تمرین ۹ درس طراحی الگوریتم

بنفشه کریمیان
سید صالح اعتمادی
دانشگاه علم و صنعت ۹۸-۹۷

لطفا به نکات زیر توجه کنید:

- مهلت ارسال این تمرین ۲۸ اردیبهشت ماه ساعت ۱۱:۵۹ ب.ظ است.
- این تمرین شامل سوال های برنامه نویسی می باشد، بنابراین توجه کنید که حتماً موارد خواسته شده را رعایت کنید. .
- نام شاخه، پوشه و پول ریکوست همگی دقیقا $A9$ باشد.
- اگر در حل تمرین شماره ی ۹ مشکلی داشتید،لطفا به @BTS_BangTanARMY مراجعه کنید..

موفق باشید.

توضیحات کلی تمرین

تمرین این هفته ی شما،۳ سوال دارد که باید به همه ی این سوال ها پاسخ دهید. برای حل این سری از تمرین ها مراحل زیر را انجام دهید:

۱. ابتدا مانند تمرین های قبل، یک پروژه به نام A۹ بسازید.

۲. کلاس هر سوال را به پروژه ی خود اضافه کنید و در قسمت مربوطه کد خود را بنویسید. هر کلاس شامل دو متد اصلی است:

متد اول: تابع solve است که شما باید الگوریتم خود را برای حل سوال در این متد پیاده سازی کنید.

متد دوم: تابع process است که مانند تمرین های قبلی در TestCommon پیاده سازی شده است. بنابراین با خیال راحت سوال را حل کنید و نگران تابع process نباشید! زیرا تمامی پیاده سازی ها برای شما انجام شده است و نیازی نیست که شما کدی برای آن بزنید.

۳. اگر برای حل سوالی نیاز به تابع های کمکی دارید؛ می توانید در کلاس مربوط به همان سوال تابع تان را اضافه کنید.

اکنون که پیاده سازی شما به پایان رسیده است، نوبت به تست برنامه می رسد. مراحل زیر را انجام دهید.

۱. یک UnitTest برای پروژه ی خود بسازید.

۲. فولدر TestData که در ضمیمه همین فایل قرار دارد را به پروژه ی تست خود اضافه کنید.

۳. فایل GradedTests.cs را به پروژه ی تستی که ساخته اید اضافه کنید. برای این تمرین مانند $A7$ و $A8$ برای هر سوال تست جداگانه با زمان جداگانه در نظر گرفته شده. بعد از حل کردن هر تمرین Assert.Inconclusive را از ابتدای تست حذف کرده و آن را اجرا کنید. چنانچه سوالی را حل نکردید از Assert.Inconclusive استفاده کنید.

دقت کنید که TestCommon تغییر یافته است. بنابراین شما باید نسخه ی جدید آن را با دستورPull

git دریافت کنید .

۱ میزان انرژی مواد سازنده

شما در این مسئله برای بدست آوردن میزان انرژی موادغذایی تشکیل دهنده ی منوی یک رستوران (با استفاده از لیست مواد تشکیل دهنده و میزان کالری هرکدام) الگوریتم `Gaussian Elimination` را پیاده‌سازی می‌کنید. شما منوی یک رستوران را در اختیار دارید که در آن برای هر غذا لیست مواد تشکیل دهنده و تخمینی از میزان کالری آن غذا مشخص شده اند. شما باید میزان کالری مواد تشکیل دهنده را برای تخمین میزان کالری غذای مورد علاقه‌ی خود بدست آورید.

فرمت ورودی: در خط اول تعداد غذاهای هر منو (تعداد مواد تشکیل دهنده برابر تعداد غذاهای در منو است) و در خطوط بعدی E و $a_1, \ldots, a_n$ را در یافت میکنید که برای هر غذا (هر خط) a_i میزان ماده‌ی i ام و E تخمینی از کالری کلی این غذا است. اگر ماده ای در غذایی استفاده نشده باشد مقدار آن 0 دریافت میشود ولی توجه داشته باشید که کد شما باید برای مقادیر منفی هم کار کند. فرمت خروجی: برای هر کدام از مواد تشکیل دهنده کالری تخمین زده ی خود را رند کنید. به این صورت که اگر اعشار شما از ۰/۲۵ کمتر بود اعشار را برداشته و یا از ۰/۷۵ بزرگتر مساوی بود به بالا رند کنید (اعشار را برداشته و در صورت ثبت بودن جواب یک عدد به جواب اضافه و در غیر اینصورت یک عدد از آن کم کنید) در غیر این صورت اعشار را ۰/۵ قرار دهید.

