به نام خدا

درس مبانی برنامه‌سازی

تمرین ۷

دانشکده مهندسی کامپیوتر

دانشگاه صنعتی شریف

نیم سال اول ۰۰ ـ ۹۹

استاد:

رضا فکوری

مهلت ارسال:

۲۴ بهمن ـ ساعت ۲۳:۵۹:۵۹

مسئول تمرین‌ها:

امیرمهدی نامجو، پرهام صارمی

مسئول تمرین ۷:

صابر ظفرپور

طراحان تمرین ۷:

علیرضا هنرور، امیرصدرا عبدالهی، اشکان خادمیان، میلاد سعادت، محمدامین آریان، نازنین آذریان، پویا اسمعیلی

فهرست

سوالات

سوال ۱. تمرین ساده

چون این تمرین قراره خیلی آسون باشه، با یه سوال آسون هم شروعش کنیم. قراره چند تا رشته از ورودی بخونید و بعد از این که الفبایی مرتبشون کردید به طور مرتب چاپشون کنید.
اولویت مرتب کردن به ترتیب:

- علامت‌ها که شامل (_ – !@()%)می‌شه و اولویت خودشونم به همون ترتیب از راست به چپ است.
- اعداد
- حروف بزرگ
- حروف کوچک
- طول کوتاه تر (برای مثال بین دو رشته‌ی abc و ab ، اولویت ab بیشتر است)

برای مرتب کردن فرض کنید اسپیس وجود نداره و بدون توجه به اسپیس مرتب کنید

ورودی

در خط اول n که تعداد رشته‌ها است به شما داده می شود. در n خط بعدی، در هر خط یک رشته داده می شود که باید طبق توضیحات این رشته‌ها را مرتب کنید.

خروجی

پس از مرتب کردن رشته‌ها، در خط i ام خروجی، رشته‌ای که پس از مرتب کردن اولویت آن i است را چاپ کنید.

مثال

ورودی نمونه ۱

```
6
Bitcoin
bitc oin
_bitcoin
!Bitcoin
   -bitcoin
  bitcoi
```

خروجی نمونه ۱

```
_bitcoin
   -bitcoin
!Bitcoin
Bitcoin
  bitcoi
bitc oin
```

سوال ۲. امین

برنامه ای بنویسید که ابتدا یک رشته بدون space از کاربر دریافت کند، سپس در هر خط یک دستور دریافت کرده و روی رشته، تغییراتی ایجاد کند و بعد از هر دستور، در صورت نیاز، رشته تغییر یافته را چاپ کند و همچنین تا زمانی که دستور exit وارد نشده، به گرفتن دستورات ادامه دهد.

ورودی

در خط اول، رشته مورد نظر وارد میشود. این رشته از لحاظ نوع کاراکتر محدودیتی ندارد و حداکثر شامل ۱۰۰۰ کاراکتر است.

دستورات مربوط به تغییر رشته به صورت زیر است:

```
copy [n]
```

این دستور، رشته را n بار به خود اضافه میکند. اگر هیچ عددی به عنوان ورودی داده نشد، یکبار این کار را انجام می دهد. برای مثال اگر رشته مورد نظر، Simple Text باشد و دستور copy وارد شود، رشته به Simple TextSimple Text تبدیل می شود.

```
append [new string]
```

این دستور، رشته جدید را به ادامه رشته قبلی اضافه میکند.

```
find [substring]
```

این دستور، به دنبال زیررشته وارد شده در رشته اصلی میگردد؛ اگر وجود داشت، تعداد دفعات ظاهر شدن زیررشته در رشته اصلی را به آخر رشته اضافه میکند؛ در غیراین صورت، خود زیررشته را به آخر رشته اضافه میکند.

```
count (alphabets/digits)
```

تعداد حروف/اعداد در رشته را میشمارد و حاصل را به آخر رشته اضافه میکند.

```
delete [substring]
```

تمام زیررشته های موجود در رشته اصلی را پاک میکند. به عنوان مثال اگر رشته اصلی

TextSimpleSimpleText باشد و دستور delete Simple وارد شود، رشته به -Text Text تبدیل میشود.

```
reverse
```

این دستور، حروف رشته را برعکس میکند. به طور مثال رشته Simple را به elpmiS تبدیل میکند.

```
reverse [m],[n]
```

ین دستور، زیررشته در بازه $[m, n)$ در رشته اصلی را معکوس میکند. به عنوان مثال دستور ۱،۴ reverse ، رشته Simple را به Spmile تبدیل میکند.

```
PRINT_MODE (ON/OFF)
```

این دستور، وضعیت چاپ شدن رشته بعد از هر دستور را تغییر میدهد. به صورت پیشفرض، این وضعیت روی حالت ON قرار دارد.

```
print
```

این دستور، رشته را چاپ میکند.

```
exit
```

این دستور، برنامه را به پایان میرساند. توجه کنید که بعد از دستور exit ، رشته چاپ نمیشود.

