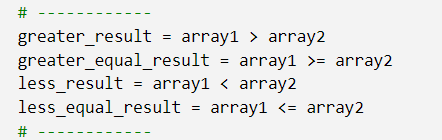
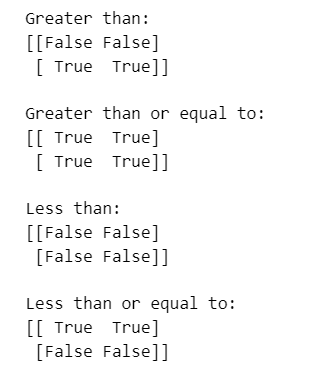
گزارش تمرین CI\_HW1 – محمد اصولیان 99521073

# Q1)

با توجه به این که جنس ورودی‌ها از جنس آرایه numpy هستند، از عملگرهای بزرگتری کوچکتری خود numpy استفاده کردم:

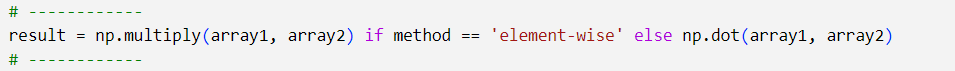


نتیجه تست برای آرایه‌های نمونه مشخص شده:



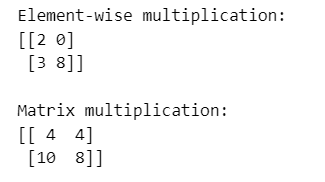
# Q2)

با چک کردن method، ضرب‌های خواسته شده را به کمک توابع numpy انجام دادم:



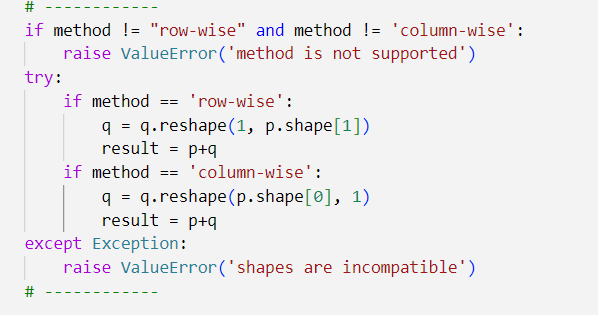
چون در صورت سوال مشخص نشده بود که مقدار متغیر method برای حالت دیگر یعنی ضرب ماتریسی چه هست، به ازای هر مقداری جز element-wise، ضرب ماتریسی انجام دادم.

نتیجه تست:



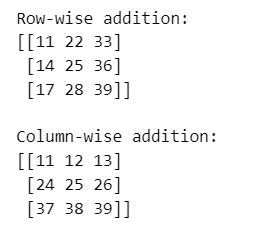
# Q3)

ابتدا مقدار method را چک کردم و در صورتی که متد پشتیبانی نمی‌شد، ValueError دادم. سپس با توجه به مقدار method، عملیات خواسته شده را انجام دادم



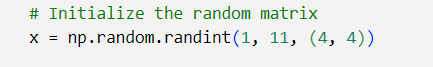
برای این کار از توابع broadcasting خود numpy استفاده کردم. در صورتی که عملیات row-wise باشد، ماتریس دوم را به حالت سطری و در غیر این صورت به حالت ستونی درآوردم تا numpy اروری ندهد. در صورتی که ابعاد این دو ماتریس با هم تطابق نداشته باشند، ValueError داده می‌شود.

نتیجه تست:

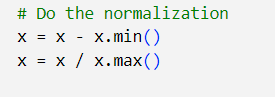


# Q4)

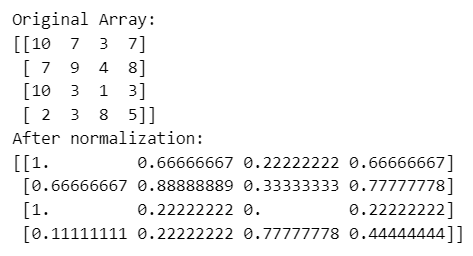
ماتریس اولیه را با استفاده از np.random.randint ایجاد کردم



سپس با توجه به مقدار مینیمم و ماکسیسمم، نرمالایزیشن انجام دادم.

