai  $W$ , perhatikan bahwa salah satu dari ketiga nilai adalah 0.
  - Apabila nilai  $|kX - M| = 0$ , maka  $k = 1,6$ . Sehingga nilai  $(|kX - M| + |kY - K| + |kZ - B|) = 400$ .

- Apabila nilai  $|kY - K| = 0$ , maka  $k = 3$ . Sehingga nilai  $(|kX - M| + |kY - K| + |kZ - B|) = 225$ .
- Apabila nilai  $|kZ - B| = 0$ , maka  $k = 4$ . Sehingga nilai  $(|kX - M| + |kY - K| + |kZ - B|) = 500$ .
- Nilai  $W$  adalah nilai minimum dari ketiganya, sehingga  $W = 225$ .
- Nilai  $F$  dihitung dengan formula  $F = 40$ .

Sehingga pada setiap Pak Dengklek melakukan proses menambahkan dan mencampurkan (yang telah dijelaskan pada paragraf 2), bebek-bebek Pak Dengklek akan memberi tahu nilai  $F$  kepada Pak Dengklek.

Bantulah Pak Dengklek untuk mengecat rumahnya dengan warna semirip mungkin dengan warna favorit bebek-bebeknya, yaitu mendapatkan nilai  $F$  sebesar mungkin!

## Format Interaksi

Pada awalnya, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.
- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, atau berisi '.' jika bukan.
- Untuk setiap nilai  $i$  di antara 1 hingga banyaknya subsoal, berlaku:
  - jika kasus uji tersebut memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi  $i$ , atau
  - jika kasus uji tersebut tidak memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0 . . 345, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Selama interaksi berlangsung, program Anda diharuskan mencetak salah satu dari dua perintah: TAMBAH [WARNA] [TAKARAN] dan SELESAI.

Cetaklah TAMBAH [WARNA] [TAKARAN], jika Anda hendak menambahkan [TAKARAN] cc cat berwarna [WARNA] ke dalam ember. Volume cairan setelah ditambahkan tidak boleh melebihi kapasitas ember yakni 1.000 cc dan merupakan bilangan riil positif dengan 4 angka di belakang koma. Warna yang boleh ditambahkan hanyalah warna MERAH, KUNING, atau pun BIRU. Perhatikan bahwa perintah ini hanya dapat dipanggil maksimal 100 kali. Setelah mencetak perintah ini, program anda akan menerima input sebuah bilangan riil  $F$  persen dengan 4 angka di belakang koma sesuai dengan deskripsi soal.

Cetaklah SELESAI, jika Anda hendak menyelesaikan proses menambahkan dan mencampurkan. Campuran cat yang berada di dalam ember akan digunakan Pak Dengklek

untuk mengecat rumahnya. Setelah mencetak perintah ini, interaksi akan selesai dan penilaian akan dilakukan.

### Contoh Interaksi

Keluaran Program Peserta (Pak Dengklek)	Umpan Balik <i>Grader</i> (Bebek)	Informasi Tambahan
	0..3	
TAMBAH MERAH 100.0000		
	0.0000	W = 800.0000
TAMBAH KUNING 123.4500		
	11.8159	W = 486.0267
TAMBAH KUNING 76.5500		
	30.7180	W = 300.0000
TAMBAH BIRU 50.0000		
	51.0102	W = 150.0000
TAMBAH MERAH 25.0000		
	40.0000	W = 225.0000
SELESAI		
	(interaksi selesai)	

### Penjelasan Contoh Interaksi

Pada kasus tersebut, nilai M, K, dan B berturut-turut bernilai 200, 600, dan 200.

Pada akhirnya, warna yang digunakan Pak Dengklek untuk mengecat rumahnya merupakan campuran dari :

- X = 125 cc warna Merah,
- Y = 200 cc warna Kuning, dan
- Z = 50 cc warna Biru.

Nilai yang didapatkan peserta adalah nilai F terakhir yaitu 40.

### Subsoal

Tidak ada subsoal pada soal ini. Poin total yang Anda peroleh adalah akumulasi nilai dari 100 buah kasus uji. Semua kasus uji akan termasuk di dalam setidaknya 1 dari 3 himpunan kasus uji berikut.

#### Himpunan Kasus Uji 1 (20 kasus uji)

- $M = K$
- Nilai M, K, dan B adalah sebuah bilangan bulat.



### Himpunan Kasus Uji 2 (30 kasus uji)

- $B = 500$
- Nilai  $M$ ,  $K$ , dan  $B$  tidak dipastikan sebuah bilangan bulat.

### Himpunan Kasus Uji 3 (50 kasus uji)

- Nilai  $M$ ,  $K$ , dan  $B$  tidak dipastikan sebuah bilangan bulat.

Untuk membantu Anda memahami interaksi, disediakan game yang dapat diakses [di sini](#). Kasus uji yang diberikan pada game ini hanyalah untuk visualisasi dan tidak termasuk dalam penilaian seperti pada game di soal interaktif sebelumnya.

### Catatan

Yang perlu diperhatikan adalah bahwa untuk tipe soal interaktif seperti ini, Anda harus selalu memberikan perintah `"fflush(stdout);"` (bagi pengguna C/C++) atau `"flush(output);"` (bagi pengguna PASCAL) setiap kali Anda mencetak keluaran (dengan kata lain, setiap kali ada perintah mencetak keluaran misalnya `write`, `writeln`, `printf`, `cout`, atau `puts`, tepat di bawahnya harus ada perintah `fflush/flush`).

Sebagai contoh, berikut adalah contoh *source code* dalam bahasa Pascal yang selalu hanya menambahkan 100 cc cat untuk masing-masing warna ke dalam ember.

```
var subsoal: string;
    F: double;
    m, k, b: double;

begin
    m := 100;
    k := 100;
    b := 100;

    readln(subsoal);

    writeln('TAMBAH MERAH ', m:0:4);
    flush(output);

    // baca keluran dari grader, meskipun tidak digunakan
    readln(F);

    writeln('TAMBAH KUNING ', k:0:4);
    flush(output);

    // baca keluran dari grader, meskipun tidak digunakan
    readln(F);

    writeln('TAMBAH BIRU ', b:0:4);
    flush(output);

    // baca keluran dari grader, meskipun tidak digunakan
    readln(F);

    writeln('SELESAI');
end.
```

Dan berikut adalah contoh *source code* dalam bahasa C dan C++.

```
#include <stdio.h>
#include <string.h>

char subsoal[100];
double F;
double m, k, b;

int main() {
    m = 100;
    k = 100;
    b = 100;

    scanf("%s", subsoal);

    printf("TAMBAH MERAH %.4lf\n", m);
    fflush(stdout);

    /* baca keluran dari grader, meskipun tidak digunakan */
    scanf("%lf", &F);

    printf("TAMBAH KUNING %.4lf\n", k);
    fflush(stdout);

    /* baca keluran dari grader, meskipun tidak digunakan */
    scanf("%lf", &F);

    printf("TAMBAH BIRU %.4lf\n", b);
    fflush(stdout);

    /* baca keluran dari grader, meskipun tidak digunakan */
    scanf("%lf", &F);

    printf("SELESAI\n");
    fflush(stdout);

    return 0;
}
```

## Peringatan

Apabila program Anda melakukan salah satu dari hal-hal di bawah ini:

- melakukan tindakan tidak sesuai format sehingga tidak dikenali oleh *grader*,
- memanggil perintah TAMBAH lebih dari 100 kali,
- menambahkan cat hingga melebihi kapasitas ember, atau
- menambahkan cat dengan takaran bilangan negatif.

maka program Anda akan dihentikan secara otomatis dan Anda memperoleh nilai 0 pada kasus uji yang bersangkutan. Meskipun memperoleh nilai 0, perhatikan bahwa Anda **tetap mendapatkan *verdict Accepted*** untuk kasus uji tersebut.