خروجی

درصورتی که PRINT_MODE در حالت فعال باشد، بعد از هر دستور صحیح، رشته چاپ میشود و همچنین هربار که دستور print وارد شود، رشته چاپ میشود.

مثال

ورودی نمونه ۱

```
YekJomlehMasalanBaMaeni
append Simple
reverse 2,9
delete Masalan
```

```
count digits
find mp
PRINT_MODE OFF
delete helmoJkBa
print
copy
print
exit
```

خروجی نمونه ۱

```
YekJomlehMasalanBaMaeniSimple
YehelmoJkMasalanBaMaeniSimple
YehelmoJkBaMaeniSimple
YehelmoJkBaMaeniSimple0
YehelmoJkBaMaeniSimple01
YeMaeniSimple01
YeMaeniSimple01YeMaeniSimple01
```

سوال ۳. هوع تو میک مانی!!!

بنجامن یک دانشجوی ترم سه ای است که زیر بار درس در حال له شدن است و از طرف دیگر از بی پولی در عذاب.به دلیل فشار درس و دانشگاه زمان لازم برای کار کردن و کسب درامد را ندارد و تدریس به بچه دبیرستانی ها دیگر کفاف خرجش را نمیدهد از این رو تصمیم گرفته است تا از مهارت برنامه نویسی اش استفاده کرده و یک دایره مالی طراحی کند که بتواند مشترکان زیادی را به خود جذب کند اما از طرفی خودش بیشترین سود را ببرد.

او در حال برنامه ریزی برای دایره مالی اش بود که فهمید باید خودش را برای امتحان شفاهی یکی از دروسش آماده کند ازین رو قوانینی که برای دایره مالی تهیه کرده بود را در اختیار شما قرار داد تا در این مهم او را یاری کرده و دست جوان ایرانی را بگیرید

دایره دارای دستورات و قوانین زیر می باشد :

- دایره برای شروع کار نیاز به یک بنیان گذار دارد که باید مبلغ ۵۰۰۰ دلار برای ایجاد میز به بنجامن پرداخت کند. اگر بودجه او بیشتر از این مبلغ باشد به عنوان ذخیره در حسابش قرار می گیرد اما اگر کمتر باشد نمیتواند میز ایجاد کند.
- به دلیل نظام سرمایه داری دایره مالی شامل طبقات مختلف است. هر طبقه یک میز گرد است که در آن هر فرد جدید سمت راست آخرین فرد اضافه شده به میز می نشیند.
- بنیان گذار در طبقه ۰ قرار دارد و بعد از آن طبقات از ۱ شماره گذاری شده اند. هر طبقه ظرفیت محدودی از افراد را در خود جای میدهد که طبق تابع درجه دو x^2 محاسبه می شود وافراد با توجه به طبقه ای که در آن قرار دارند سود متفاوتی در یافت میکنند.
- بعد از ایجاد میز توسط بنیان گذار سایر افراد از دو طریق میتوانند وارد دایره شوند:

۱. از طریق معرفی شدن توسط کسانی که از قبل در میز بوده؛ در این شرایط آنها در اولین طبقه دارای ظرفیت پایین تر از معرف خود قرار می گیرند. مبلغی که با خود به همراه می آورند به این صورت تقسیم میشود:

- ۲۰ درصد حساب خودشان
- ۱۰ درصد حساب بنیان گذار میز
- ۵ درصد معرف

- ۱۵ درصد به عنوان کارمزد به بنجامن پرداخت میشود.
- و درنهایت ۵۰ درصد باقیمانده به صورت مساوی بین طبقات بالایی تقسیم میشود. (یعنی اگر ۵ طبقه داشته باشیم ۲۰ درصد از آن به هر طبقه میرسد) و پولی که به هر طبقه رسیده بین اعضای آن به طور مساوی تقسیم می شود.

