رئیس زندان دلتنگ

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

رئیس زندان، پرونده‌ی زندانیانی که هر سال وارد زندان می‌شوند را به ترتیب روی هم می‌گذارد. هر پرونده شماره‌ای دارد که رئیس زندان، هر زندانی را با آن شماره می‌شناسد. زندان n سال است که تاسیس شده و به ازای هر سال ستونی از پرونده در بایگانی است. حالا چند روزی است که رئیس زندان دلتنگ شده است و می‌خواهد بدون بهم ریختن پرونده‌ها خاطراتش را مرور کند.

رئیس زندان در ابتدا عدد q را به شما می‌دهد که نشان‌دهنده‌ی تعداد خاطراتی است که می‌خواهد مرور کند. سپس q زوج مرتب (i, j) به شما می‌دهد و می‌خواهد به او شماره پرونده jامین زندانی که در سال iام وارد زندان شده است را بدهید.

مدیریت حافظه‌ی پرونده‌ها در این برنامه باید پویا باشد؛ پس به **تخصیص** و **آزادسازی** حافظه دقت کنید.

ورودی

در خط اول ورودی، عدد طبیعی n آمده است که بیانگر تعداد سال‌هایی است که زندان تاسیس شده. در n خط بعدی، ابتدا عدد p می‌آید که بیانگر تعداد زندانیان آن سال است و سپس p عدد به فاصله از هم آمده است که به‌ترتیب شماره پرونده‌ی زندانیان آن سال است. سپس، عدد q آمده است که تعداد سؤالات رئیس زندان را نشان می‌دهد و در q خط بعدی دو عدد با فاصله از هم آمده است که به ترتیب سال و زندانی درخواستی را نشان می‌دهد.

خروجی

شماره‌ی پرونده‌ی زندانی‌های درخواستی را چاپ کنید.

توجه

- در صورتی که از تخصیص آرایه پویا یا **آرایه‌ی نامتوازن** استفاده نکنید، هیچ نمره‌ای دریافت نمی‌کنید.

مثال

ورودی نمونه

```
3
4 1 2 3 4
2 5 6
5 7 8 9 10 11
4
1 2
1 4
3 3
2 1
```

خروجی نمونه

```
2
4
9
5
```

بازدوباره تخصیص بده

یکی از توابع مهم در stdlib که معمولا در زبان سی بیش‌تر استفاده می‌شود، `realloc` است. این تابع مشخصات جالبی دارد و با ورودی‌های مختلف می‌تواند کارایی‌های مختلف داشته‌باشد برای همین شایان توجه است.

توضیح این تابع را در زیر بخوانید:

```
 The realloc() function changes the size of the memory block pointed  to by pt
The contents will be unchanged in the range from the start of the region up to
If the new size is larger than the old size, the added memory will not be init
If ptr is NULL, then  the  call  is  equivalent  to  malloc(size),
 for all values of size;
if size is equal to zero, and ptr is not NULL, then the call is equivalent to
It is guranteed that if ptr is not NULL, it is returned by an earlier call to
```

در این تمرین می‌خواهیم تابعی شبیه realloc با این امضا پیاده‌سازی کنیم:

```
1
2  int *our_realloc(int *ptr, long old_size, long new_size){
3
4  }
```

مشخصات این تابع عبارت است از:

- یک پوینتر به حافظه‌ی از پیش اختصاص‌داده‌شده، طول قبلی حافظه (به واحد int)، طول جدید حافظه (باز هم به واحد int) می‌گیرد.
- هدف اصلی آن این است که حافظه‌ی قدیمی را تبدیل به حافظه‌ای با طول جدید کند.
- در عملیات تبدیل، مقدارهای حافظه‌ی قدیمی به حافظه ی جدید منتقل می‌شوند.
- در صورتی که اندازه حافظه جدید کوچک‌تر از حافظه قبلی بود فقط به اندازه حافظه کوچک‌تر عملیات انتقال انجام شود.

- دو حالت خاص استفاده از این تابع وجود دارد. در صورتی که پوینتر NULL به آن ورودی داده شود، مثل malloc (یا new) کار می‌کند یعنی فقط اختصاص حافظه انجام می‌دهد.
- و در صورتی که سایز نهایی صفر باشد فقط عملیات free انجام می‌شود.
- در نهایت آدرس حافظه جدید برمی‌گردد. (در صورتی که بند بالا برقرار است مقدار برگشتی اهمیتی ندارد)

- در آن فقط از توابع malloc و free (یا فقط از new و delete!) استفاده شده.
- همانطور که متوجه شدید، در این تمرین (برخلاف realloc واقعی) به جای کار با «هر پوینتری» از int* استفاده می‌کنیم.
- می‌توانید اندازه int را ۴ بایت در نظر بگیرید.
- دقت کنید که عملیات free (یا delete) کردن بلوک حافظه قبلی را فراموش نکنید.

