به نام خالق یکتا و به یاد یوسف زهرا

گزارش تمرین برنامه‌سازی پیشرفته

تمرین سری پنجم

سیدرضا هاشمی‌راد

۱۰ خرداد ۱۳۹۸

فهرست مطالب

لینک دستیابی به فایل ها در گیت هاب

۱ سوال یک

این سوال مانند سوالات تمرین سری گذشته، استفاده از کتابخانه‌های مناسب استاندارد C++ است. در ابتدا برای پر کردن وکتورهای با اعداد پشت سر هم از `std::iota` استفاده می نماییم که با دادن ورودی و خروجی وکتور آن را پر می کنیم. برای پرینت کردن هم مشابه تمرین سری قبل از `std::copy` استفاده می نماییم. برای اضافه کردن یک وکتور به وکتور دیگر، از `std::insert` استفاده می نماییم. همچنین برای اینکه اعداد فرد را جدا نماییم، با یک لاندا فانکشن اسن کار را انجام می دهیم و المان های فرد را پیدا می کنیم. برای برعکس کردن وکتورها، از شمارنده معکوس `rbegin` و `rend` استفاده نمودیم. اما برای سرت کردن، از `std::sort` استفاده نمودیم اما برای پردازش موازی، می خواستیم از کتابخانه `<execution>` استفاده نماییم که به دلیل نامشخصی نه در ویندوز و نه در لینوکس، این کتابخانه پیدا نشد.

۲ سوال دو

در پایتون مثل تمام زبان‌های برنامه نویسی این امکان را داریم که برای یک تابع ورودی های نامحدود بگیریم. این کار توسط `args*` و `keywords**` صورت می پذیرد. وقتی آرگومان ها را با `args*` به تابع ارسال می نماییم، در واقع یک توپل را ارسال نموده ایم که می توان در تابع از آن استفاده نمود. وجود * قبل از نام متغیر نشان دهنده توپل بودن آن است. همچنین اگر آرگومان ها را با `keywords**` به تابع ارسال کنیم، آنها به صورت دیکشنری وارد تابع می شوند. و می توان سایر ورودی ها را به این صورت دریافت نمود.

مثال در کد آورده شده است.

۳ سوال سه

خروجی متغیر `A0` یک دیکشنری است که به صورت متناظر و یک به یک اعداد و حروف را به هم نگاشت می کند. بنابراین برای دستیابی به مقدار `'1'` باید از کلید `'a'` استفاده شود و به همین ترتیب.

خروجی متغیر `A1` یک سری از اعداد صحیح است که معمولا در حلقه ها استفاده می شود و خروجی خاصی ندارد. در واقع یک نوع تایپ است.

خروجی متغیر `A2` یک لیست است که بین اعداد یک تا ۹ را جستجو کرده و مقادیری که در `A0.values()` یک خود یک لیست را نمایش می دهد. نکته اینجاست که در صورتی که این متغیر ر اعین صورت سوال استفاده نماییم خروجی یک لیست خالی خواهد بود چون اول شرط بودن و نبودن را در نهایت باید در یک لیست چک کرد و نه در یک دیکشنری. برای همین این شرط در `A0.values()` چک شده است و همچنین چون مقادیر ما به صورت استرینگ ذخیره شده اند باید برای چک کردن هر اندیس آن را به استرینگ تبدیل نمود. در نهایت این متغیر را به صورت ذیل نوشتیم:

```
A2 = [i for i in A1 if str(i) in A0.values()]
```

خروجی متغیر A3 یک لیست است که در واقع مرتب شده لیست ورودی آن به صورت صعودی است. اما در اینجا نیز به دلیل کاراکتر بودن مقدار A0[i] لازم است تا آن را به int تبدیل نماییم که عمل مرتب سازی به درسی انجام شود، در غیر این صورت اگر اعداد بیش از یک رقم باشند تنها کارکتر اول مقایسه می شود و طبیعتا مرتب سازی درست نخواهد بود.

خروجی متغیر A4 در واقع یک لیست به طول ۱۰ است که هر یک از عناصر آن خود یک لیست به طول ۲ است و درایه اول آن عدد بین صفر تا ۹ (متغیر A1)و درایه دوم آن توان دوم متناظر آن عدد است.

برای پرینت کردن نیز به سادگی با چک کردن اینکه شمارنده در لیست مورد نظر وجود دارد یا خیر، آن را پرینت می کنیم. برای زیبایی خروجی به صورت عمل می کنیم که در ۴ پرینت اول، کارکتر انتهایی پرینت که به طور پیش فرض
n است را روی قرار می دهیم و تنها برای آخرین پرینت در هر حلقه آن را همان مقدار پیشفرض قرار می دهیم.

۴ سوال چهار

با توجه به توضیحات صورت سوال روند این سوال به سادگی انجام پذیرفت. ابتدا توابع خواسته شده را می نویسیم و پله به پله از آنها استفاده می کنیم. یک تابع برای اینکه هر نقطه داخل دایره قرار می گیرد یا خیر. دوما تابعی می نویسیم که به تعداد ورودی عدد تولید کرده و نسبت آن ها را در خروجی بدهد.

سپس تابع find() را می نویسیم به این صورت که تا زمانی که خطا بیشتر از یک صدم است، عدد تولید شود و تعداد اعداد را در تابع قبلی زیاد می نماییم تا به خطای مورد نظر برسیم. سپس تابع انتهایی را برای فیلتر میانگین گیر می نویسیم که به تعداد ورودی تابع find() را اجرا می نماییم و در انتها از نتایج به دست آمده میانگین می گیریم.