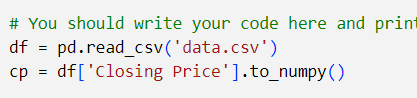


نتیجه تست:



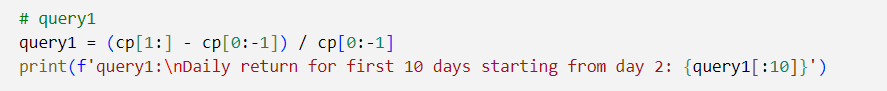
# Q5)

فایل را به کمک pd.read\_csv خواندم و مقادیر closing price را در یک آرایه np ذخیره کردم:

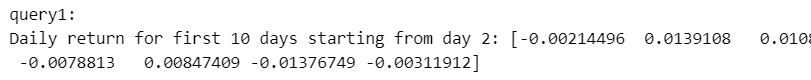


## Q5-1)

طبق فرمول گفته شده مقدار بازده روزانه را حساب کردم و ده مقدار اول را برای تست نشان دادم.

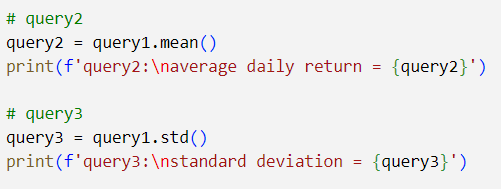


نتیجه:

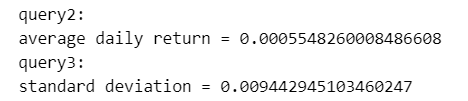


## Q5-2 , Q5-3)

برای محاسبه میانگین و انحراف معیار از توابع np.mean و np.std استفاده کردم

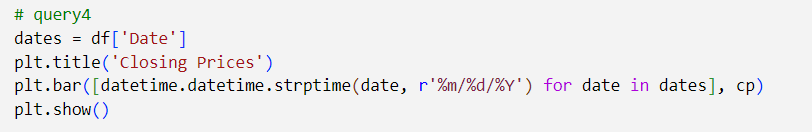


نتیجه:

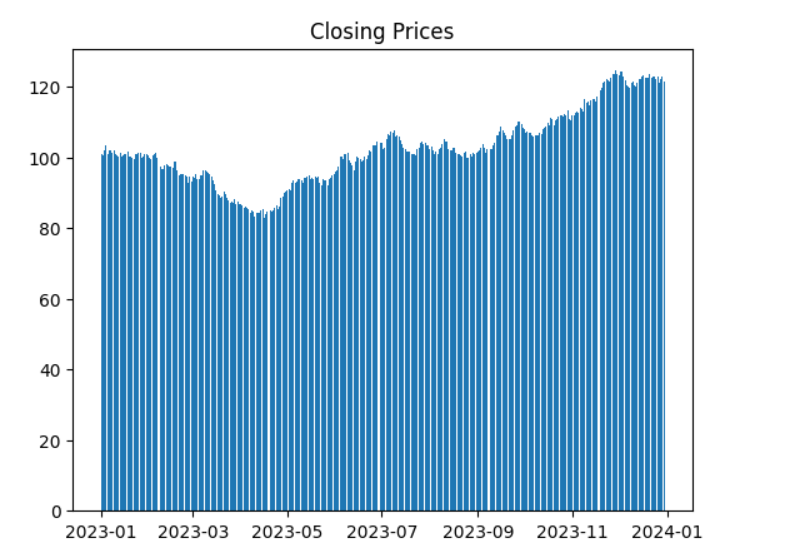


## Q5-4)

ابتدا تاریخ هایی که به صورت string بودند رو به کمک datetime.strptime به فرمت تاریخ درآوردم. بعد به کمک plt.bar و مقادیر closing price که از قبل ذخیره داشتم، نمودار را رسم کردم

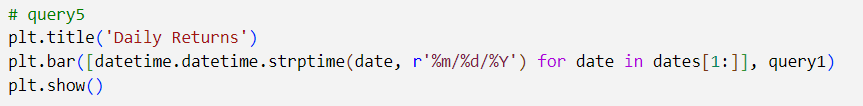


نتیجه:

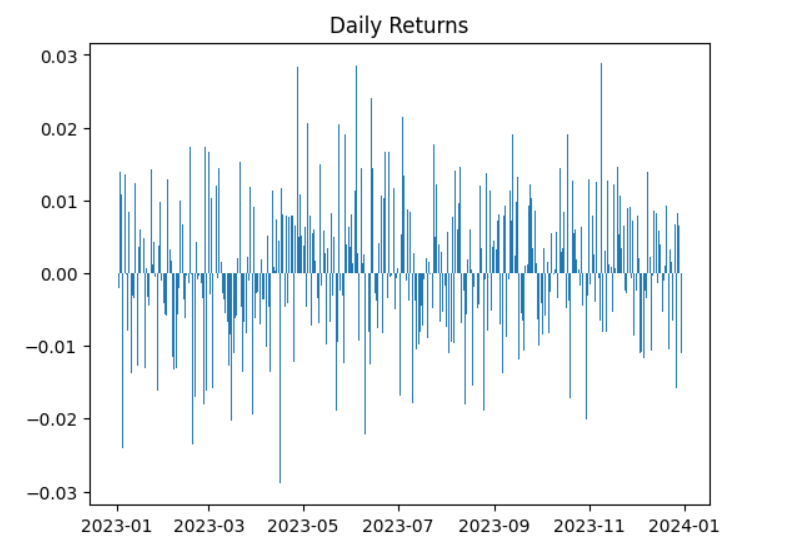


## Q5-5)

مثل سوال قبل، فقط به جای مقادیر closing price از مقادیر بازده روزانه که در بخش اول به دست آورده بودم استفاده کردم

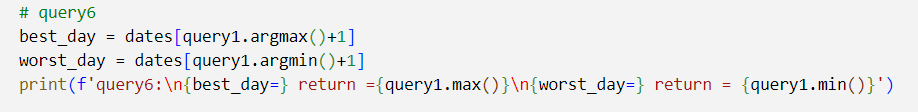


نتیجه:

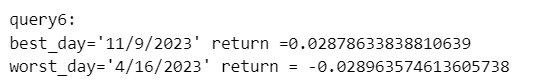


## Q5-6)

بیشترین مقدار و اندیس آن را به کمک np.max و np.argmax پیدا کردم و به کمک اندیس، روز مربوطه رو پیدا کردم. برای کم ترین مقدار هم همینطور

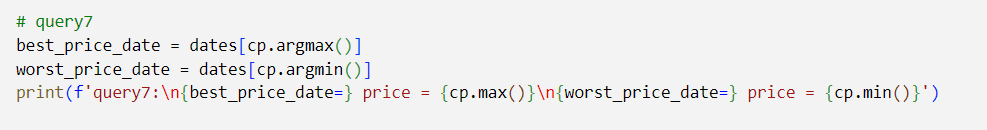


نتیجه:

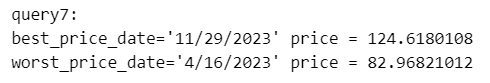


## Q5-7)

باز مانند سوال قبل عمل کردم اما با مقادیر closing price

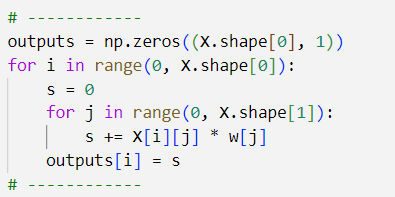


نتیجه:

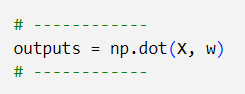


# Q6)

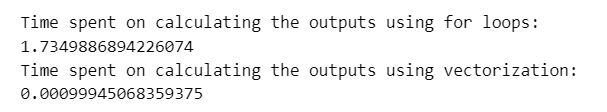
برای قسمت حلقه، یک حلقه روی نمونه ها زدم و برای هر نمونه یک حلقه زدم و مقادیر فیچر و وزن را ضرب و جمع کردم و به این صورت آرایه f را پر کردم



اما برای قسمت vectorization، تنها به کمک تابع np.dot، ماتریس X و w را در هم ضرب کردم



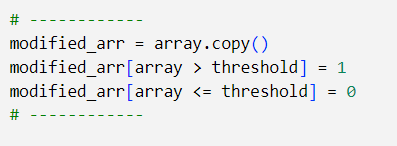
نتیجه اجرا ها به این صورت بود:



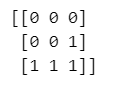
میبنیم که در اجرا به کمک vectorization، عملیات تا هزار برابر سریع تر انجام می‌شود. این به خاطر این است که numpy، این عملیات ها را به کمک کدهای c++ که سرعت بیشتری دارند اجرا میکند و نکته مهمتر این است که ضرب ماتریسی را به صورت parallel انجام میدهد.