# Reduksi Pesan

Batas Waktu	1 detik
Batas Memori	32 MB

## Deskripsi

Pak Dengklek bersama tim risetnya sedang mengembangkan program untuk mereduksi pesan yang berupa sederetan huruf alfabet kapital (A – Z). Program ini sudah selesai dikembangkan, dan kini memasuki tahap pengujian. Sayangnya, Pak Dengklek dan tim kebingungan bagaimana memastikan bahwa program yang mereka kembangkan bekerja dengan baik. Akhirnya ia meminta bantuan Anda untuk membuat program yang dapat membantu mengoreksi pekerjaan Pak Dengklek.

Program yang diinginkan Pak Dengklek sederhana. Anda akan diberikan sebuah untai pesan berisi simbol A – Z (semua kapital) tanpa spasi. Setiap huruf akan dinomori 1 sampai N, dengan N adalah panjang dari pesan itu sendiri. Kemudian Pak Dengklek akan memberikan beberapa pertanyaan yang dinyatakan oleh pasangan bilangan i dan j. Program anda harus mengeluarkan hasil reduksi *substring* dari posisi ke-i hingga posisi ke-j. Adapun aturan reduksi yang diinginkan adalah sebagai berikut:

- Untuk setiap sekumpulan huruf yang saling bersebelahan dan memiliki simbol yang sama, hapus semua kecuali satu. Dengan kata lain, pada hasil reduksi pesan, tidak ada dua huruf yang sama akan saling bersebelahan. Contoh: HAALLLLLOOOOO → HALO, MATARAM → MATARAM, dan AAAAA → A.
- Hitung panjang pesan setelah dilakukan proses reduksi. Keluarkan angka ini, karena ini merupakan informasi yang penting untuk Pak Dengklek. Contoh: HAALLLLLOOOOO → HALO (4 huruf).
- Jika panjang pesan yang tereduksi **lebih kecil dari 10**, cetak juga pesan hasil reduksi tersebut.

Bisakah Anda menyelesaikan program yang diinginkan Pak Dengklek? Sebagai catatan, Pak Dengklek hanya memberikan Anda sedikit waktu karena sebentar lagi hasil penelitian mereka akan dipublikasikan.

## Format Masukan

Pada baris pertama, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.
- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, atau berisi '.' jika bukan.
- Untuk setiap nilai i di antara 1 hingga banyaknya subsoal, berlaku:

- jika kasus uji tersebut memenuhi batasan subsoal ke-i, maka karakter ke-i berisi i, atau
- jika kasus uji tersebut tidak memenuhi batasan subsoal ke-i, maka karakter ke-i berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0..345, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Baris kedua berisi 2 bilangan, N dan Q, dimana N adalah panjang pesan dan Q adalah banyaknya pertanyaan.

Baris berikutnya adalah sebuah pesan yang terdiri dari simbol A – Z, sepanjang N.

Q baris berikutnya berisi 2 buah bilangan i dan j.

## Format Keluaran

Untuk setiap pertanyaan, keluarkan panjang pesan setelah melakukan reduksi potongan pesan dari posisi ke-i hingga posisi ke-j. Apabila panjang pesan lebih kecil dari 10, maka tampilkan hasil reduksinya di baris yang sama dipisahkan oleh spasi.

## Contoh Masukan

```
0..345
20 5
AAABBCCCCCQWERTYUIOP
1 1
1 3
2 9
1 20
18 20
```

## Contoh Keluaran

```
1 A
1 A
3 ABC
13
3 IOP
```

## Subsoal

### Subsoal 1 (7 poin)

- $N = 15$
- $Q = 10$
- Kasus uji dapat diunduh [di sini](#).

### Subsoal 2 (11 poin)

- $N = 100$
- $Q = 10$
- Kasus uji dapat diunduh [di sini](#).

**Subsoal 3 (21 poin)**

- $1 \leq N \leq 100$
- $1 \leq Q \leq 100$

**Subsoal 4 (27 poin)**

- $1 \leq N \leq 100$
- $1 \leq Q \leq 100.000$

**Subsoal 5 (34 poin)**

- $1 \leq N \leq 100.000$
- $1 \leq Q \leq 100.000$

**Peringatan**

Bagi pengguna C/C++ diwajibkan untuk menggunakan perintah `scanf()` (bukan `cin()`) untuk membaca masukan pada soal ini agar tidak terjebak *Time Limit Exceeded*.

# Pelontar Bebek

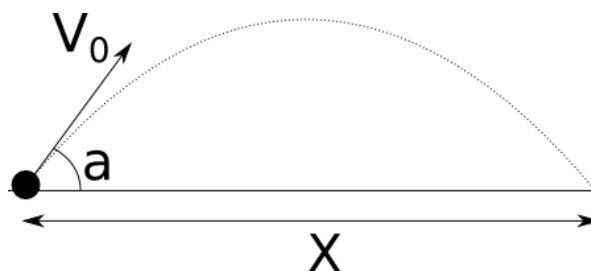
Batas Waktu	1 detik
Batas Memori	32 MB

## Deskripsi

Kwek dan Kwak, seperti layaknya bebek lain sedang seru-serunya bermain permainan “Rocket Duck”. Pada Rocket Duck, pemain menjalankan seekor bebek bernama “Rocky” yang ingin mencari arti hidup, yang diyakini berada diujung belahan dunia. Di luar dugaan, Rocky mencari arti hidup dengan cara yang tidak biasa dan ironis: melempar dirinya sejauh mungkin dengan alat pelontar (dan berharap mencapai ujung belahan dunia). Tentu ini cara yang ekstrim dan berbahaya. Tapi siapa peduli, Rocky hanyalah sebuah tokoh imajiner dalam sebuah *game*, yang terpenting bahwa permainan ini berhasil memikat banyak bebek muda.

Semua bebek ambisius mengejar *high-score* di Rocket Duck, begitu pula Kwek dan Kwak. Mereka membayangkan namanya terpampang pada tabel *leaderboard*, yang menjadi sebuah kepuasan tersendiri bagi mereka. Namun Kwek dan Kwak punya kehidupan lain, mereka tidak bisa terus menerus bermain Rocket Duck. Sehingga kemampuan mereka tidak sejago pemain lainnya. Untungnya Kwek dan Kwak kenal dengan Anda, sang programmer handal yang konon dapat menyelesaikan berbagai masalah hanya dengan hentakan jari-jemari di atas *keyboard*. Kini mereka memerlukan bantuan kalian untuk mengejar *high-score* pada permainan Rocket Duck.

