

به نام خدا

گزارش تمرین 1

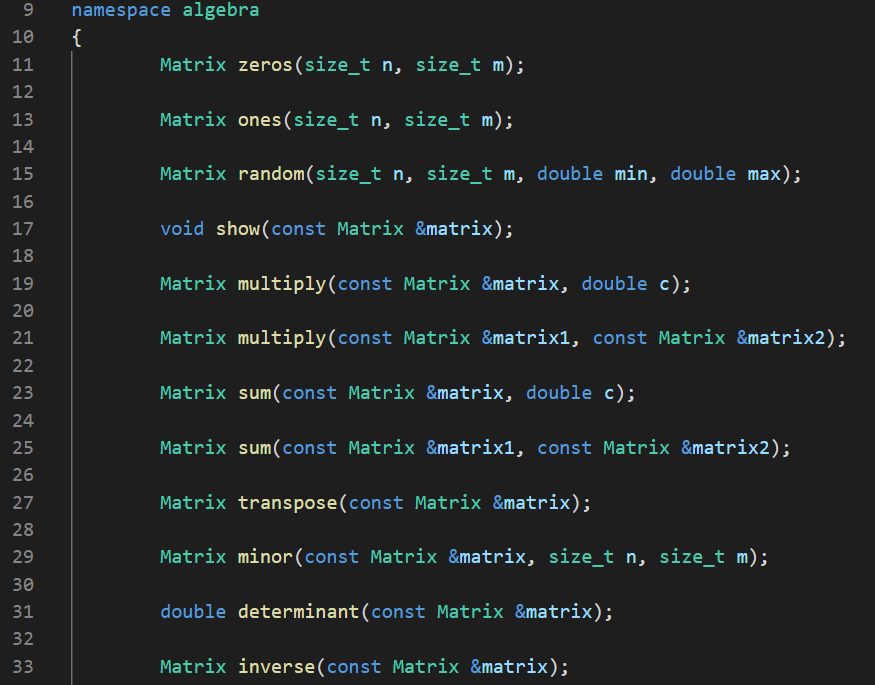


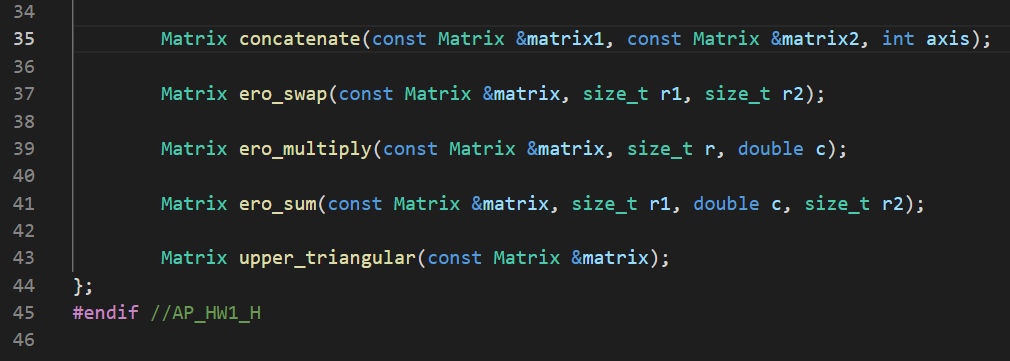
محمد برآبادی 9723015

در ابتدا با استفاده از فیلم جلسه اول کلاس تدریسیاری تنظیمات اولیه را انجام دادم تا VS Code را آماده کرده و با اتصال به Host 127.0.0.1 شروع به کدزنی کردم.

* فایل hw1.h :

با توجه به توضیحات داده شده در github ابتدا در فایل .h تابع‌ها را در یک namespace تعریف میکنیم.





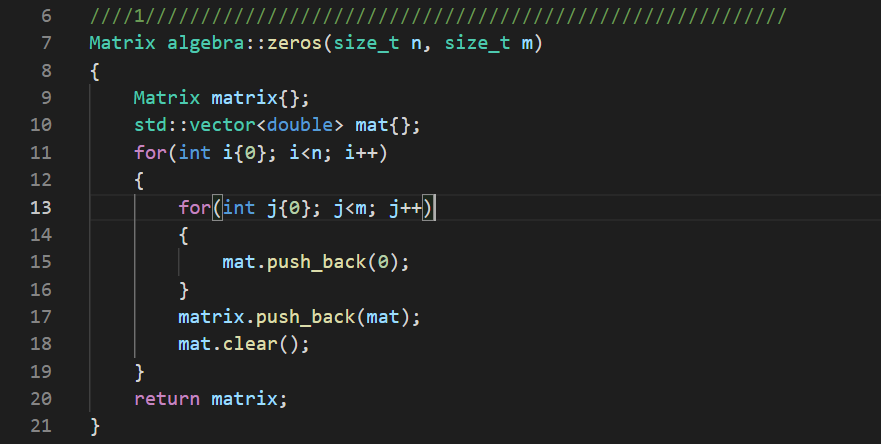
* فایل hw1.cpp:

در مرحله بعد قصد داریم که هر تابع را پیاده‌سازی کنیم با توجه به نتیجه‌ای که از ما خواسته شده و همچنین unit\_test موجود در پوشه.