# Q7)

از سینتکس مخصوص آرایه‌های numpy برای انتخاب عناصر آرایه که از threshold بیشتر هستند استفاده کردم و مقدار آنها را با 1 جایگزین کردم. سایر عناصر هم به همین ترتیب با 0 جایگزین کردم



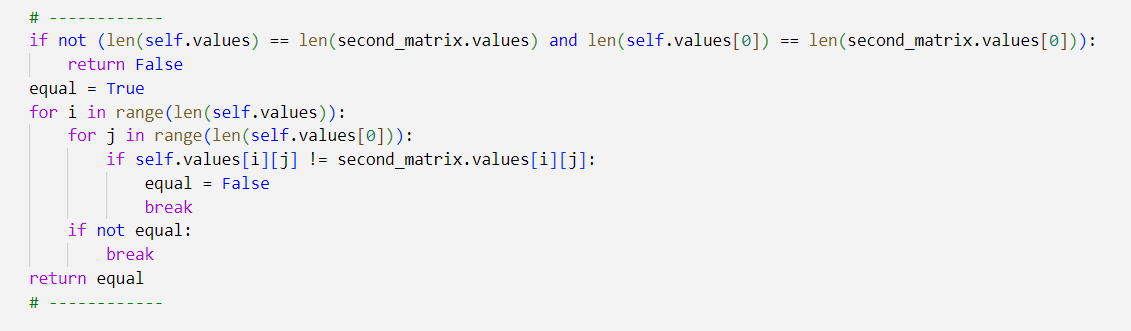
نتیجه:



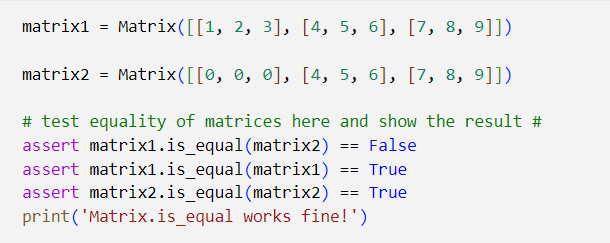
# Q8)

## Q8-1)

ابتدا ابعاد ماتریس ها را چک کردم و در صورتی که برابر نبودند، مقدار False خروجی دادم. سپس روی تمام اعضای ماتریس‌ها حلقه زدم و در صورتی که یک عنصر نابرابر داشتند، False خروجی دادم

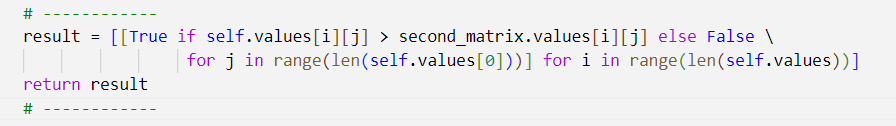


تست را به این صورت نوشتم و با موفقیت اجرا شد:

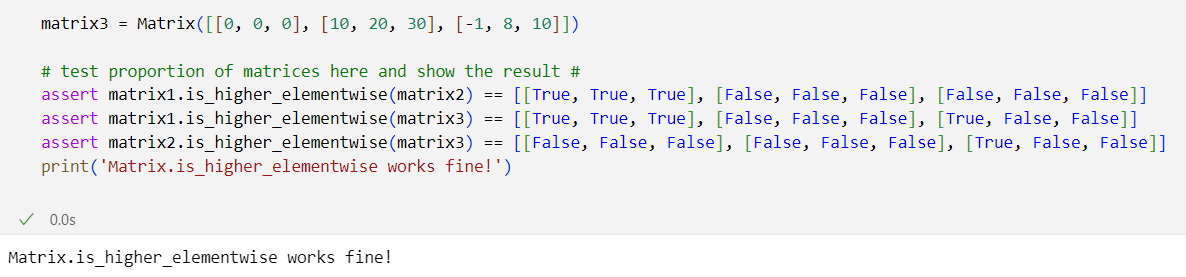


## Q8-2)

با استفاده از generator های لیست در پایتون، عناصر را با هم مقایسه کرم و مقدار False یا True را برای هر خانه انتخاب کردم