Rocket Duck adalah permainan 2D, di mana pemain berusaha mencapai jarak terjauh dengan menggunakan alat pelontar bebek. Alat ini dapat melontarkan bebek pada derajat dan kecepatan awal tertentu. Bebek kemudian bergerak secara parabola, hingga akhirnya jatuh ke tanah. Nilai yang didapat dari pemain adalah jarak yang dicapai. Makin jauh jarak yang dicapai, makin tinggi nilai yang didapat. Mekanisme lontaran roket mengikuti mekanisme dasar gerak parabolik. Berikut ini adalah ilustrasi mekanisme dari gerak parabolik:



Untuk Pemain, akan disediakan beberapa pilihan roda gigi untuk merakit pelontar bebek. Setiap roda gigi memiliki dua properti, yaitu nilai kecepatan  $K$  dan nilai sudut  $S$ . Sebuah pelontar bebek nantinya akan terdiri dari beberapa roda gigi yang terpasang pada mesin. Sudut akhir dari mesin pelontar ini adalah total nilai sudut dari semua roda gigi yang dipasang. Kecepatan dari mesin pelontar ini adalah kecepatan maksimum dari semua roda gigi yang dipasang.

Sebagai tambahan, mesin pelontar ini tidak bisa menembak dengan sudut yang lebih besar dari 180 derajat. Artinya konfigurasi mesin yang menyebabkan hal tersebut terjadi tidak diperbolehkan. Perlu diperhatikan pula bahwa nilai pemain didapat dari jauhnya jarak. Artinya arah tembak dapat diabaikan (sudut kurang dari 90 derajat memiliki arah tembak berlawanan dengan sudut lebih dari 90 derajat). Jumlah roda gigi yang ada bersifat terbatas, artinya pemain hanya dapat memakai tiap roda gigi maksimal sekali saja.

Untuk membantu, berikut adalah rumus untuk menghitung jarak tembak  $X$  pada permainan Rocket Duck, diberikan nilai sudut  $a$  dan kecepatan awal  $V_0$ .

$$X = |2 \times V_0^2 \times \sin a \times \cos a \times 9.8|$$

Bantulah Kwek dan Kwak memilih konfigurasi roda gigi optimal, untuk mendapatkan jarak terjauh.

## Format Masukan

Pada baris pertama, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.
- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, atau berisi '.' jika bukan.
- Untuk setiap nilai  $i$  di antara 1 hingga banyaknya subsoal, berlaku:
  - jika kasus uji tersebut memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi  $i$ , atau
  - jika kasus uji tersebut tidak memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0 . . 345, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Baris kedua adalah bilangan  $N$  yang menyatakan banyaknya pilihan roda gigi yang tersedia.

$N$  baris berikutnya terdiri dari dua angka  $K_i$  dan  $S_i$ , yang menyatakan nilai kecepatan dan nilai sudut dari roda gigi yang bersesuaian. Nilai sudut diberikan dengan skala perbesaran  $10\times$  lipat sehingga nilai sudut 900 maksudnya adalah 90 derajat, nilai sudut 135 maksudnya adalah 13,5 derajat, dan sebagainya.

## Format Keluaran

Untuk tiap kasus uji, keluarkan sebuah bilangan riil hingga 12 angka di belakang penanda desimal yang menyatakan jarak terjauh yang dapat dicapai dengan konfigurasi roda gigi

optimal. Perbedaan perhitungan absolut atau perhitungan relatif di bawah 0,001 akan diterima.

## Contoh Masukan

```
0..3.567
3
50 160
105 320
101 290
```

## Contoh Keluaran

```
107453.118185065090
```

### Subsoal

**Pada semua subsoal, berlaku:**

- $1 \leq S_i \leq 1.800$
- $K_i$  dan  $S_i$  adalah bilangan bulat.

### Subsoal 1 (8 poin)

- $N = 5$
- $1 \leq K_i \leq 100$
- Nilai  $K$  tiap roda gigi adalah sama.
- Kasus uji dapat diunduh [di sini](#).

### Subsoal 2 (10 poin)

- $N = 5$
- $1 \leq K_i \leq 100$
- Kasus uji dapat diunduh [di sini](#).

### Subsoal 3 (13 poin)

- $1 \leq N \leq 10$
- $1 \leq K_i \leq 100$

### Subsoal 4 (14 poin)

- $1 \leq N \leq 50$
- $1 \leq K_i \leq 100$
- Nilai  $K$  tiap roda gigi adalah sama.

### Subsoal 5 (15 poin)

- $1 \leq N \leq 50$
- $1 \leq K_i \leq 100$



**Subsoal 6 (19 poin)**

- $1 \leq N \leq 1.000$
- $1 \leq K_i \leq 1.000$

**Subsoal 7 (21 poin)**

- $1 \leq N \leq 500.000$
- $1 \leq K_i \leq 1.000$

**Catatan**

Untuk melakukan perhitungan  $\sin(a)$  atau  $\cos(a)$ , terdapat fungsi yang sudah ada dan bisa Anda gunakan baik pada Pascal maupun pada C/C++. Namun, kedua fungsi itu menerima sudut bukan dalam derajat, melainkan dalam radian (satuan yang lain untuk sudut). Anda dapat menggunakan fungsi berikut untuk merubah derajat menjadi radian.

Jika menggunakan Pascal, Anda **wajib** menuliskan 'uses math' sebelum mendeklarasikan variabel. Contoh program:

```
uses math;
var
    sudut:double;

function toRadian(a:double):double;
begin
    toRadian := (a * arccos(-1)) / 180;
end;

begin
    sudut := 20; // perhatikan bahwa sudut ini tidak dalam perbesaran 10x
    lipat

    // menghitung sin(20)
    writeln(sin(toRadian(sudut))); // hasilnya: 3.4202014332566871E-0001

    // menghitung cos(20)
    writeln(cos(toRadian(sudut))); // hasilnya: 9.3969262078590839E-0001
end.
```

Jika menggunakan C++, Anda **wajib** menulis '#include <cmath>' pada bagian deklarasi *library* yang akan digunakan. Contoh program:

```
#include <cstdio>
#include <cmath>

double sudut;

double toRadian(double a){
    return (a * acos(-1)) / 180;
}

int main() {
    sudut = 20; /* perhatikan bahwa sudut ini tidak dalam perbesaran
    10x lipat */
```

```
/* menghitung sin(20) */
printf("%lf\n", sin(toRadian(sudut))); /* hasil: 0.342020 */

/* menghitung cos(20) */
printf("%lf\n", cos(toRadian(sudut))); /* hasil: 0.939693 */

return 0;
}
```

Jika menggunakan C, Anda **wajib** menulis '#include <math.h>' pada bagian deklarasi *library* yang akan digunakan. Contoh program:

```
#include <stdio.h>
#include <math.h>

double sudut;

double toRadian(double a){
    return (a * acos(-1)) / 180;
}

int main() {
    sudut = 20; /* perhatikan bahwa sudut ini tidak dalam perbesaran
10x lipat */

    /* menghitung sin(20) */
    printf("%lf\n", sin(toRadian(sudut))); /* hasil: 0.342020 */

    /* menghitung cos(20) */
    printf("%lf\n", cos(toRadian(sudut))); /* hasil: 0.939693 */

    return 0;
}
```

## Peringatan

Gunakan tipe data **double** untuk menggantikan tipe data real/float agar ketelitian bilangan riil menjadi lebih tinggi.