۲. به صورت مستقل که در این صورت به آخرین طبقه یا در صورت نبود ظرفیت یک طبقه جدید در انتهای طبقات اضافه می شوند، . مبلغی که با خود به همراه می آورند به این صورت تقسیم میشود:

- ۱۵ درصد حساب خودشان
- ۱۰ درصد حساب بنیان گذار میز
- ۲۵ درصد از مقدار باقی مانده به صورت کارمزد به بنجامن پرداخت میشود.
- و درنهایت مقداری که باقی مانده به صورت مساوی بین طبقات بالایی تقسیم میشود (یعنی اگر ۵ طبقه داشته باشیم ۲۰ درصد از آن به هر طبقه میرسد) و پولی که به هر طبقه رسیده بین اعضای آن به طور مساوی تقسیم می شود

- اگر فردی ۵ نفر را معرفی کرده و وارد دایره کند به یک طبقه بالا منتقل می شود. اگر طبقه ی بالا پر باشد، جای این فرد با فردی از طبقه ی بالا که کمترین تعداد معرفی و سپس میزان پول را دارد، عوض میشود. پس از هر تغییر طبقه، تعداد افراد معرفی شده فرد ۰ می شود.
- اگر فردی بخواهد از دایره خارج شود ۵ درصد از مبلغ موجود در حسابش به عنوان کارمزد به بنجامن پرداخت شده و با مبلغ باقی مانده خارج می شود.
- هر فرد نام کاربری ای دارد که با آن شناخته می شود.
- در هر لحظه برنامه باید بتواند هر یک از اطلاعات زیر را در اختیارمان قرار دهد:

۱. تعداد طبقات
۲. تعداد کل اعضا
۳. تعداد اعضای هر طبقه
۴. معرف هر فرد
۵. نفرات سمت چپ و راست هر فرد

۶. موجودی حساب هر فرد
۷. افرادی که با یک فرد سر یک میز هستند
۸. سود بنجامن تا آن لحظه

نکته: بنجامن بنیان گذار میز نیست بلکه سود او از کارمزد هایی ک دریافت میکند تامین میشود.

ورودی و خروجی

ایجاد یک دایره

```
Create_a_table_for <username> with_deposit_of <money>
```

کار با این دستور شروع می‌شود.
اگر از قبل بنیان گذار داشته باشیم:

```
We already have a founder
```

اگر بودجه فرد کمتر از ۵۰۰۰ دلار باشد:

```
Money is not enough
```

اگر دایره با موفقیت ایجاد شود:

```
You now own a table
```

معرفی یک عضو جدید

```
Invitation_request_from <username> for <username> with_deposit_of
<money>
```

اگر نام کاربری فرد قبلا استفاده شده باشد:

```
Username already taken
```

اگر فرد با موفقیت اضافه شود:

```
User added successfully in level <level>
```

اضافه شدن به صورت مستقل

```
Join_request_for <username> with_deposit_of <money>
```

اگر نام کاربری فرد قبلا استفاده شده باشد:

```
Username already taken
```

اگر فرد با موفقیت اضافه شود:

```
User added successfully in level <level>
```

خروج

```
Leave_request_for  <username>
```

در اینجا بنیان‌گذار خارج نمی‌شود. پس از خروج کاربر دیگر نمی‌توان از یوزرنیم او در برنامه استفاده کرد یعنی وقتی کاربری از برنامه خارج شود فرد دیگری با آن یوزرنیم وارد دایره نخواهد شد.

اگر این نام کاربری موجود نباشد:

```
No_such_user_found
```

اگر فرد با موفقیت حذف شود:

```
User deleted successfully from level <level>
```

تعداد طبقات

```
Number_of_levels
```

خروجی تعداد طبقات خواهد بود.

تعداد کل اعضا

```
Number_of_users
```

خروجی تعداد اعضا خواهد بود.

تعداد اعضای یک طبقه

```
Number_of_users****_****in_level <level>
```

اگر همچین طبقه ای موجود نباشد:

```
No_such_level_found
```

در غیراین صورت خروجی تعداد اعضا در طبقه داده شده خواهد بود.

معرف یک فرد

```
Introducer_of <username>
```

اگر این نام کاربری موجود نباشد:

```
No_such_user_found
```

در غیر این صورت خروجی نام معرف فرد خواهد بود.

دوستان یک فرد (الکی مثلا کسی که سمت چپ و راست نشسته دوست اون فرده)

```
Friends_of <username>
```

اگر این نام کاربری موجود نباشد:

```
No_such_user_found
```

اگر فردی در میز نبود:

```
No_friend
```

در غیر این صورت خروجی در ۲ خط به ترتیب یوزرنیم دوست سمت چپ و راست فرد خواهد بود. اگر این کاربر در میزی نبود خروجی‌ای چاپ نکنید.

موجودی حساب یک فرد

```
Credit_of <username>
```

اگر این نام کاربری موجود نباشد:

```
No_such_user_found
```

در غیر این صورت خروجی موجودی حساب فرد خواهد بود.