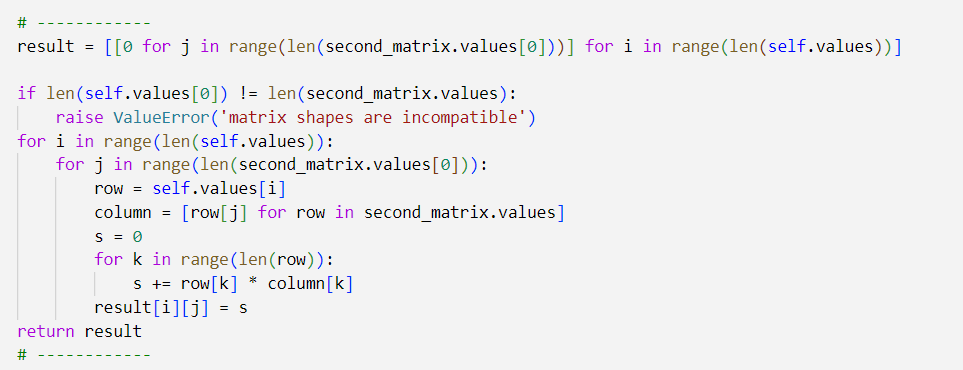


نتیجه تست:

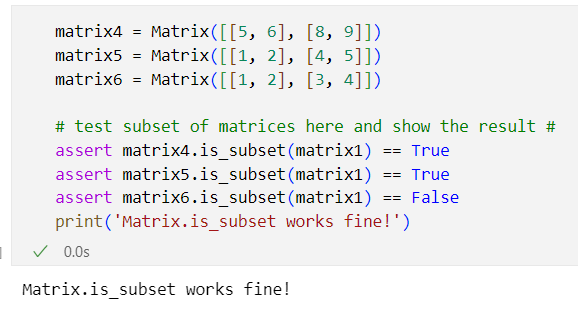


## Q8-3)

ابتدا چک کردم که ابعاد ماتریس زیرمجموعه، از ابعاد ماتریس دیگر کمتر باشد. سپس تمام حالات ممکن وجود یک ماتریس به ابعاد ماتریس زیرمجموعه را در ماتریس بزرگ تر بررسی کردم و با استفاده از تابع is\_equal که عقب تر نوشته بودم، تساوی این حالات با ماتریس زیرمجموعه را بررسی کردم. در صورتی که هیچ حالتی صدق نکرد، False برگرداندم.

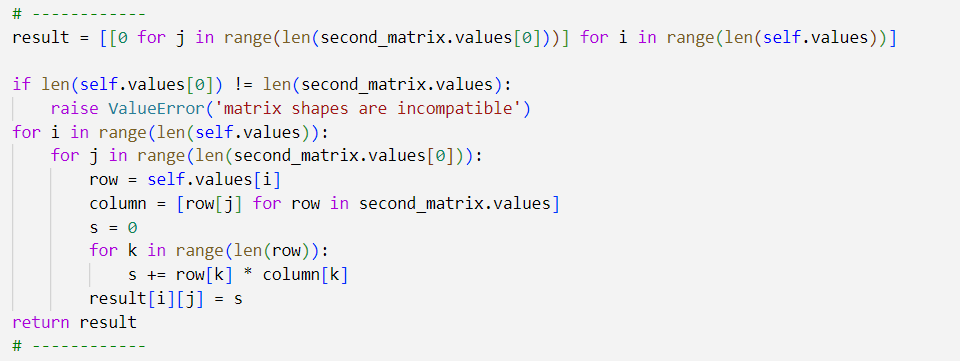


نتیجه تست:



## Q8-4)

ابتدا ابعاد ماتریس اول و دوم را چک کردم و در صورتی که با هم تطابق نداشتند، ValueError دادم. سپس روی سطرهای ماتریس اول و ستون های ماتریس دوم حلقه زدم و عناصر را در هم ضرب و جمع کردم و ماتریس نتیجه را خانه به خانه پر کردم



نتیجه تست:

