خلاصه‌سازی

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت

با دریافت مجموعه‌ای از کلمات، برای کلماتی که طول آن‌ها **بیشتر از ده** می‌باشد، حروف اول و آخر را نگه داشته و به جای حروف میانی **تعداد** آن‌ها را قرار دهید.

ورودی

در خط اول n یا تعداد کلمات و در n خط بعد خود کلمات با حداکثر طول l نوشته می‌شوند.

$$1 \leq n, l \leq 1000$$

خروجی

در n خط از خروجی استاندارد کلمات ورودی را با ترتیب اولیه و به صورت خواسته شده چاپ کنید.

ورودی نمونه ۱

```
3
more
localization
industrial
```

خروجی نمونه ۱

```
more
l10n
industrial
```

ورودی نمونه ۲

```
2
read
abbbbbbbbbbbbbbbbbbbbbbbbbbbbba
```

خروجی نمونه ۲

```
read
a30a
```

زیررشته‌ی صدادار

برنامه‌ای بنویسید که رشته s را از ورودی گرفته و طول بزرگ ترین زیر رشته ای که فقط از حروف صدادار باشد را در خروجی نشان دهد.

حداکثر اندازه ورودی ۱۰۰ کاراکتر است.

مثال

ورودی نمونه

```
abaaucad
```

خروجی نمونه

```
3
```

سانسورچی

برنامه‌ای بنویسید شامل تابعی که دو رشته به عنوان ورودی بگیرد و رشته‌ی دوم را در رشته‌ی اول سانسور کند! سپس رشته اول را سانسور شده چاپ کند.

حداکثر طول هر یک از رشته‌های ورودی ۲۰۰ است.

مثال

ورودی نمونه

```
Mohammad Ali Reza Navid Ali Hossein Alireza
Ali
```

خروجی نمونه

```
Mohammad *** Reza Navid *** Hossein ***reza
```

عملیات رشته‌ای

برنامه‌ای بنویسید شامل تابعی که یک رشته به‌عنوان ورودی دریافت کرده و حاصل آن را به‌دست آورد:

$$f(\text{ "123-157+64-322" }) = -292$$

توجه: رشته‌ی ورودی شامل اعداد، + و - است.

ورودی

در یک خط از ورودی استاندارد، رشته‌ش شامل عملیات ریاضی وارد می‌شود.

حداکثر طول ورودی را ۵۰۰ کارکتر در نظر بگیرید.

خروجی

در یک خط از خروجی استاندارد، حاصل عملیات را چاپ کنید.

ورودی نمونه

```
-12-12-12+13
```

خروجی نمونه

```
-23
```

رشته‌ی خوب

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۶۴ مگابایت

آرش علاقه‌ی زیادی به حرف a دارد، زیرا نام او با حرف «آ» آغاز شده است. آرش رشته‌ای که بیش از نصف کاراکترهای آن a باشند را یک *رشته‌ی خوب* می‌نامد. برای مثال، aaacb و acaa رشته‌های خوبی هستند، اما baba و abbba رشته‌های خوبی نیستند.

آرش می‌تواند برای تبدیل یک رشته به یک رشته‌ی خوب، برخی از کاراکترهای آن رشته را حذف کند. برنامه‌ای بنویسید که با دریافت یک رشته، **بیشینه‌ی طول** رشته‌ی خوبی که آرش می‌تواند با حذف کاراکترهایی از رشته‌ی ورودی بسازد را محاسبه کند. ممکن است رشته‌ی ورودی خوب باشد؛ در این‌صورت، آرش کاراکتری را از آن حذف نمی‌کند.

تضمین می‌شود که رشته‌ی ورودی شامل حداقل ۱ کاراکتر a است.

ورودی

در یک خط از ورودی استاندارد، رشته‌ی S نوشته می‌شود. این رشته تنها شامل حروف کوچک انگلیسی است.

$$1 \leq |S| \leq 10^7$$

خروجی

در یک خط از خروجی استاندارد، **بیشینه‌ی طول** رشته‌ی خوبی که آرش می‌تواند با حذف کاراکترهایی از S بسازد را چاپ کنید.

ورودی نمونه ۱

xaxxxxa

خروجی نمونه ۱

3

ورودی نمونه ۲

aaabaa

خروجی نمونه ۲

6

وارووواژه

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت

به رشته‌ای که بتوان آن را با جابه‌جایی کاراکترهای رشته‌ای دیگر ساخت، *وارووواژه* (*anagram*) گفته می‌شود.

برنامه‌ای بنویسید که با دریافت دو رشته‌ی S و T، چک کند که آیا S و T وارووواژه‌ی یکدیگر هستند یا خیر.

ورودی

در خط اول ورودی استاندارد، رشته‌ی S و در خط دوم، رشته‌ی T نوشته می‌شود. این دو رشته تنها شامل حروف کوچک انگلیسی هستند.

$$1 \leq |S| = |T| \leq 10^4$$

خروجی

در یک خط از خروجی استاندارد، اگر S و T وارووواژه‌ی یکدیگر هستند، `YES` و در غیر این‌صورت، `NO` را چاپ کنید.

ورودی نمونه ۱

```
prep
pepr
```

خروجی نمونه ۱

```
YES
```

ورودی نمونه ۲

```
abcd
efgh
```

خروجی نمونه ۲

```
NO
```

ساختار ساده

در این تمرین شما باید فاصله دو نقطه از یکدیگر را در دستگاه دکارتی پیدا کنید اما چون سوال برای انتهای ترم بیش از حد ساده است انتظار می‌رود مقداری "ساختار"مندتر این کار را انجام دهید.