مثال ۱)

```
4
1 0 0 0 1
0 1 0 0 5
0 0 1 0 4
0 0 0 1 3
```

```
1 5 4 3
```

مثال ۲)

```
2
1 1 3
2 3 7
```

```
2 1
```

مثال ۳)

```
5
5 -5 -1
-1 -2 -1
```

```
0 0.5
```

دقت کنید که جواب دقیق برابر با ۰/۴ و ۰/۲ بوده که به ۰ و ۰/۵ رند شده.

۲ مسئله ی رژیم بهینه

در این مسئله یک الگوریتم برای حل `Linear Programming` با تعداد نامساوری های کم برای حل مسئله ی رژیم بهینه پیاده سازی میکنید. شما میخواهید رژیم خود را بهینه کنید به این معنا که علاوه بر رعایت تمام ضوابط پیشنهاد داده شده توسط متخصص تغذیه، میخواهید بیشترین لذت را از غذای خود ببرید. شما ضوابط برای هر غذا و تخمینی از میزان علاقه خود به غذای مورد نظر را دارید. مثالی ضوابط میتواند "جمع میزان مصرف روزانه کره و پنیر باید از ۲ واحد کمتر باشد" باشد. هدف این مسئله افزایش لذت وعده غذایی خود علاوه بر رعایت ضوابط است. میزان لذت شما از هر وعده غذایی برابر جمع میزان علاقع شما از خورد هر غذا و میزان پیشنهادی الگوریتم شما برای آن غذا است. ضوابط داده شده را میتوان به فرم یک سری نامساوی های خطی نوشت. برای مثال ضابطه ی "جمع میزان مصرف روزانه کره و پنیر باید از ۲ واحد کمتر باشد" را میتوان به صورت $amount_1 + amount_2 < 2$ نوشت. برای این مسئله در نظر داشته باشید که $amount_i >= 0$ است. برای رسیدن به هدف مسئله باید بیشترین میزان ممکنه برای جمع $amount_i \times pleasure_i$ بیابید. برای این کار باید توجه کنید که در ورودی حداکثر ۸ نامساوی داده می‌شود و جواب بهینه همواره در یکی از یال‌های چند ضلعی حاصل از نامساوی است. در نظر داشته باشید که تعداد کل نامساوی ها برابر مجموع تعداد نامساوی ها(n) و تعداد متغیرها(m) است (هر متغیر باید بزرگتر از ۰ باشد که خود یک نامساوی تلقی می‌شود) و m نامساوی از$n + m$ نامساوی تبدیل به معادله تساوی میشود. برای حل باید m نامساوی را به عنوان معادله تساوی حل کنید و جواب را چک کنید که برای بقیه نامساوی ها جواب دهد و در بین این جواب ها مقدار بهینه را انتخاب کنید.

فرمت ورودی: در خط اول تعداد ضوابط و تعداد غذاها (که با n و m به ترتیب نشان داده میشوند) و در n خط بعدی ماتریس A (که با ابعاد $m \times n$ است) را دریافت میکنید به صورتی که خط iام $a_{i1}, \ldots, a_{im}$ را شامل میشود. پس از دریافت ماتریس A در خط بعدی وکتور B به طول n را دریافت میکنید به طوری که $Ax < B$ است (دقت کنید که x میزان مصرفی از هر غذا و به طول m است). برای مثال اگر فرض کنیم ۲ نوع غذای f_1 و f_2 داریم ($m == 2$) اگر در خط iام از ماتریکس ۱ و ۳ را به عنوان ورودی بگیریم و مقدارiام وکتور B برابر ۵ باشد یعنی $1 \times amount_{f1} + 3 \times amount_{f2} < 5$ باید باشد. در خط آخر از ورودی وکتور V که نمایانگر میزان علاقه به هر غذا است را به عنوان ورودی میگیرید. برای مثال اگر خط آخر ورودی به ترتیب ۲ و -1 باشد یعنی برای رسیدن به هدف مسئله باید

$$amount_{f1} \times 2 + -1 \times amount_{f2}$$

را بیشینه کنید. در نظر داشته باشید که میزان علاقه به یک غذا میتواند منفی نیز باشد. فرمت خروجی: اگر هیچ رژیمی با این شرایط یافت نمیشد `No solution` و اگر تعداد زیادی رژیم با این محدودیت ها بود Infinity را به خروجی دهید. در غیر اینصورت `Bounded solution` را در خط اول و وکتوری از میزان

پیشنهادی برای مصرف هر غذا در این رژیم پس از رند شدن به روش توضیح داده را در خط دوم بنویسید.