1. Zeros:

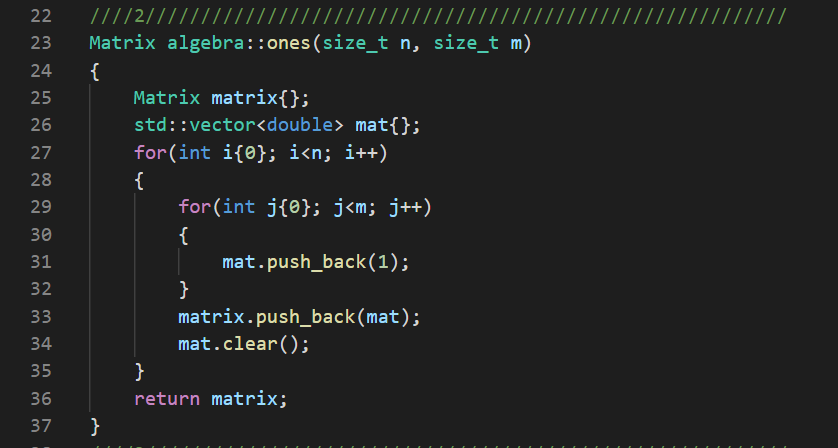
نکته مهمی که در ابتدا باید ذکر شود استفاده از vector میباشد و برای همین برای اضافه کردن هر مقدار باید آن را با دستور push\_back اضافه کنیم.

هر سطر را در یک for ساخته و در ماتریس قرار میدهیم.

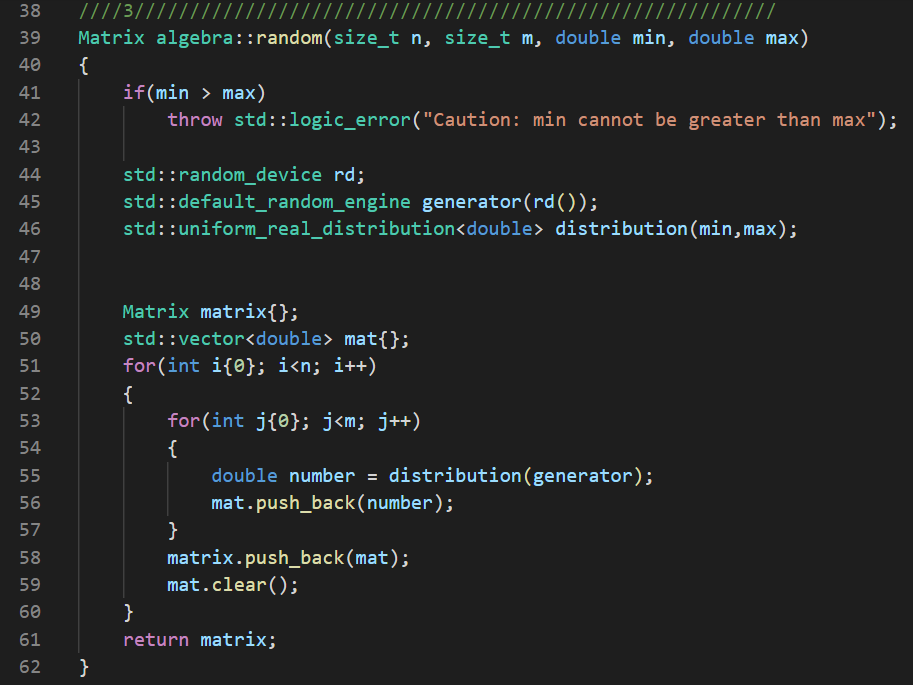


1. Ones:

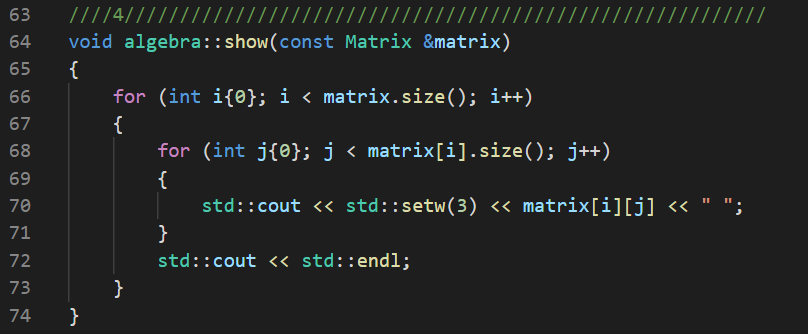
مانند تابع قبل میباشد و فقط تنها فرق این میباشد که باید همه ماتریس را با 1 به جای 0 پر کنیم.