ابتدا طول و عرض نقطه اول و سپس طول و عرض نقطه دوم به برنامه داده می‌شود سپس فاصله بین دو نقطه در خروجی (تا پنج رقم اعشار) چاپ می شود.

دقت کنید که در صورت عدم استفاده از struct نمره‌ای به شما تعلق نمی‌گیرد.

استفاده از math و cmath مجاز است.

مثال

ورودی نمونه ۱

```
1
1
2
2
```

خروجی نمونه۱

```
1.41421
```

وروی نمونه ۲

```
2.5
5
5
2.5
```

خروجی نمونه ۲

```
3.53553
```

مرتب‌سازی لیست کارها

پیاده‌سازی این سوال بسیار در پروژه نهایی به شما کمک خواهد کرد.

فرض کنید یک آرایه از **کار**ها داریم.

هر **کار** یک ساختار با مشخصات زیر است:

۱. عنوان از جنس رشته
۲. توضیحات از جنس رشته
۳. اهمیت از جنس عدد ۱ تا ۱۰ ساختاری مناسب برای نگه‌داری این کار طراحی کنید.

در طول برنامه ما کاربر می‌تواند یک کار جدید ایجاد و مشخصاتش را در ورودی وارد کند. برای این منظور یک تابع بنویسیم. نکات:

۱. در صورتی که رشته را در ساختار، به شکل `char*` تعریف کرده‌اید، این تابع جای مناسبی برای تخصیص حافظه برای آن است.
۲. با توجه به مورد بعد، تابع را طوری طراحی کنید که فضای مورد نیاز برای ساختار ساخته‌شده را در heap تخصیص دهد و پوینتر به آن را برگرداند.
۳. در تابع main خود آرایه‌ای از پوینتر به struct های کار داشته باشید و نتیجه فراخوانی تابع new_user را در آن بریزید.

همچنین ما نیاز داریم که کارهای موجود در سیستم را ببینید. بنابراین یک تابع داشته باشید که وظیفه چاپ یک ساختار را بر عهده داشته‌باشد. در مورد اینکه ورودی این تابع خود ساختار باشد یا اشاره‌گر به آن، تصمیم بگیرید.

در آخر ما نیاز داریم که آرایه‌ای از کارهایمان را به صورت مرتب شده داشته باشیم. برای این منظور (همانطور که پیش‌تر گفته شد) یک ارایه از پوینتر به ساختارهای خود بسازید. می‌توانید فرض کنید نهایتا ۱۰۰۰ کار در سیستم وجود دارد و سایز آرایه را ثابت بگیرید. برای اضافه کردن کار جدید به آرایه شده، از ایده‌ی مرتب‌سازی درجی استفاده کنید.

مبنای مرتب‌سازی را اهمیت کارها در نظر بگیرید. کارهای مهم‌تر (عدد اولویت بالاتر) اول بیایند. (در صورتی که چند کار یک درجه اولویت داشتند ترتیب بین آن‌ها اهمیتی ندارد)

ورودی

در اولین خط ورودی n می‌آید که تعداد کارهاییست که قرار است وارد کنیم. در n*3 خط بعدی، n عدد کار می‌آیند.

خروجی

آرایه مرتب‌شده‌ی کارها را چاپ کنید. (ابتدا اولویت‌های بالاتر)

مثال

ورودی نمونه ۱

```
2
title1
desc1
1
title2
desc2
10
```

خروجی نمونه ۱

```
title: title2
descr: desc2
prior: 10
title: title1
descr: desc1
prior: 1
```

دو کار ما بعد از مرتب‌سازی به صورت برعکس چاپ شدند. .

ورودی نمونه ۲

```
3
title1
desc1
2
title2
desc2
10
last_title
another_desc
1
```

خروجی نمونه ۲

```
title: title2
descr: desc2
prior: 10
title: title1
descr: desc1
prior: 2
title: last_title
descr: another_desc
prior: 1
```

راهنمایی: هر المنت آرایه مد نظر ما، `struct task*` است. حال کافیست در هر iteration از حلقه، کار جدید را در اندیس i ام بریزیم و سپس وایل داخلی insertion sort را رویش صدا بزنیم تا همزمان با insert شدن، مرتب شده و در جای خود قرار بگیرد.

مثلا آرایه می‌تواند چنین تعریفی داشته باشد:

```
1    struct task* tasks[100];
2    // or
3    task* tasks[100];
```

ماشین حساب اعداد کسری

می‌خواهیم یک ماشین حساب ساده برای اعداد کسری پیاده‌سازی کنیم. این ماشین حساب قابلیت انجام ۴ عمل اصلی و سپس ساده‌کردن جواب تا جای ممکن را دارد. (برای ذخیره کردن عدد کسری حتما باید از ساختار استفاده شود.)

ورودی

در ۴ خط اول ورودی، صورت و مخرج دو عدد کسری به عنوان ورودی داده می‌شود. این اعداد از نوع integer هستند و تضمین می‌شود که مخرج صفر نیست.

در خط بعدی یکی از کارکترهای / * - + به عنوان عملگر به برنامه داده می‌شود.

خروجی

حاصل ساده شده‌ی عبارت مشابه نمونه

مثال

ورودی نمونه ۱

```
2
4
3
6
*
```

خروجی نمونه ۱

```
1/4
```

ورودی نمونه ۲

```
2
4
3
6
-
```

خروجی نمونه ۲

```
0/24
```

ورودی نمونه ۳

```
3
7
12
28
+
```

خروجی نمونه ۳

```
6/7
```