Perhatikan potongan-potongan kode program di atas sebagai contoh penggunaannya.

# Perang Dunia Ketiga

Batas Waktu	0,5 detik
Batas Memori	64 MB

## Deskripsi

Perang dunia ketiga sudah pecah! Kedamaian di Negara Bebek pun terancam. Pak Dengklek yang sangat cinta Negara Bebek pun membuat sebuah senjata canggih yang menggunakan teknologi laser dan diyakini mampu menghancurkan musuh sekuat apapun.

Senjata canggih Pak Dengklek memiliki sebuah layar dan 3 tombol:

1. Layar: Untuk menampilkan kekuatan laser saat ini (harus selalu berupa bilangan asli).
2. Tombol "Tembak": Untuk menembakkan laser ke musuh.
3. Tombol "Kali 2": Membuat laser menjadi 2 kali lebih kuat dari sebelumnya.
4. Tombol "Bagi 2": Membuat kekuatan laser menjadi setengah dari sebelumnya. Jika kekuatan laser sebelum dilakukan operasi ini berupa bilangan ganjil, maka tidak terjadi apa-apa.

Nilai awal kekuatan laser hanya dapat diatur sekali sebelum pasukan musuh datang, karena untuk mengganti kekuatan laser membutuhkan waktu yang sangat lama (kecuali menggunakan tombol "Kali 2" dan tombol "Bagi 2"). Perhatikan bahwa nilai awal kekuatan laser ini harus lebih dari 0.

Senjata laser buatan Pak Dengklek ini tentunya menggunakan daya yang sangat besar. Pak Dengklek berkata bahwa untuk setiap satuan kekuatan laser yang ditembakkan menggunakan 1 unit energi. Tentunya kekuatan setiap pasukan musuh berbeda-beda, sehingga kekuatan laser yang diperlukan untuk menghancurkan mereka pun bisa jadi berbeda-beda. Untungnya, BINB (Badan Intelijen Negara Bebek) sudah mengetahui kekuatan musuh yang akan menyerang Negara Bebek dan sudah menghitung kekuatan laser yang diperlukan untuk menghancurkan masing-masing musuh.

Diberikan kekuatan laser minimum yang dibutuhkan untuk menghancurkan pasukan-pasukan negara musuh, berapa minimal unit energi yang harus terbuang untuk menghancurkan seluruh pasukan musuh? Anda harus menentukan nilai awal kekuatan laser, yang boleh ditentukan secara bebas.

Unit energi yang terbuang untuk menghancurkan satu pasukan musuh didefinisikan sebagai selisih dari unit energi minimal untuk menghancurkan pasukan tersebut dan unit energi yang digunakan Negara Bebek untuk menghancurkan pasukan tersebut. Sedangkan unit energi yang terbuang untuk menghancurkan seluruh musuh adalah jumlah dari unit energi yang terbuang untuk setiap pasukan musuh.

## Format Masukan

Pada awalnya, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.
- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, atau berisi '.' jika bukan.
- Untuk setiap nilai  $i$  di antara 1 hingga banyaknya subsoal, berlaku:
  - jika kasus uji tersebut memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi  $i$ , atau
  - jika kasus uji tersebut tidak memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0..345, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Baris kedua pada berkas masukan berisi sebuah bilangan bulat  $N$  yang menyatakan banyaknya pasukan musuh yang akan menyerang Negara Bebek.

Baris ketiga berisi  $N$  buah bilangan bulat non-negatif  $X_i$  yang menyatakan kekuatan laser minimum untuk menghancurkan pasukan ke- $i$ .

## Format Keluaran

Sebuah baris berisi sebuah bilangan bulat yang menyatakan unit energi yang terbuang minimal yang diperlukan untuk menghancurkan semua pasukan musuh.

### Contoh Masukan 1

```
0..3456
2
7 31
```

### Contoh Keluaran 1

```
2
```

### Contoh Masukan 2

```
0..3456
3
4 8 2
```

### Contoh Keluaran 2

## Penjelasan

Untuk contoh masukan nomor 1, kekuatan awal laser dapat diset sama dengan 8. Saat pasukan musuh nomor 1 datang, pasukan nomor 1 tersebut dapat langsung dihancurkan dengan menggunakan 8 unit energi (sehingga  $8 - 7 = 1$  unit energi terbang). Saat pasukan musuh nomor 2 datang, Pak Dengklek bisa menekan tombol "Kali 2" sebanyak 2 kali dan menghancurkannya dengan menggunakan 32 unit energi (sehingga  $32 - 31 = 1$  unit energi terbang). Total 2 unit energi terbang untuk menghancurkan pasukan musuh.

Untuk contoh masukan nomor 2, kekuatan awal laser diset sama dengan 4. Saat pasukan musuh nomor 1 datang, pasukan nomor 1 tersebut dapat langsung dihancurkan dengan menggunakan 4 unit energi (sehingga  $4 - 4 = 0$  unit energi terbang). Saat pasukan nomor 2 datang, Pak Dengklek bisa menekan tombol "Kali 2" sebanyak 1 kali dan menghancurkannya dengan menggunakan 8 unit energi (sehingga  $8 - 8 = 0$  unit energi terbang). Saat pasukan musuh nomor 3 datang, kita bisa menekan tombol "Bagi 2" sebanyak 2 kali dan menghancurkannya dengan menggunakan 2 unit energi (sehingga  $2 - 2 = 0$  unit energi terbang). Total 0 unit energi terbang untuk menghancurkan pasukan musuh.

## Subsoal

### Subsoal 1 (11 poin)

- $N = 10$
- $X_i = i$
- Kasus uji dapat diunduh [di sini](#).

### Subsoal 2 (7 poin)

- $N = 12$
- $0 \leq X_i \leq 10$
- Kasus uji dapat diunduh [di sini](#).

### Subsoal 3 (23 poin)

- $1 \leq N \leq 100$
- $0 \leq X_i \leq 100$

### Subsoal 4 (12 poin)

- $1 \leq N \leq 3.000$
- $0 \leq X_i \leq 3.000$

### Subsoal 5 (31 poin)

- $1 \leq N \leq 100.000$
- $0 \leq X_i \leq 100.000$

### Subsoal 6 (16 poin)

- $1 \leq N \leq 1.000.000$
- $0 \leq X_i \leq 1.000.000$

### Peringatan

Bagi pengguna C/C++ diwajibkan untuk menggunakan perintah `scanf()` (bukan `cin()`) untuk membaca masukan pada soal ini agar tidak terjebak *Time Limit Exceeded*.