۵ سوال پنج

حل این سوال روند روالی داشت و طبق توضیحات ماژول os.path به خوبی انجام شد. تنها تذکر به یک نکته ضروری است و آن این است که در هنگام آدرس دهی برای دادن برخی آدرس ها به توابع با مشکل ترجمه \ در رشته مواجه شدیم که برای این کار چندین روش وجود دارد که یکی از آنها استفاده از کلید r قبل از رشته است که متاسفانه برای ورودی به صورت متغیر قابل استفاده نیست. روش بعدی استفاده از \\ به جای یک عدد است که در این حالت مشکل قبلی رخ نمی دهد. روش سومی هم نیز وجود دارد و آن استفاده از / است که اگر چه طبق روال معمول آدرس دهی در ویندوز نیست ولی خوشبختانه پایتون روی ویندوز می تواند به خوبی با آن کار کند و مدیریت فایل ها را انجام دهد.

کار اضافه: برای بهبود رابط کاربری کد مدیریت فایل، ابتدا نوع کاری که مخاطب می خواهد انجام دهد را از او می پرسیم و سپس با گرفتن اسم و آدرس، کار خواسته شده را انجام می دهیم. نکته این جاست به دلیل اینکه در هنگام حذف کردن فایل دیگر هیچ دسترسی به فایل نخواهیم داشت، ابتدا از مخاطب پرسیده می شود که آیا مطمئن به حذف کردن هست یا خیر و در صورت پاسخ صحیح، فایل را حذف می نماییم.

۶ سوال شش

به دلیل توضیحات کامل و جامع در صورت سوال، روند کد زدن برای این سوال نیاز به سادگی انجام پذیرفت و نکته ی خاصی نبود. دقیقا توابع مورد نیاز در محاسبه انتگرال را نوشتیم. همچنین به دلیل استفاده از اسم های متناسب برای هر تابع، دیگر از کامنت گذاری بیش از حد

جلوگیری شد. به طور کلی محاسبه انتگرال به این روش به دو بخش کلی تقسیم می شود، یکی محاسبه ضرایب و یکی هم محاسبه صفر تابع لاگرانژ که از روش نیوتن استفاده می کنیم که از رابطه ذیل محاسبه می شود:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad (1)$$

برای محاسبه ضرایب هم تابع لاگرانژ را به صورت بازگشتی اجرا می نماییم.

۷ سوال هفت

حل این سوال بدون محدودیت تعداد خطوط کد، کاری بسیار آسان است اما برای پیاده سازی آن با تعداد خطوط محدود لازم است تا از لیست های پیشرفته پایتون استفاده نماییم. این کار هم در نگاه اول بسیار ساده به نظر می رسد اما نکته اینجاست که ما برای حل این مسئله در هر لحظه هم المان و هم به اندیس آن نیاز داریم. اما چون فقط یک جا لیست ورودی را می توانیم صدا بزنیم، لذا کار کمی پیچیده می شود. در روش دوم، لیست ورودی را در یک متغیر ریخته و در سطر دوم از آن برای تولید لیست و در نهایت رشته نهایی استفاده می نماییم.

در روش اول استفاده شده، برای اینکه کد را به یک خط برسانیم سعی بر این داشتیم که ابتدا تمام المان هایی که اندیس آنها مضربی از شش است را داخل یک لیست ریخته و در ادامه با بررسی اینکه کدام از آنها بر شش بخش پذیر هستند، لیست و در نهایت رشته نهایی را بسازیم. اما به دلیل محدودیتی که در ورودی گرفتن داشتیم این عمل میسر نشد و باز هم مجبور به نوشتن دو خط کد شدیم. در واقع خط ذیل برای دسترسی به المان های لیست ورودی و جداسازی المان ها با اندیس مضرب ۶، اجرا نمی شد!!

```
input().split(' ')[i]
```

۸ ارسال روی گیت

برای ارسال روی گیت، مطابق آنچه در کلاس گفته شد ابتدا یک ریپازیتوری[1] به نام درس برنامه سازی پیشرفته درست کردیم و سپس در سیستم لوکال خود آن را دانلود کرده و فایل های مورد نظر را با پوشه بندی مناسب در دایرکتوری ایجاد شده قرار دادیم. با استفاده از دستور

```
git add DIRECTORYofFILES
```

فایل ها را اضافه کرده و سپس با دستور

```
git commit -m "MYCOMMENT"
```

آنها را روی سیستم لوکال کامیت یا ثبت می کردیم. حال با دستور

```
git push origin master
```

فایل های کامیت شده را روی سرور github.com و روی برنچ مستر بارگذاری نمودیم.

توجه نمایید که در ابتدا برای گرفتن ریپو از دستورهای ذیل به ترتیب استفاده نمودیم. (از طریق ssh)

```
git init
git clone git@github.com:SRHashemirad/AUT_AP_course.git
```

[1] Repositoriy.

همچنین قابل ذکر است که فایل `gitignore.` را به به صورتی قرار می دهیم که فایل هایی که در حین کامپایل ایجاد می شوند[۲] را ثبت نکرده و همچنین نسخه هایی که نرم افزار emacs ایجاد می کند[۳] را نیز ثبت ننماید.

شما می توانید برای دستیابی به فایل ها در گیت هاب از این لینک اقدام نمایید.

با تشکر فراوان از حسن دقت نظر و توجه شما

[۲] *.o

[۳] *.*~