تمرین ۴ درس ساختمان داده

محمد مهدی عبدالله پور
سارا کدیری
سید صالح اعتمادی

دانشگاه علم و صنعت ۹۹-۹۸

لطفا به نکات زیر توجه کنید:

- مهلت ارسال این تمرین شنبه ۲۷ مهر 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 تابع ها و تست ها باید همگی مطابق آنچه که خواسته شده است؛ باشد.
- نام شاخه، پوشه و پول ریکوست همگی دقیقا "A4" باشد.
- در صورتی که به اطلاعات بیشتری نیاز دارید می توانید با ایدی های تلگرام زیر در ارتباط باشید.

@m9m8a
@Codeiry

موفق باشید.

توضیحات کلی تمرین

تمرین این هفته ی شما، ۶ 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 A4 بسازید.

۲. فایل های cs. به پروژه ی خود اضافه کنید و در تابع Solve مربوط به کلاس هر سوال کد خود را بنویسید. توجه کنید که کلاس هر سوال تابع دیگری به نام Process نیز دارد که به طور کامل برای شما پیاده سازی شده است و نیاز به تغییر ندارد. اگر کنجکاوید می توانید با دیباگ کردن تست های خود، ساز و کار این تابع و پروژه ی TestCommon را متوجه شوید.

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 پروژه ی شما به پایان رسیده است، نوبت به پروژه ی تست می رسد. مراحل زیر را انجام دهید.

۱. یک پروژه ی تست برای پروژه ی خود بسازید.

۲. پوشه ی TestData را همان طور که در کلاس به شما گفته شده است به پروژه ی تست خود اضافه کنید. این پوشه در فایل زیپ ضمیمه شده قرار دارد.

۳. فایل GradedTests.cs را به پروژه ی تستی که ساخته اید اضافه کنید. توجه کنید که لازم نیست که برای هر سوال TestMethod بنویسید و تمامی تست ها از قبل نوشته شده اند. پاس شدن این تست ها در زمان مقرر معیار نمره ی تمرین شما هستند. به هیچ وجه متود ها و محدودیت زمانی هر متود را تغییر ندهید.

۱ Money Change

در این سوال، قرار است شما یک الگوریتم حریصانه ابتدایی که توسط صندوقداران در سراسر جهان میلیون ها بار در روز استفاده می شود، را طراحی و پیاده سازی کنید.

فرض کنید سه سکه با مقدار های ۱، ۵ و ۱۰ در اختیار دارید. حداقل تعداد سکه هایی که لازم است تا مقدار پول ورودی را با سکه های مذکور بسازید؛ چقدر است؟ برای حل این سوال الگوریتم حریصانه ای را در تابع Solve کلاس Q1ChangingMoney پیاده سازی کنید. در ورودی تابع به شما مقدار پولی که میخواهید خرد کنید داده شده است.

* محدودیت زمانی : **۲۰۰ میلی ثانیه**

۲ Maximum Value of the Loot

یک دزد غنیمت هایی پیدا کرده است که از ظرفیت کیسه اش بیشتر است. به او برای پیدا کردن با ارزش ترین ترکیب از غنیمت ها کمک کنید. فرض کنید که هر کسری از یک غنیمت را می تواند در کیسه خود قرار دهد. برای مثال اگر ظرفیت کیسه ی دزد ۱۰ باشد و یک عنمیت با ارزش ۳۰۰ و ظرفیت ۳۰ وجود داشته باشد؛ دزد می تواند کسری از غنیمت که ظرفیتش ۱۰ و ارزشش ۱۰۰ می باشد را بردارد و در کیسه ی خود بگذارد. الگوریتم خود را در تابع Solve کلاس Q2MaximizingLoot به صورت حریصانه بنویسید. در ورودی تابع، ظرفیت کیسه ی دزد، وزن هر غنیمت و ارزش آن ها داده شده اند.