# Sang Pelompat

Batas Waktu	1 detik
Batas Memori	64 MB

## Deskripsi

Kwik adalah bebek Pak Dengklek yang senang menonton TV. Salah satu film favoritnya adalah serial “*The Indiana Duck*”. Serial ini mengisahkan seekor bebek yang bekerja sebagai arkeologi dan menemukan harta karun historis di seluruh dunia. Indiana Duck terkenal dengan atribut legendarisnya berupa topi koboi dan cambuk untuk membela diri dari serangan musuh.

Pada suatu hari, Kwik menemukan peta harta karun dalam kotak kecil di salah satu sudut gudang Pak Dengklek. Dalam peta itu tertulis jika seseorang mendaki turun melalui sumur tua di belakang gudang, maka ia akan menemui gua raksasa berukuran  $R \times C$  dengan lautan magma di bawahnya. Dari dasar lautan magma tersebut menyembul beberapa bongkahan batu keras yang dapat digunakan sebagai pijakan. Pada salah satu bongkahan batu terdapat harta karun yang sudah dijaga selama beberapa generasi keluarga Dengklek. Kwik sangat senang karena dia bisa berlagak meniru Indiana Duck tokoh idolanya.

Satu-satunya cara untuk berpindah dari suatu bongkahan batu ke bongkahan batu lainnya adalah dengan melompat (Kwik tidak boleh menyentuh lautan magma jika ingin kembali hidup-hidup). Selama berada dalam suatu bongkahan batu, Kwik dapat menjelajahi bongkahan batu tersebut tanpa perlu melompat (misalnya, untuk berpindah ke sisi lain kemudian baru melompat). 2 petak batu akan membentuk sebuah bongkahan batu besar jika kedua petak batu tersebut berbagi sisi.

Karena Kwik masih kecil, dia tidak dapat melakukan gerakan yang sulit. Kwik hanya bisa melompat untuk menyeberangi lautan magma secara garis lurus (tidak dapat berbelok di udara). Kwik juga hanya dapat melompat ke arah utara, timur, selatan, atau barat. Karena perjalanan panjang, Kwik akan menghemat tenaganya dan akan **selalu** mendarat di bongkahan batu terdekat yang ditemuinya di arah lompatan.

Kwik berencana mendapatkan harta karun yang disebutkan dalam peta Kwik pun menceritakan rencana hebatnya kepada Anda dan meminta agar Anda tidak melapor ke Pak Dengklek. Anda sangat khawatir dengan keselamatan Kwik, namun Anda juga tidak bisa menemani Kwik dalam petualangannya karena Anda tidak ingin ketinggalan acara “*Duck’s Got Talent*”. Karena itu Anda hendak membantu Kwik dengan menentukan berapa lompatan minimal yang harus dilakukan Kwik agar dapat sampai ke harta karun (tentunya, semakin sedikit melompat akan menghemat energi Kwik dan menjamin keselamatannya).

## Format Masukan

Pada awalnya, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.
- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, atau berisi '.' jika bukan.
- Untuk setiap nilai *i* di antara 1 hingga banyaknya subsoal, berlaku:
  - jika kasus uji tersebut memenuhi batasan subsoal ke-*i*, maka karakter ke-*i* berisi *i*, atau
  - jika kasus uji tersebut tidak memenuhi batasan subsoal ke-*i*, maka karakter ke-*i* berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0 . . 345, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Baris kedua dalam berkas masukan berisi 2 buah bilangan *R* dan *C* yang dipisahkan oleh tepat sebuah spasi. *R* baris berikutnya berisi *C* buah karakter yang menyatakan konfigurasi petak batu dan lokasi harta karun sesuai deskripsi dalam peta. Sebuah petak dalam gua raksasa tersebut hanya dapat berupa salah satu dari kemungkinan karakter di bawah ini.

1. '#' : petak tersebut merupakan petak batu yang bisa diinjak.
2. '.' : petak tersebut merupakan lautan magma yang tidak boleh diinjak.
3. 'S' : petak ini merupakan bagian akhir dari tangga pada sumur, atau dengan kata lain tempat Kwik memulai petualangannya. Petak ini pasti merupakan bagian dari petak batu yang bisa diinjak.
4. 'T' : petak ini tempat harta karun yang dimaksud. Petak ini pasti merupakan bagian dari petak batu yang diinjak.

## Format Keluaran

Cetak sebuah bilangan dalam 1 baris yang menyatakan berapa banyak lompatan **paling sedikit** yang harus dilakukan Kwik agar dapat sampai ke petak harta karun. Cetak -1 jika Kwik tidak akan pernah menemukan harta karun.

## Contoh Masukan

```
0 . . . . 56
7 7
# . . . . #T
# . . # . .
. . # # . . #
. . . # . . #
. # . # . . #
. # . . S . .
. # . . # . .
```

## Contoh Keluaran





- $1 \leq R, C \leq 100$
- Semua bongkahan batu terdiri dari tepat satu petak batu.

**Subsoal 4 (21 poin)**

- $1 \leq R, C \leq 100$
- Semua bongkahan batu berbentuk persegi panjang.

**Subsoal 5 (26 poin)**

- $1 \leq R, C \leq 100$

**Subsoal 6 (16 poin)**

- $1 \leq R, C \leq 1.000$

# Suten

Batas Waktu	1 detik
Batas Memori	32 MB

## Deskripsi

Tahukah anda apa itu suten? Suten, sut, suit (suwit), atau pingsut adalah cara mengundi untuk dua orang dengan cara mengadu jari untuk menentukan siapa yang menang.

Di Indonesia, jari yang dipergunakan dalam suten adalah ibu jari yang diumpamakan sebagai gajah, jari telunjuk yang diumpamakan sebagai manusia, dan jari kelingking yang diumpamakan sebagai semut. Dua orang yang bersuten secara serentak mengacungkan jari yang dipilihnya. Hasilnya seri terjadi bila kedua belah pihak yang bersuten mengacungkan jari yang berkekuatan sama, misalnya: kelingking melawan kelingking. Biasanya, apabila terjadi seri maka suten diulang hingga ada pihak yang menang, namun peraturan ini tidak berlaku untuk saat ini.

Jari yang menjadi pemenang suten:

- Ibu jari versus telunjuk: pemenang adalah ibu jari.
- Telunjuk versus kelingking: pemenang adalah telunjuk.
- Kelingking versus ibu jari: pemenang adalah kelingking.

Anda, sebagai salah satu finalis OSN Komputer 2014, diminta untuk menyelesaikan tantangan berikut.

Ada  $N$  anak yang bermain suten bersama-sama. Anak-anak ini memiliki jari favorit mereka masing-masing sehingga setiap melakukan suten, mereka selalu mengeluarkan jari favoritnya untuk diadu. Setiap anak melakukan suten dengan anak lainnya tepat sekali, sehingga ada  $(N \times (N - 1)) / 2$  buah pertandingan suten yang akan dilakukan.