کد نمونه

```c
#include <stdlib.h>
#include <stdio.h>


int *our_realloc(int *ptr, long old_size, long new_size){
     // TODO
}


int main(){

    int *a = our_realloc(NULL, 0, 2); // allocate
    a[0] = 1;
    a[1] = 2;

    a = our_realloc(a, 2, 4); // bigger memory
    a[2] = 3;
    a[3] = 4;

    printf("%d %d %d %d\n",a[0], a[1], a[2], a[3]);

```

```
22
23      a = our_realloc(a, 4, 1); // smaller memory
24      printf("%d\n",a[0]);
25
26
27      our_realloc(a, 1, 0); // free
28
29      return 0;
30  }
```

خروجی کد نمونه

```
1 2 3 4
1
```

توجه:

- امضای متد را تغییر ندهید.
- برای اینکه از صحت برنامه خود مطمئن شود، برنامه نمونه را اجرا کنید.
- برای آپلود، کد نمونه را کامل کرده و آپلود کنید. (به قسمت main دست نزنید)
- این سوال به صورت دستی تصحیح می‌شود.

ادیت

آپلود cpp هم باز شد، زین پس می‌توانید از new و delete (یا delete[]) هم استفاده کنید.

ویندوز 9

همانطور که می‌دانید هستهٔ اکثر سیستم‌عامل‌ها با زبان‌های `C` و `C++` نوشته می‌شوند. مایکروسافت نیز برای نوشتن هسته ویندوز که به `windows NT` معروف است از همین زبان‌ها استفاده می‌کند.

همچنین می‌دانید که پوینترها و مدیریت حافظه نقش بسیار مهمی در سیستم‌عامل دارند. برای همین مهندسان مایکروسافت از بین کسانی انتخاب می‌شوند که مبحث کار با حافظه و پوینتر را به خوبی فهمیده باشند! در درس سیستم‌عامل مطالب بیشتری در این خصوص می‌آموزید.

متاسفانه مهندسان مایکروسافت در زمانی که مشغول تست ویندوز 9 بودند متوجه باگ‌های عجیبی شدند. آن‌ها به دلایل تجاری دست از تلاش برای دیباگ این کد کشیدند و پروژه ویندوز 10 را استارت زدند و سپس با موفقیت عرضه کردند. با این وجود سورس کد ویندوز 9 سوژه خوبی برای تمرین‌های مبانی است!

سورس‌کد مربوط به بخشی از ویندوز 9 است که در اختیار تیم حل تمرین دانشگاه شهید بهشتی برای مقاصد آموزشی قرار گرفته. فایل را از این لینک دانلود کنید و به آن نگاه کنید.

آن چه خواسته شده

چیزی که از شما خواسته می‌شود این است که برنامه پر از خطا را گرفته و آن را **پیرایش کنید.** به این معنی که **حق ندارید چیزی به آن اضافه کنید،** فقط می‌توانید یک خط یا مقداری از یک خط را کامنت کنید.

مثال‌های صحیح از کامنت‌کردن کد.

```
// int a;
int /*123*/a;
```

تضمین می‌شود که در هر یک از توابع حداکثر یک خط مشکل‌دار وجود دارد و با کامنت کردن صحیح می‌توانید مشکل را حل کنید.

مشکلاتی که باید حل کنید

این کد چندین مشکل در کار با پوینتر و حافظه‌ها دارد. وظیفه شما این است که برنامه را طوری تغییر دهید که:

- اجرای آن باعث خطای زمان اجرا نشود.
- حافظه بیش‌از حد نیاز نگیرد.
- حافظه‌ای که می‌گیرد را در جای صحیح آزادسازی کند.

نحوه داوری و نمره‌دهی

این سوال به صورت دستی تصحیح می‌شود و در کد شما موارد ذکرشده بررسی می‌شود اما برای درک تقریبی خودتان از وضعیت و صحت کد، داوری توسط کوئرا صورت می‌گیرد. این داوری *برخی از memory leakها* و *runtime error* ها را بررسی می‌کند.