افرادی که با یک فرد سر یک میز هستند

```
Users_on_the_same_level_with <username>
```

اگر این نام کاربری موجود نباشد:

```
No_such_user_found
```

اگر فردی در میز نبود:

```
No_level
```

در غیر این صورت خروجی لیستی از نام اعضا که با فرد سر یک میز هستند به جز خود فرد، به ترتیبی که به میز اضافه شده‌اند، خواهد بود که با اسپیس جدا شده اند.

سود بنجامن تا آن لحظه

```
How_much_have_we_made_yet
```

خروجی این دستور یک عدد خواهد بود که برابر با مجموع همه وجوهی است که تا آن لحظه به عنوان کامزد پرداخت شده اند.

پایان برنامه

```
End
```

سوال ۴. *Con Respecto*

سپهر که به تازگی خیلی در Blockchain و Smart Contract خفن شده است و به استخدام فردی به نام هانی در همین زمینه درآمده و هفته ای دست کم دو پروژه بلاک چین را در برنامه هفتگی خود دارد. هانی که فردی جاه طلب است می‌خواهد یک miner بخرد و کد آن را دست‌کاری کند تا بتواند کمی هم از شبکه بلاک‌چین پولشویی کند. از آنجا که خودش عرضه این کار ها را ندارد آن‌را به سپهر می‌سپارد.

بیش ازین حرف نزنیم و بگیم، تو باید چه چیز هایی بدونی، و چه کاری از تو انتظار می‌ره:

از بدو خلقت تا یگانه لحظه اکنون که این متنو می‌خونی m بلاک در شبکه ساخته شده که به صورت زنجیر وار به هم متصل شده‌اند (blockchain) . درون هر بلاک n تراکنش (transaction) وجود دارد (ممکن است تعداد تراکنش های یک بلاک با بلاکی دیگر فرق کند!) که هر تراکنش متناظر با یک رشته است. هر یک از بلاک ها نیز رشته ای معرفی کننده و بسیار مهم به نام hash دارد که این رشته تابعی از تمام تراکنش های موجود در بلاک و همچنین hash بلاک قبلی است(البته ما در این سوال فرضیاتی برای ساده سازی مسیله انجام دادیم که در واقعیت به این شکل نیست:). یعنی برای تولید hash بلاک i ام به تمام تراکنش های این بلاک و همچنین hash بلاک i-۱ ام نیاز داریم.(قطعا باید تا الان نگران تولید hash مربوط به بلاک شماره ۱ خلقت شده باشی، نگران نباش اون hash رو ما به شما می‌دیم.)

اما چه ترکیب خاصی از تراکنش ها و hash قبلی، hash جدید را می‌سازد؟

ابتدا تابع $value(s : string, P : int)$ را به صورت زیر تعریف می‌کنیم:

$$value(s, P) = ((\sum_{k=0}^{len(s)-1} s[k].P^k) mod 94) + 33$$

ورودی های تابع *value*:

- اول s که یک رشته است و s[k] که کد ASCII نظیر به کاراکتر kام از آن رشته است که در محاسبات استفاده می‌شود.
- دوم P یک عدد اول منسوب به بلاک است؛ به این معنا که برای هر بلاک می‌تواند با

بلاک دیگر فرق کند.

خروجی تابع *value*:

- خروجی عملیات ریاضی فوق منطقا یک عدد است که تابع value آنرا به کارکتر نظیر به کد اسکی‌ای برابر همان عدد تبدیل می‌کند و return می‌کند. تضمین می‌شود این عدد قطعا کارکتر منطقی‌ای می‌سازد و در بازه معقولی از کد های ASCII قرار می‌گیرد(به این فکر کن که چجوری تضمین می‌شه!)

تابع value دو جا استفاده می‌شود:

اول: برای بدست آوردن Primary String مربوط به بلاک. اینکه Primary String کجا استفاده می‌شه رو یکم دیگه بهت می‌گیم ولی اینکه چجوری این رشته رو برای یه block بسازی:

```
block.primary_string[i] = value(block.transactions[i], block.P)
```

همانطور که معلومه تابع value، کاراکتر iم از رشته Primary String یک بلاک را با عمل کردن روی تراکنش iم همان بلاک و با استفاده از عدد اول (P) همان بلاک را می‌سازد.

دوم: حال رسیدیم به **اصل کار** (ویلسون)، بدست آوردن hash بلاک. hash همون Primary String مربوط به بلاک است. فقط یه کارکترشو باید عوض کنی (جایی که hash بلاک قبلی استفاده می‌شه). سوالایی که باید واست ایجاد شده باشه اینه: کدوم کارکترو عوض کنم؟ جاش چی بذارم؟

ابتدا دومیو جواب می‌دیم. جاش alter رو بذار:

$$alter = value(previous\_hash, block.P)$$

که مسلما previous_hash همون هش بلاک قبلی‌است.