Diberikan nilai  $N$ , Anda (dan program Anda) akan ditanyakan  $(N \times (N - 1)) / 2$  buah pertandingan suten dan menentukan siapa pemenangnya (atau menentukan adanya seri). Apabila Anda salah menjawab, maka secara otomatis Anda gagal menyelesaikan tantangan dan akan mendapatkan nilai 0.

Karena Anda tidak mengetahui jari favorit masing-masing anak, Anda diperbolehkan untuk memilih PASS atau memilih untuk tidak menjawab sebuah pertandingan. Apabila Anda memilih untuk PASS, maka anda akan diberikan hasil pertandingan tersebut. Anda diminta untuk memilih PASS **sesedikit mungkin**, dengan kata lain Anda diminta untuk menjawab setiap hasil pertandingan sebanyak mungkin.

## Format Interaksi

Pada awalnya, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.
- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, dan berisi '.' jika bukan.
- Untuk setiap nilai  $i$  di antara 1 hingga banyaknya subsoal, berlaku:
  - jika kasus uji tersebut memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi  $i$ , atau
  - jika kasus uji tersebut tidak memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0 . . 345, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Kemudian, program Anda akan menerima input sebuah bilangan  $N$  yaitu banyaknya anak yang bermain suten. Kemudian, sebanyak  $(N \times (N - 1)) / 2$  kali program Anda akan menerima input 2 buah bilangan  $X$  dan  $Y$  yang menandakan pertandingan suten antara anak ke- $X$  dan anak ke- $Y$ . Anda diminta untuk mencari tahu hasil pertandingan suten tersebut satu per satu.

Keluaran yang dapat Anda keluarkan adalah:

- Apabila pertandingan suten dimenangkan oleh anak ke- $Z$ , program Anda diminta untuk mengeluarkan output  $Z$  MENANG.
- Apabila pertandingan suten berlangsung seri, program Anda diminta untuk mengeluarkan output SERI.
- Apabila Anda memilih untuk PASS atau tidak menjawab, program Anda diminta untuk mengeluarkan output PASS. Apabila memilih PASS, maka program Anda akan menerima hasil pertandingan suten tersebut dalam bentuk  $Z$  MENANG ataupun SERI.

### Contoh Interaksi

Keluaran Program Peserta	Umpan Balik Grader
	0 . . . 567
	4
	1 2
SERI	

Keluaran Program Peserta	Umpan Balik <i>Grader</i>
	1 3
3 MENANG	
	1 4
1 MENANG	
	2 3
PASS	
	3 MENANG
	2 4
PASS	
	2 MENANG
	3 4
4 MENANG	
	(interaksi selesai)

### Penjelasan Contoh Interaksi

Pada kasus tersebut terdapat 4 anak yang bermain suten, sehingga terdapat 6 buah pertandingan. Pada kasus tersebut pula, jari favorit masing-masing anak adalah sebagai berikut:

- Anak ke-1 : Ibu Jari.
- Anak ke-2 : Ibu Jari.
- Anak ke-3 : Kelingking.
- Anak ke-4 : Telunjuk.

Karena tidak ada jawaban interaksi yang salah maka interaksi yang terjadi merupakan interaksi yang valid dan program tersebut berhasil menyelesaikan tantangan dengan melakukan 2 kali PASS.

### Subsoal

Pada semua subsoal, berlaku:

- $3 \leq N \leq 100$
- Dimisalkan P adalah banyaknya PASS paling banyak yang boleh dilakukan peserta.

#### Subsoal 1 (6 poin)

- $N = 5$
- $P = 6$
- Untuk setiap pertandingan, berlaku  $X < Y$ .
- Input pertandingan akan diurutkan dengan aturan berikut:
  - Pertandingan diurut mula-mula dari nilai X terkecil.
  - Apabila 2 buah pertandingan memiliki nilai X yang sama, maka diurut mula-mula dari nilai Y terkecil.

#### Subsoal 2 (12 poin)

- $N = 8$
- $P = 7$

Khusus untuk subsoal 1 dan subsoal 2:

- Hanya terdapat sebuah kasus uji (satu subsoal dinyatakan oleh satu kasus uji), yang dapat dimainkan [di sini](#).
- Anda boleh memainkan permainan ini berulang kali tanpa mendapatkan penalti.
- Jika Anda sudah memenangkan permainan untuk subsoal tertentu, Anda dapat memilih pilihan pada permainan untuk mengeluarkan *source code* yang dapat langsung Anda kirimkan ke *grader* dan menjawab dengan benar pada subsoal yang telah Anda menangkan.
- Anda tidak diwajibkan memainkan permainan ini untuk mengerjakan kedua subsoal ini. Anda diperbolehkan untuk menulis kode Anda sendiri untuk mengerjakan kedua subsoal ini.

#### Subsoal 3 (8 poin)

- $N = 3$
- $P = 2$

#### Subsoal 4 (13 poin)

- $P = (N \times (N - 1)) / 2 - N + 2$

#### Subsoal 5 (14 poin)

- $P = N - 1$
- Untuk setiap pertandingan, berlaku  $X < Y$
- Input pertandingan akan diurutkan dengan aturan berikut:
  - Pertandingan diurut mula-mula dari nilai X terkecil.
  - Apabila 2 buah pertandingan memiliki nilai X yang sama, maka diurut mula-mula dari nilai Y terkecil.

#### Subsoal 6 (19 poin)

- $P = N - 1$
- Untuk  $N - 1$  pertandingan pertama, berlaku  $X < Y$ .
- Untuk  $N - 1$  pertandingan pertama, pada pertandingan ke- $i$  nilai  $Y = i + 1$ .

**Subsoal 7 (28 poin)**

- $P = N - 1$

**Catatan**

Yang perlu diperhatikan adalah bahwa untuk tipe soal interaktif seperti ini, Anda harus selalu memberikan perintah `"fflush(stdout);"` (bagi pengguna C/C++) atau `"flush(output);"` (bagi pengguna PASCAL) setiap kali Anda mencetak keluaran (dengan kata lain, setiap kali ada perintah mencetak keluaran misalnya `write`, `writeln`, `printf`, `cout`, atau `puts`, tepat di bawahnya harus ada perintah `fflush/flush`).

Sebagai contoh, berikut adalah contoh *source code* dalam bahasa Pascal yang akan selalu menjawab PASS tanpa mempedulikan nilai  $N$  yang diberikan.

```
var subsoal, jawaban: string;
    N, X, Y, i: longint;

begin
  readln(subsoal);
  readln(N);

  for i := 1 to (N*(N-1) div 2) do begin
    readln(X,Y);

    writeln('PASS');
    flush(output);

    readln(jawaban);
  end;
end.
```

Dan berikut adalah contoh *source code* dalam bahasa C++.

```
#include <cstdio>
#include <cstring>

char subsoal[100], jawaban[100];
int N, X, Y, i;

int main() {
  gets(subsoal);
  scanf("%d\n", N);

  for (i = 1; i <= N*(N-1)/2; i++) {
    scanf("%d %d\n", X, Y);

    printf("PASS\n");
    fflush(stdout);

    gets(jawaban);
  }
}
```

```
    return 0;  
}
```

## Peringatan

Apabila program Anda melakukan salah satu dari hal-hal di bawah ini:

- melakukan tindakan tidak sesuai format sehingga tidak dikenali oleh *grader*,
- salah menjawab di suatu hasil pertandingan, atau
- selesai menjawab semua hasil pertandingan namun memilih PASS lebih dari P kali,

maka program Anda akan dihentikan secara otomatis dan Anda mendapatkan *verdict Wrong Answer* pada kasus uji yang bersangkutan.