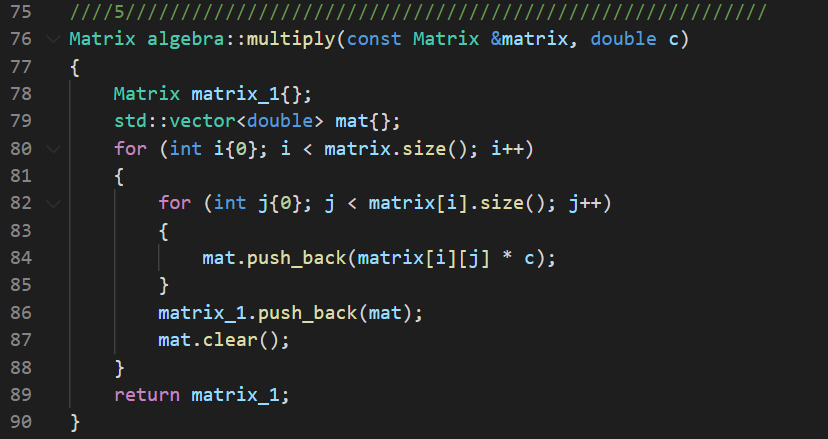
1. Random:

در این تابع قصد داریم ماتریس را به صورت رندوم با اعدادی بین min و max پر کنیم. یکی از نکاتی که باید رعایت شود این میباشد که مقدار min از max بزرگتر نباشد چون به صورت منطقی با مشکل مواجه میشویم.با توجه به صورت تمرین از rand و srand نمیتوانیم استفاده کنیم و از random استفاده میکنیم و دستورات مربوط به آن را از اینترنت پیدا کردیم.

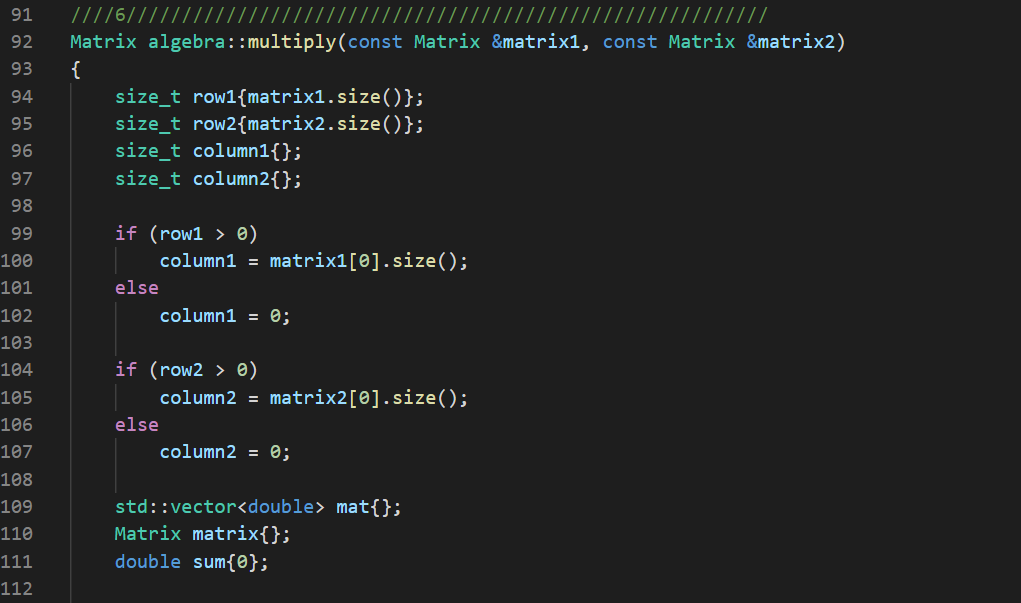
1. Show:

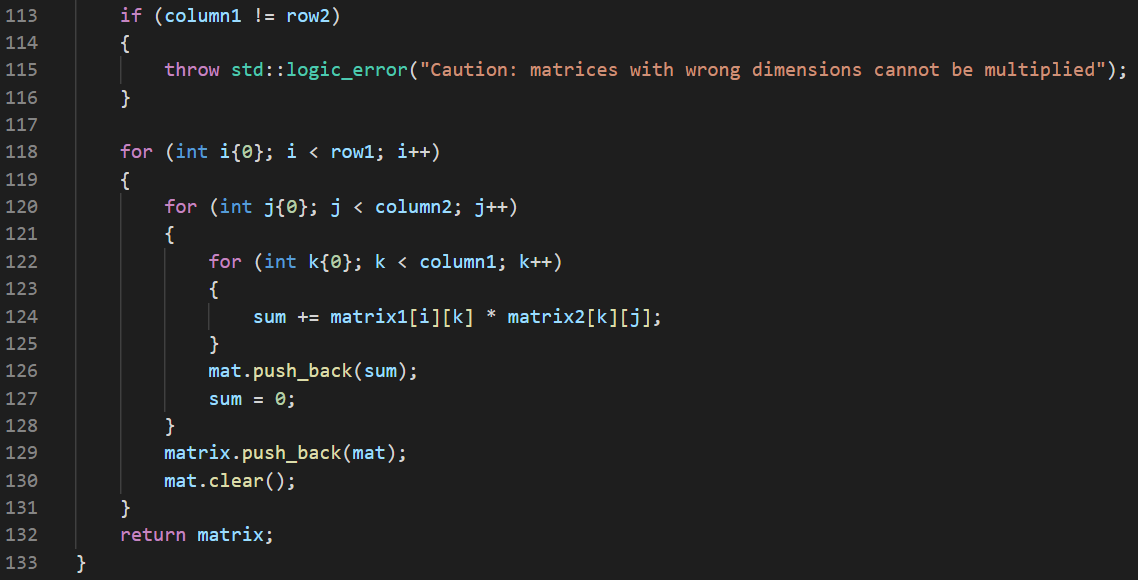
در این تابع ماتریس را نمایش میدهیم ،با همان ملاحظاتی که گفته شده است

1. Multiply:

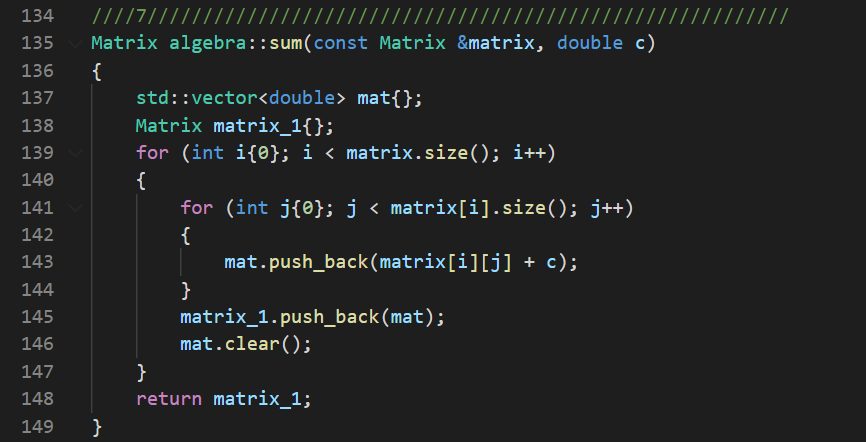
ما در این تابع قصد داریم یک عدد را در مانریس ضرب کنیم.مانند تابع 1و2 که مقدار صفر و یک در ماتریس قرار میدادیم اینبار هر درایه ماتریس را ضرب میکنیم در عدد وارد شده و در ماتریس قرار میدهیم. 

1. Multiply:

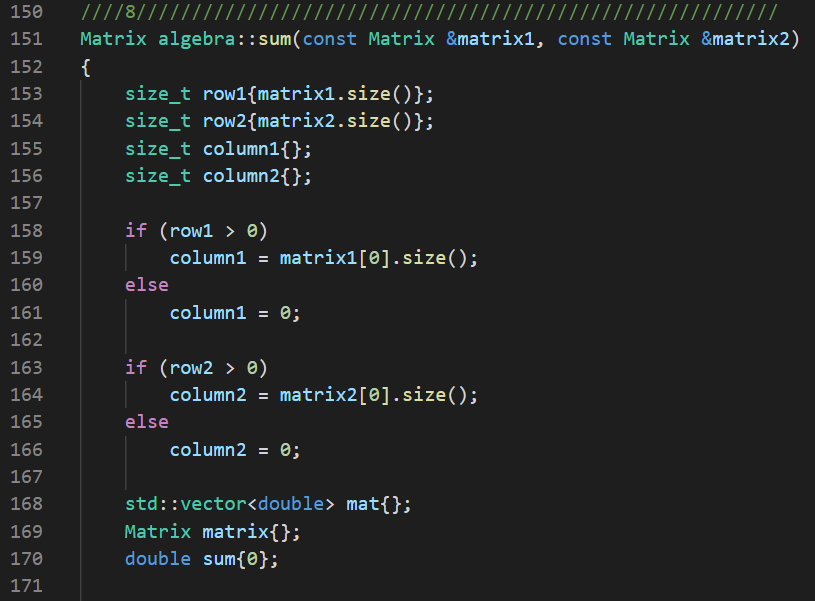
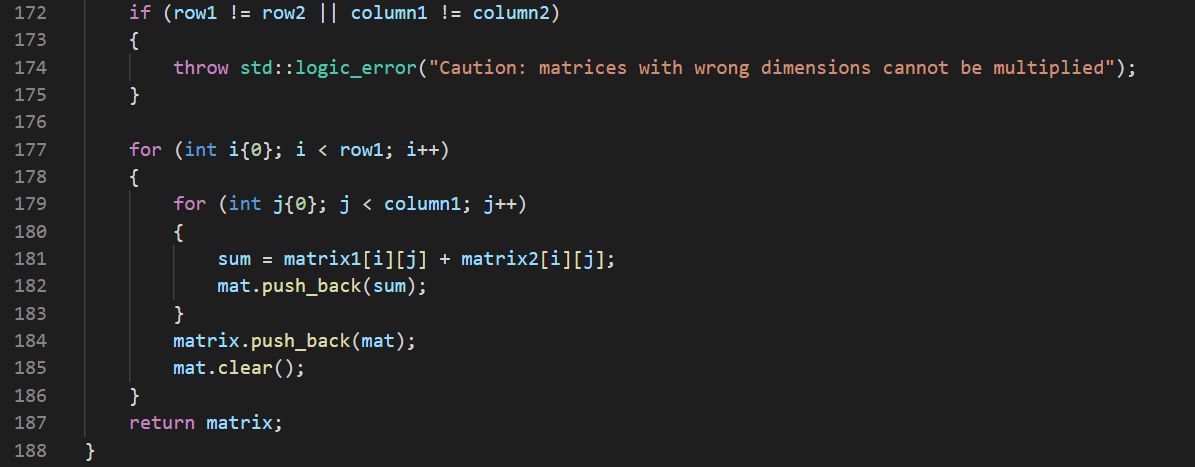
این تابع ضرب دو ماتریس را داریم ولی باید در ابتدا بررسی شود که ماتریس ها خالی نباشند و همچنین امکان ضرب آن‌ها وجود داشته باشد یعنی اینکه تعداد ستون‌های ماتریس اول با تعداد سطرهای ماتریس دوم برابر باشد.

اگر این شرایط وجود داشته باشد سپس ضرب درایه‌ها را انجام میدهیم.

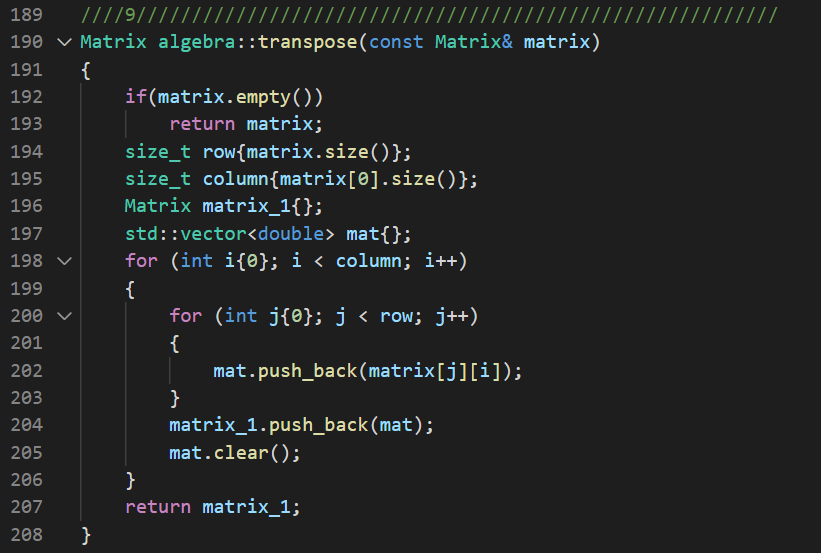
1. Sum:

مانند تابع ضرب با عدد اسکالر، هر درایه با یک عدد جمع میشوند.