حالا سوال اولت، کارکتر alter رو بذار جا اندیس زیر:

$$alter \bmod block.n$$

که block.n تعداد تراکنش های همین بلاکیه ک داری زور می‌زنی hash اون رو حساب کنی.

اجبار سپهر

سپهر که گنگش بالاتر ازیناست که بخواهد کد کثیف بزند ما را مجبور کرد بهت شی گرایی یاد بدیم. ولی ما که می‌دونیم اگ حرفی از شی گرایی وسط بکشیم باید قید تی‌ای بودنو بزنیم صداشو در نمیاریم ولی بجاش ازت می‌خواهیم عناصر زیر رو در ابتدای کدت قرار بدی:

```
    struct block {
    /* block variables */
};

typedef struct block Block;
Block* new_block(int, int);
```

پیاده سازی محتوای ساختاری block که متناسب سوال باشد بر عهده شماست ولی تابع new_block چیست؟ پروتوتایپ این تابع به شما داده شده است و پیاده سازی آن تابع هم به عهده خود شماست. شما باید به گونه ای آنرا پیاده سازی کنید که هر گاه نیاز به یک متغیر از نوع block struct داشتید آن تابع را صدا بزنید و برایتان یک متغیر ازین نوع بسازد و پوینتری از آنرا برگرداند. **بسیاار مهم است که بدانید** هر جای کد، غیر از درون تابع new_block سعی کنید به هر شکلی یک متغیر از نوع struct block بسازید، **سپهر در نهایت که کد ها را چک می‌کند ۵۰ درصد نمره دریافتی‌تان ازین سوال رو برای شما در نظر نمی‌گیرد حتی اگر کدتان از کوئرا نمره کامل گرفته باشد** چرا که انتظار می‌رود همانطور که گفته شد برای ساختن یک بلاک جدید در هر جای دیگر کد فقط از خروجی تابع new_block استفاده کنید. ترم بعد می‌بینید چیزی که زدید شدیدا شبیه مفهوم Constructor ها در شی گرایی‌ست!

ورودی

سپهر سرش خیلی شلوغه چند تا پروژه دیگه‌ست. همینطور که می‌بینی به ما گفته داک این پروژه کوچیکو براتون بنویسیم و حتی خودش وقت نکرده بیاد بهت بلاک‌چینو توضیح بده. ما در ابتدا در یک خط به شما m را می‌دهیم که تعداد بلاک هاییه ک مونده رو دستمون. تضمین می‌کنیم m عددی مثبت است.

در خط بعد رو یک خط کامل بدون space بهت hash بلاک اول این دسته بلاک جا مونده رو می‌دیم. پس از دادن hash بلاک اول m - ۱ بار اطلاعات بلاک های ۲ تا m را

به شما می‌دهیم. اطلاعات هر بلاک به صورت زیر است:

در یک خط به ترتیب ابتدا n و سپس P بلاک داده می‌شود. یه تضمین دیگه هم می‌کنیم که P عدد اول است. در نهایت در n خط بعدی n رشته بدون space به عنوان رشته های تراکنش های ۱ تا n مربوط به این بلاک به شما داده می‌شود.

خروجی

لطف کن و hash مربوط به بلاک آخر (m م) رو تو یه خط چاپ کن ما هم بریم به بقیه کارامون برسیم.

مثال

ورودی نمونه ۱

```
2
abc
2 7
defghijklm
nopqrstuv
```

(از خط اول می‌فهمیم) ۲ بلاک داریم که (از خط دومش می‌فهمیم) hash بلاک اول که قول دادیم بهتون بدیم abc است. بلاک دوم مشخصاتش به شکلیه ک اومده. ۲ تا تراکنش داره و عدد اولش هم P=۷ عه. تو دو خط بعد اون هم رشته مربوط به تراکنش های اول و دومِ این بلاک اومده.

خروجی نمونه ۱

```
ny
```

بلاک آخر، بلاک دومه که اگ Primary String شو بر اساس عدد اولش و تراکنش هاش حساب کنی رشته بدست اومده nw خواهد بود. نتیجه حاصل از پاس دادن hash قبلی (abc) به تابع value هم حرف y با کد اسکی ۱۲۱ خواهد بود. $121\%2 = 1$ پس، از رشته پرایمری بدست آمده برای بلاک آخر، اندیس ۱ را با y جا گذاری می‌کنیم که میشه همین hash ی که بالا نوشته شده است.