مثال ۱)

```
3 2
-1 1
1 0
0 1
-1 2 2
-1 2
```

```
Bounded Solution
0 2
```

مثال ۲)

```
2 2
1 1
-1 -1
1 -2
1 1
```

```
No Solution
```

مثال ۳)

```
1 3
0 0 1
3
1 1 1
```

```
Infinity
```

۳ جاگذاری تبلیغات آنلاین

یکی از بیزینس های سوداور در دنیا تبلیغات آنلاین است. گوگل و فیسبوک سالانه بیلیونها دلار سود از این راه بدست میآورند و غالب بر ۹۰ درصد سود خود را از تبلیغات بدست میآورند. در این مسئله شما به یک سیستم آنلاین تبلیغات مثل `Google AdSense` یا `Yandex` در جایگذاری تبلیغات با پیاده‌سازی الگوریتم Simplex کمک میکنید تا علاوه بر پر کردن محل تبلیغات، نیازمندی های متقاضیان تبلیغ را پوشش دهید.

شما n مشتری دارید که هرکدام میخواهند تبلیغشان توسط تعدادی کاربر که در قرارداد ذکر شده دیده شود. شبکه ی تبلیغات آنلاین شما m جا برای تبلیغات دارد. شما میزان پولی که هر مشتری برای هر کاربر که تبلیغش را ببیند پرداخت میکند، تعداد حداقل کاربرانی که میخواهد تبلیغش را ببیند و همچنین تعداد کاربرانی که هر کدام از m جای مخصوص تبلیغات را میبینند در اختیار دارید. شما میتوانید تبلیغات متفاوت را در طول یک ماه در یک جا نمایش بدهید یا یک جا را فقط ب یک تبلیغ خاص اختصاص دهید. هدف بیشینه کردن سود که برابر جمع مبلغی است که هر مشتری برای تعداد کاربرهایی که تبلیغش را دیده پرداخت میکند. اگر فرض کنیم x_{ij} تعداد کاربرانیاست که تبلیغ امi را در جای امj دیده اند در این صورت میتوان تمام شروط را به صورت یک سری نامساوی خطی نوشت. برای مثال $\sum x_{ij} = S_j$ برابر تعداد کل کاربرانی است که تبلیغ i ام را میبینند و اگر مشتری i ام بخواهد تبلیغاتش توسط حداقل U_i کاربر دیده شوند آنگاه نامساوری $\sum x_{ij}U_i$ میبایست درست باشد. دقت کنید که $x_{ij}0$ است. اگر میزان مبلغی که مشتریiام برای هر کاربر که تبلیغش را در جای امj میبیند، با cij نشان دهیم، هدف مسئله بیشینه کردن $\sum\sum c_{ij}x_{ij}$ است که یک مسئله‌ی `linear programming` ولی با تعداد متغیرهای بیشتر است. فرمت ورودی: شما مسئله را ساده شده به صورت یک مسئله ی `linear programming` دریافت میکنید. در خط اول شما به ترتیب p و q را که نشان دهنده ی تعداد نامساوی ها و تعداد متغیرهاست را دریافت میکنید. در p خط بعد ماتریس A (ابعاد $p \times q$ است) را دریافت میکنید به صورتی که خط iام $a_{i1}, a_{i2}, ..., a_{iq}$ را شامل میشود. پس از دریافت ماتریس A در خط بعدی وکتور B به طول q را دریافت میکنید به طوری که $Ax < B$ است. در خط آخر شما وکتور C را دریافت میکنید که هدف مسئله بیشینه کردن $\sum z_i x_j$ با رعایت ضوابط است. فرمت خروجی: اگر هیچ رژیمی با این شرایط یافت نمیشد solution No و اگر تعداد زیادی رژیم با این محدودیت ها بود Infinity را به خروجی دهید. در غیر اینصورت solution Bounded را در خط اول و وکتوری از میزان پیشنهادی برای مصرف هر غذا در این رژیم پس از رند شدن به روش توضیح داده شده را در خط دوم بنویسید. (دقت کنید که $x = x_{ij}$ همان تعداد کاربرانی است که تبلیغiام در جای امj می‌بینند ولی در این سوال با $x_1, x_2, , x_q$ نماش داده می‌شوند).

مثال ۱)

```
3 2
-1 1
1 0
0 1
-1 2 2
-1 2
```

```
Bounded Solution
0 2
```

مثال ۲)

```
2 2
1 1
-1 -1
1 -2
1 1
```

```
No Solution
```

```
1 3
0 0 1
3
1 1 1
```

```
Infinity
```