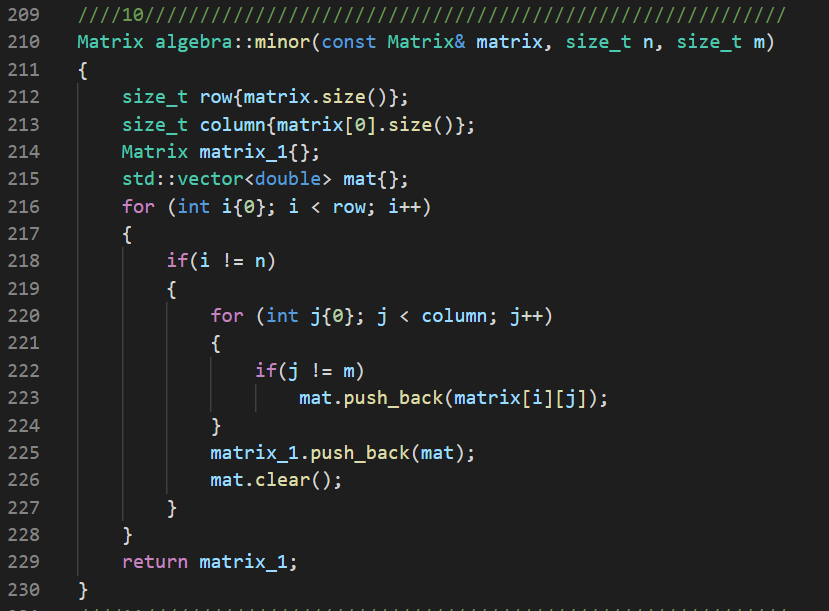
1. Sum:

باز باید بررسی شود که ماتریس‌ها تهی نباشند و برای انجام جمع باید ستون‌ها و سطرها با هم برابر باشند و در نهایت هم درایه‌های متناظر با هم جمع میشوند و در ماتریس جدید قرار میگیرند.

1. Transpose:

در ابتدا بررسی میشود که آیا ماتریس تهی است یا خیر و اگر تهی باشد، ترانهاده آن همان ماتریس تهی میباشد. و در با حرکت در ستون ماتریس ، سطر ماتریس جدید را میسازیم و به این شکل ماتریس ترانهاده ایجاد میشود.

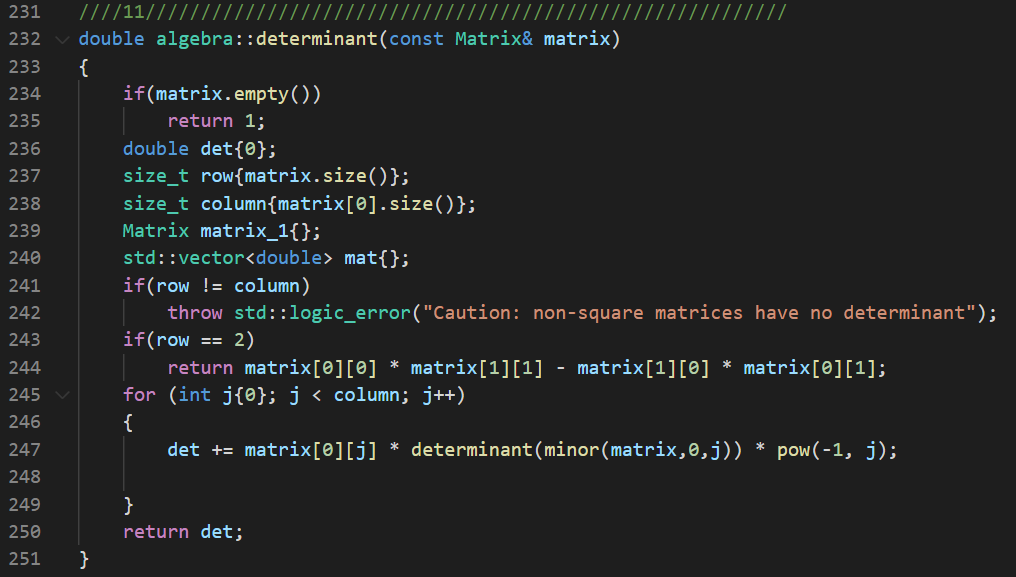
1. Minor:

ماینور n,m به این معنا است که سطر n و ستون m را حذف کنیم.

1. Determinant:

ماتریس اگر تهی باشد دترمینان 1 را خروجی میدهیم و اگر ماتریس مربعی نباشد اصلا دترمینانی وجود ندارد و در غیر این دو صورت میتوانیم دترمینان را بدست آوریم.