* محدودیت زمانی : **۲۰۰ میلی ثانیه**

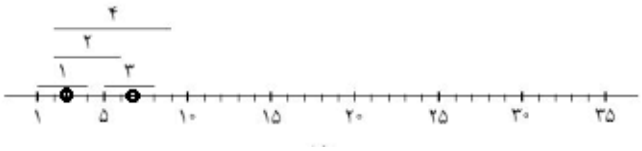
۳ Maximum Advertisement Revenue

شما n عدد آگهی برای قرار دادن در یک صفحه اینترنتی محبوب را دارید. برای هر آگهی، شما می دانید چه مقدار هزینه قرار است آگهی دهنده به ازای یک کلیک در این آگهی پرداخت کند. شما در صفحه خود n تا Slot را تنظیم کرده اید و تعداد کلیک های مورد انتظار را برای هر Slot در هر روز تخمین زده اید. شما باید الگوریتمی طراحی کنید که آگهی ها را میان Slot ها به گونه ای توزیع کند که درآمد کل به حداکثر برسد. دقت کنید که در خط اول هر TestCase تعداد Slot ها یا آگهی ها می باشد. سپس در خط های بعدی عدد اول هزینه ی یک کلیک بر روی آگهی i ام است و عدد دوم میانگین تعداد کلیک هایی در روز است که بر روی $slot_i$ ام انجام می شود. بنابراین شما هزینه ی کلیک بر روری یک آگهی را در یک آرایه مانند a_i و میانگین تعداد کلیک ها در روز را در آرایه دیگری مانند b_i بریزید. سپس این دو آرایه را به همراه تعداد Slot ها یعنی n به عنوان ورودی به تابع الگوریتم خود بدهید.

* محدودیت زمانی : **۲۰۰ میلی ثانیه**

۴ Collecting Signatures

شما مسئول جمع آوری امضا از همه مستاجران یک ساختمان خاص هستید. برای هر مستاجر، شما می دانیدکه در چه بازه ی زمانی در خانه است. شما باید تمام امضا ها را تا جایی که ممکن است با کمترین تعداد مراجعه به ساختمان، جمع آوری کنید. مدل ریاضی برای این مشکل به صورت شکل زیر است. شما مجموعه ای از بازه ها را در یک خط قرار داده اید و هدف شما این است که کمترین تعداد نقطه ها را بر روی خط پیدا کنید که هر بازه حداقل شامل یک نقطه باشد. دقت کنید که در خط اول هر TestCase تعداد بازه ها می باشد و در خط های بعدی، به ترتیب عدد اول شروع بازه و عدد دوم پایان بازه می باشد. بنابراین شما شروع بازه ها را در یک آرایه مانند a_i و پایان بازه ها را در آرایه دیگری مانند b_i بریزید. سپس این دو آرایه را به همراه تعداد بازه ها یعنی n به عنوان ورودی به تابع الگوریتم خود بدهید.



❊ محدودیت زمانی : **۲۰۰ میلی ثانیه**

۵ Maximum Number of Prizes

فرض کنید قرار است شما یک رقابت هیجان انگیز را برای کودکان سازماندهی کنید. یک صندوق جایزه دارید که n آب نبات در آن وجود دارد. شما باید این آبنبات ها را به بچه هایی که در مکان های یک تا k در مسابقه قرار گرفته اند، به عنوان جایزه بدهید؛ به گونه ای که رتبه های بالاتر تعداد بیشتری آبنبات بگیرند. شما باید k را به گونه ای پیدا کنید که تا جای ممکن تعداد بیشتری از بچه ها جایزه بگیرند و خوشحال شوند.در واقع مدل ریاضی این سوال به این صورت است که یک عدد صحیح مثبت n را به صورت جمعی از یک سری اعداد صحیح مثبت که دو به دو نیز متمایزهستند؛ دربیاوریم به گونه ای که تعداد این اعداد بیشترین مقدار ممکن باشد.

❊ محدودیت زمانی : **۵۰۰ میلی ثانیه**

۶ Maximum Salary

فرض کنید برای استخدام در یک مصاحبه شرکت کرده اید و به عنوان آخرین سوال مصاحبه، رئیس شما به شما چند قطعه کاغذ با اعداد روی آن می دهد و از شما می خواهد بزرگترین شماره را از این اعداد بنویسید. پاسخ شما همان حقوق شما خواهد بود، بنابراین شما بسیار علاقه مند به حداکثر رساندن این عدد هستید. چطور می توانید این کار را بکنید؟ در ویدئو های درس، الگوریتم زیر را برای ساختن بزرگترین عدد از شماره های تک رقمی داده شده در نظر گرفتیم.

```
LargestNumber(Digits):
answer ← empty string
while Digits is not empty:
  maxDigit ← −∞
  for digit in Digits:
    if digit ≥ maxDigit:
      maxDigit ← digit
  append maxDigit to answer
  remove maxDigit from Digits
return answer
```

متاسفانه این الگوریتم تنها در صورتی کار می کند که ورودی از اعداد تک رقمی باشد. به عنوان مثال، برای یک ورودی متشکل از دو عدد صحیح ۲۳ و ۳ (۲۳ عدد یک رقمی نیست!) الگوریتم عدد ۲۳۳ را برمی گرداند، در حالی که بزرگترین عدد در واقع ۳۲۳ است. به عبارت دیگر، استفاده از بزرگترین عدد از ورودی به عنوان شماره اول همیشه ما را به جواب درست نمی رساند. هدف شما در این مشکل این است که الگوریتم بالا را بهینه سازی کنید تا کارایی آن نه تنها برای عدد تک رقمی، بلکه برای هر عدد صحیح دلخواه مثبت باشد.

❊ محدودیت زمانی : **۲۰۰ میلی ثانیه**