# Hiasan Dinding

Batas Waktu	1 detik
Batas Memori	64 MB

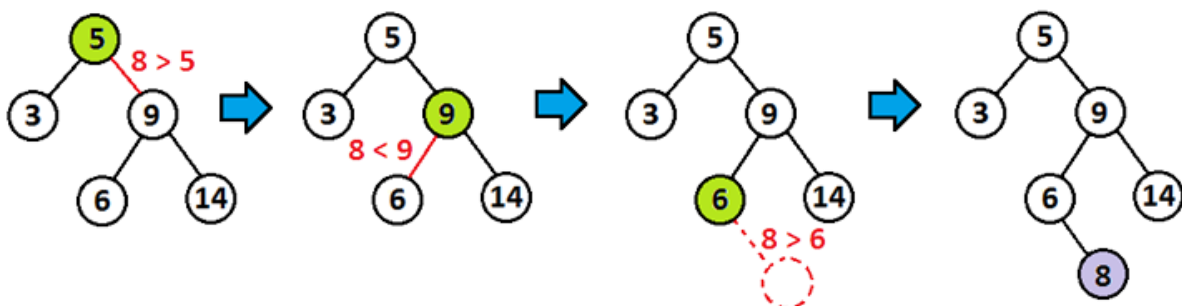
## Deskripsi

Untuk menyambut OSN 2014, Pak Dengklek berencana membuat sebuah hiasan dinding di laboratorium tempat Bidang Informatika dilaksanakan. Hiasan ini cukup sederhana, yakni hanya terdiri atas paku-paku dan untaian tali. Pak Dengklek memiliki  $N$  buah paku untuk membuat hiasan ini. Paku-paku ini dinomori secara unik dari 1 sampai dengan  $N$ .

Cara membuat hiasan dinding ini adalah sebagai berikut. Mula-mula, Pak Dengklek membariskan paku-paku tersebut dalam sebuah barisan. Lalu, Pak Dengklek memasang paku pertama pada dinding. Untuk setiap paku kedua sampai dengan ke- $N$  secara berurutan, Pak Dengklek melakukan langkah-langkah berikut:

1. Tinjau paku paling atas pada dinding.
2. Misalkan  $X$  = paku yang sedang ditinjau, dan  $Y$  = paku yang ingin dipasang.
  - a) Jika nomor dari  $Y <$  nomor dari  $X$ :
    - i. Jika  $X$  tidak memiliki paku yang terhubung dengan tali di sebelah kiri bawahnya, pasang  $Y$  persis disebelah kiri bawah  $X$ , dan hubungkan keduanya menggunakan tali. Pemasangan paku dianggap selesai.
    - ii. Jika  $X$  memiliki paku yang terhubung dengan tali di sebelah kiri bawahnya, tinjau paku tersebut dan kembali ke Langkah 2.
  - b) Jika nomor dari  $Y >$  nomor dari  $X$ :
    - i. (i) Jika  $X$  tidak memiliki paku yang terhubung dengan tali di sebelah kanan bawahnya, pasang  $Y$  persis di sebelah kanan bawah  $X$ , dan hubungkan keduanya menggunakan tali. Pemasangan paku dianggap selesai.
    - ii. Jika  $X$  memiliki paku yang terhubung dengan tali di sebelah kanan bawahnya, tinjau paku tersebut dan kembali ke Langkah 2.

Sebagai contoh, misalkan Pak Dengklek ingin memasang paku keenam bernomor 8. Berikut adalah ilustrasi langkah-langkah pemasangan paku ini:



Pak Dengklek sudah memiliki sebuah desain hiasan dinding yang dibuat oleh panitia dekorasi. Ia ingin segera memasang paku-paku agar terbentuk hiasan dinding tersebut. Tentu saja, mula-mula ia harus membariskan paku-paku tersebut. Mungkin saja ada lebih dari satu kemungkinan barisan yang menghasilkan hiasan dinding tersebut. Pak Dengklek, sebagai seorang yang senang dengan teka-teki logika, justru penasaran dengan pertanyaan berikut: jika barisan-barisan yang mungkin menghasilkan hiasan dinding tersebut diurutkan dengan urutan leksikografis, barisan apa yang berada pada posisi ke-K?

Catatan: barisan paku A lebih kecil daripada barisan paku B secara leksikografis, apabila pada posisi pertama di mana A dan B berbeda, nomor paku pada posisi tersebut di A lebih kecil daripada nomor paku pada posisi tersebut di B.

Bantulah Pak Dengklek menjawab pertanyaan tersebut, agar ia dapat segera memasang paku-paku tersebut!

## Format Masukan

Pada awalnya, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.
- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, dan berisi '.' jika bukan.
- Untuk setiap nilai  $i$  di antara 1 hingga banyaknya subsoal, berlaku:
  - jika kasus uji tersebut memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi  $i$ , atau
  - jika kasus uji tersebut tidak memenuhi batasan subsoal ke- $i$ , maka karakter ke- $i$  berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0 . . 345, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Baris kedua berisi dua buah bilangan bulat  $N$  dan  $K$ .  $N-1$  baris berikutnya mendeskripsikan sebuah desain hiasan dinding. Masing-masing baris berisi tiga buah bilangan bulat  $u$ ,  $v$ , dan  $t$ :

- Apabila  $t = 0$ , maka paku bernomor  $v$  terhubung dengan tali di sebelah kiri bawah dengan paku bernomor  $u$ .
- Apabila  $t = 1$ , maka paku bernomor  $v$  terhubung dengan tali di sebelah kanan bawah dengan paku bernomor  $u$ .

## Format Keluaran

Sebuah baris berisi barisan  $N$  buah bilangan, dipisahkan oleh spasi, yang menyatakan barisan nomor paku, sedemikian sehingga barisan ini merupakan barisan terkecil ke- $K$  secara leksikografis.

### Contoh Masukan 1

```
0..34.6
4 1
2 1 0
2 4 1
4 3 0
```

### Contoh Keluaran 1

```
2 1 4 3
```

### Contoh Masukan 2

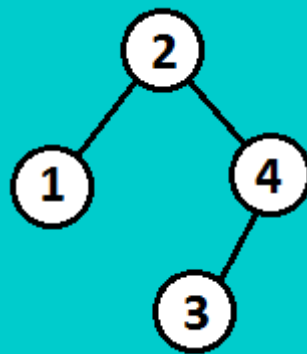
```
0..3..6
4 3
2 1 0
2 4 1
4 3 0
```

### Contoh Keluaran 2

```
2 4 3 1
```

### Penjelasan

Kedua contoh masukan mendeskripsikan desain hiasan dinding yang sama, yakni:



Terdapat 3 buah barisan nomor paku yang menghasilkan pohon tersebut, yakni:

1. 2 1 4 3
2. 2 4 1 3
3. 2 4 3 1

### Subsoal

Pada semua subsoal, berlaku:

- Setiap paku memiliki nomor unik antara 1 sampai dengan N.
- Masukan selalu merupakan desain hiasan dinding yang benar.