ورودی

در زمان داوری ورودی مناسب (۵ کاراکتر در ۵ خط) به برنامه داده می‌شود

خروجی

خروجی برنامه باید دقیقا چنین چیزی باشد:

```
Searching for problems...
We didn't find any problems
enter any character to continue...
just one in twelve
enter any character to continue...
safest cpp function in the universe
enter any character to continue...
we want to debug last code!
enter any character to continue...
5
5
enter any character to continue...
```

صف

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

حتما تا کنون در صف‌های مختلفی از جمله صف روز ثبت‌نام ایستاده‌اید.....

عموما صف‌ها سه ویژگی زیر را دارند:

- نفر جدید به **انتهای** صف اضافه می‌شود.
- افراد از **ابتدای** صف خارج می‌شوند.
- طول صف محدودیتی **ندارد**.

برنامه‌ای بنویسید که داده‌هایی به صورت **حروف کوچک انگلیسی** را با دستورات زیر در قالب یک **صف** مدیریت کند:

۱. دستور **enqueue**: مقداری را به **انتهای** صف اضافه می‌کند.
۲. دستور **dequeue**: **اولین** عنصر صف را خارج و چاپ می‌کند. اگر صف خالی باشد، عبارت **empty** چاپ می‌شود.
۳. دستور **print**: عناصر صف را به ترتیب **از ابتدا تا انتها** چاپ می‌کند. در صورتی که صف خالی باشد عبارت **empty** چاپ می‌شود.
۴. دستور **size**: طول صف را چاپ می‌کند.

در برنامه‌ی خود **باید** توابع زیر را پیاده‌سازی کرده و از آن‌ها استفاده کنید:

```
1 void enqueue(char* &queue, int &n, char data);
2 char dequeue(char* &queue, int &n);
3 void print_queue(char* queue, int n);
```

به نکات زیر توجه کنید:

- استفاده از متغیرهای *global* مجاز **نیست**.

- در توابعی که نیاز است آدرسی که پوینتر به آن اشاره می‌کند عوض شود، از **call by reference to pointer** استفاده شده‌است. در صورت نیاز، می‌توانید در این مورد بیشتر بخوانید.

ورودی

در هر خط از ورودی استاندارد، دستوراتی مطابق جدول زیر وارد می‌شوند تا زمانی که مقدار **F** وارد شود.

خروجی

برای هر دستور به جز **E**، در هر خط از خروجی استاندارد و با توجه به آن دستور، عبارت مناسب را چاپ کنید.

مثال

ورودی نمونه ۱

```
E k
D
D
E x
D
D
E w
E z
P
E m
```

```
S
D
D
S
F
```

خروجی نمونه ۱

```
k
empty
x
empty
w z
3
w
z
1
```

بررسی دستورات:

۱. مقدار **k** در صف قرار می‌گیرد.

۲. اولین عضو صف یعنی **k** خارج‌شده و چاپ می‌شود.

۳. چون صف خالی است، عبارت **empty** چاپ می‌شود.

۴. مقدار **x** در صف قرار می‌گیرد.

۵. اولین عضو صف یعنی **x** خارج شده و چاپ می‌شود.

۶. چون صف خالی است، عبارت **empty** چاپ می‌شود.

۷. مقدار **w** در صف قرار می‌گیرد.

۸. مقدار **z** بعد از **w** در صف قرار می‌گیرد.

۹. عناصر صف از ابتدا چاپ می‌شوند. عناصر فعلی صف **w z** هستند.

۱۰. عنصر **m** بعد از **z** در صف قرار می‌گیرد.

۱۱. طول صف چاپ می‌شود. صف شامل **سه** عنصر **w z m** است.

۱۲. اولین عضو صف یعنی **w** خارج شده و چاپ می‌شود. بعد از آن عناصر صف **z m** خواهند بود.

۱۳. اولین عضو صف یعنی z خارج شده و چاپ می‌شود. بعد از آن تنها عنصر صف m خواهد بود.

۱۴. طول صف چاپ می‌شود.

۱۵. پایان ورود دستورات.

ورودی نمونه ۲

```
E t
D
D
D
P
E o
D
D
D
E f
S
D
D
E b
E c
E s
E y
P
E c
D
E o
F
```

خروجی نمونه ۲

```
t
empty
empty
empty
o
empty
```

```
empty
1
f
empty
b c s y
b
```