- K tidak lebih daripada banyaknya cara penyusunan hiasan dinding yang bersangkutan.
- Meskipun cara penyusunan hiasan dinding bisa banyak sekali, nilai K tidak akan lebih daripada 2.000.000.000.

**Subsoal 1 (7 poin)**

- $N = 8$
- $K = 1$
- Kasus uji dapat diunduh [di sini](#).

**Subsoal 2 (14 poin)**

- $N = 8$
- $K = 3$
- Kasus uji dapat diunduh [di sini](#).

**Subsoal 3 (17 poin)**

- $1 \leq N \leq 8$

**Subsoal 4 (11 poin)**

- $1 \leq N \leq 100$
- $K = 1$

**Subsoal 5 (20 poin)**

- $1 \leq N \leq 100$
- Semua paku dengan nomor lebih kecil daripada nomor paku teratas tidak memiliki paku yang terhubung dengan tali di sebelah kanan bawah.
- Semua paku dengan nomor lebih besar daripada nomor paku teratas tidak memiliki paku yang terhubung dengan tali di sebelah kiri bawah.

**Subsoal 6 (31 poin)**

- $1 \leq N \leq 100$

# Komunikasi Bebek

Batas Waktu	0,4 detik
Batas Memori	256 MB

## Deskripsi

Pak Dengklek memiliki sebuah kebun yang panjangnya tak hingga. Kebun Pak Dengklek dapat direpresentasikan dalam garis sumbu  $x$  pada koordinat Kartesian. Pak Dengklek memiliki  $N$  ekor bebek pada kebun tersebut. Bebek ke- $i$  terletak pada koordinat  $x_i$  pada sumbu  $x$ , dengan  $x_i$  adalah bilangan bulat. Dijamin tidak ada dua bebek pada posisi yang sama.

Sepasang bebek dapat berkomunikasi jika dan hanya jika tidak ada bebek lain yang berada di antara dua bebek tersebut. Karena  $N$  bebek semuanya berada pada satu garis, tentu saja ada  $N - 1$  pasang bebek yang dapat berkomunikasi. Jarak dari sebuah komunikasi adalah jarak dua bebek yang sedang melakukan komunikasi tersebut.

Sebagai contoh, jika pada awalnya ada 4 bebek yang berada pada posisi  $\{1, 2, 5, 6\}$ , maka jarak komunikasi bebek 1 dan 2 adalah 1, jarak komunikasi bebek 2 dan 3 adalah 3, dan jarak komunikasi bebek 3 dan 4 adalah 1.

Pada suatu saat mungkin saja terdapat jarak komunikasi yang berbeda-beda. Pak Dengklek merasa komunikasi bebek berjalan tidak lancar jika terdapat lebih dari satu nilai jarak komunikasi. Untuk itu, Pak Dengklek mungkin harus mengganti konfigurasi kebunnya. Bebek-bebek Pak Dengklek sudah terlalu nyaman dengan posisinya, sehingga mereka tidak mau dipindah posisinya. Sehingga, yang bisa dilakukan oleh Pak Dengklek adalah menambahkan bebek baru ke kebun atau mengambil bebek lama keluar dari kebun.

Pak Dengklek akan mengambil  $X$  bebek dan meletakkan  $Y$  bebek, sehingga pada akhirnya terdapat  $(N - X + Y)$  bebek, dan posisi bebek-bebek tersebut harus berbeda-beda dan bebek ke- $i$  terletak pada  $x_i'$  dimana  $x_i'$  adalah bilangan bulat, dan semua jarak komunikasi untuk  $(N - X + Y - 1)$  pasangan bebek bernilai sama. Tentukan nilai  $X + Y$  minimal yang dapat dilakukan oleh Pak Dengklek.

## Format Masukan

Pada awalnya, program Anda akan menerima label kasus uji. Label kasus uji berisi sebuah *string* yang dijelaskan sebagai berikut:

- Panjang *string* tersebut adalah banyaknya subsoal ditambah satu.
- Digit pertama dari label adalah karakter ke-0, digit kedua dari label adalah karakter ke-1, dan seterusnya.
- Karakter ke-0 akan berisi 0 jika kasus uji tersebut merupakan contoh kasus uji, atau berisi '.' jika bukan.
- Untuk setiap nilai  $i$  di antara 1 hingga banyaknya subsoal, berlaku:

- jika kasus uji tersebut memenuhi batasan subsoal ke-i, maka karakter ke-i berisi i, atau
- jika kasus uji tersebut tidak memenuhi batasan subsoal ke-i, maka karakter ke-i berisi karakter '.'

Sebagai contoh apabila label sebuah kasus uji sebuah soal adalah 0..345, maka:

- Soal tersebut memiliki 5 buah subsoal,
- Kasus uji tersebut merupakan contoh kasus uji, dan
- Kasus uji tersebut memenuhi batasan subsoal ke-3, ke-4, dan ke-5.

Baris kedua berisi sebuah bilangan bulat N yang menunjukkan banyaknya bebek mula-mula.

Baris ketiga berisi N bilangan bulat yang dipisahkan spasi. Bilangan ke-i menunjukkan nilai dari  $x_i$ .

### Format Keluaran

Sebuah bilangan bulat yang menunjukkan nilai  $X + Y$  minimal agar konfigurasi kebun Pak Dengklek memenuhi ketentuan di atas.

### Contoh Masukan

```
0..3456
5
2 4 6 10 18
```

### Contoh Keluaran

```
2
```

### Penjelasan

Pak Dengklek akan meletakkan bebek pada posisi  $x = 14$  dan mengambil bebek pada posisi  $x = 4$ , sehingga pada akhirnya akan ada bebek-bebek pada posisi  $\{2, 6, 10, 14, 18\}$ , dimana setiap komunikasi bebek berjarak 4. Solusi ini membutuhkan 1 pengambilan bebek dan 1 penambahan bebek. Tidak ada solusi yang lebih optimal dari solusi ini.

### Subsoal

Pada semua subsoal, berlaku:

- $-2.500 \leq x_i \leq 2.500$

### Subsoal 1 (6 poin)

- $N = 6$
- Kasus uji dapat diunduh [di sini](#).

### Subsoal 2 (14 poin)

- $N = 9$
- Kasus uji dapat diunduh [di sini](#).

**Subsoal 3 (19 poin)**

- $1 \leq N \leq 15$

**Subsoal 4 (21 poin)**

- $1 \leq N \leq 50$

**Subsoal 5 (13 poin)**

- $1 \leq N \leq 300$

**Subsoal 6 (27 poin)**

- $1 \leq N \leq 2.000$