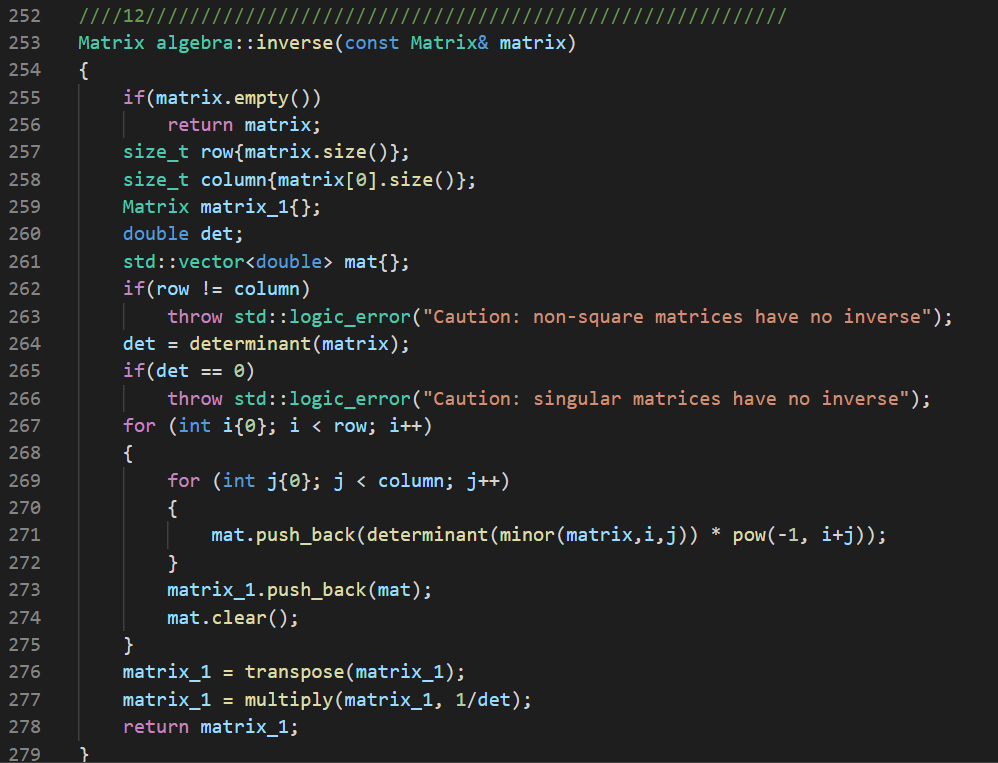
از روش معمولی محاسبه دترمینان استفاده میکنیم به این صورت که یک سطر را درنظر میگیریم و ماینورهای هر درایه را بدست آورده و در آن ضرب میکنیم.(از الگوریتم بازگشتی استفاده میکنیم چون نیاز داریم به مایتور 2\*2 برسیم)



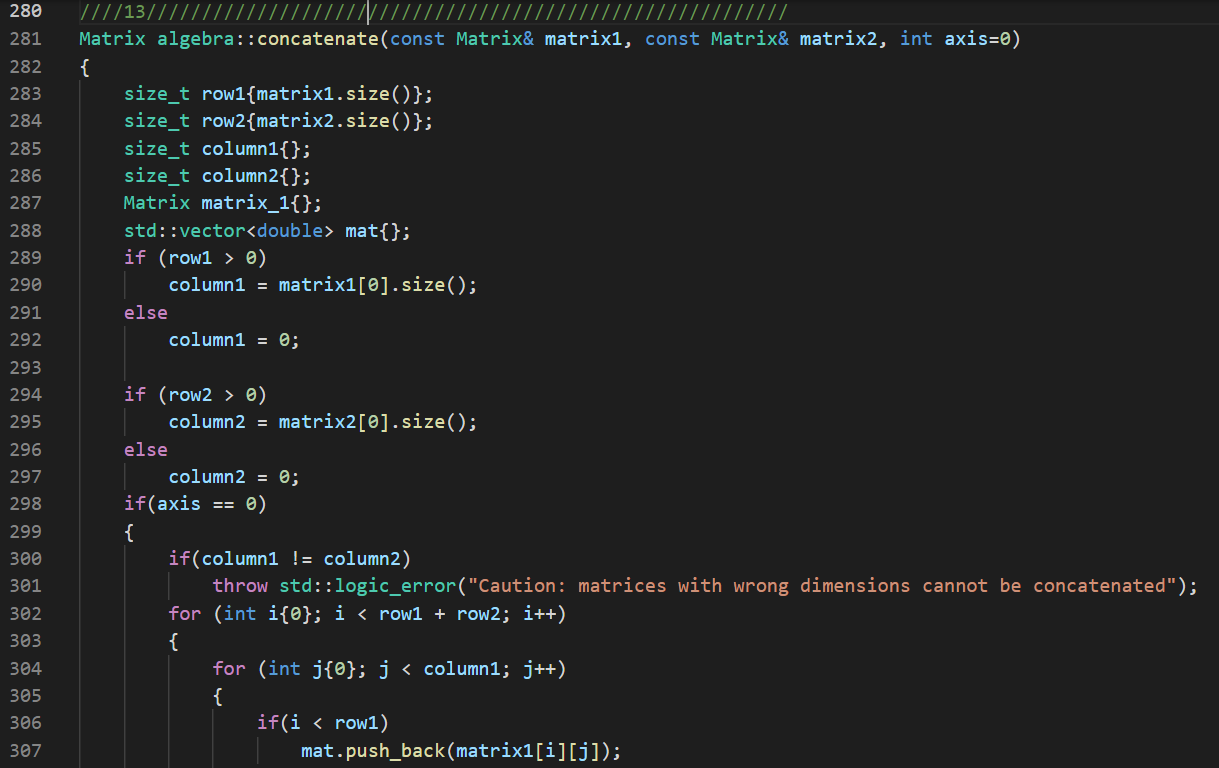
1. Inverse:

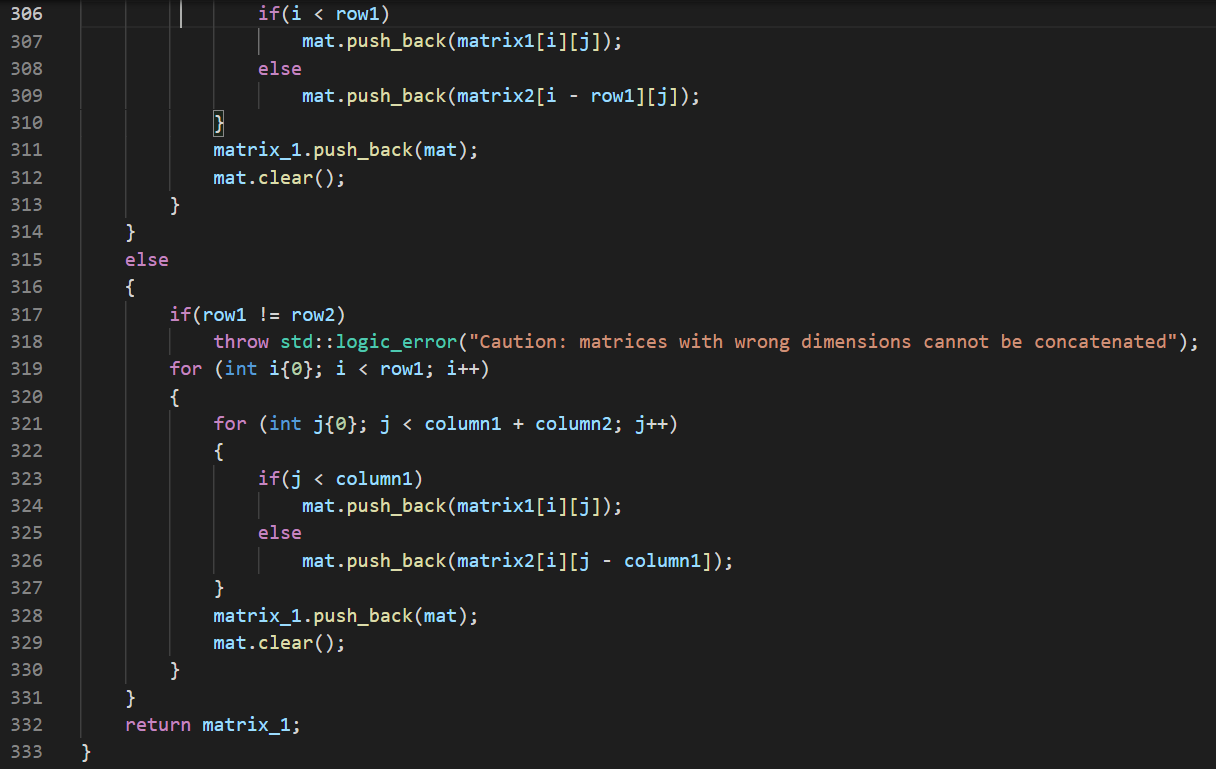
برای بدست آوردن ماتریس معکوس ابتدا نیاز داریم ماتریس کهاد را بدست آوریم و با اعمال مثبت و منفی یکی در میان به ماتریس همسازه میرسیم و با ترانهاده کردن آن به ماتریس الحاقی میرسیم و در معکوس دترمینان ضرب میکنیم و به ماتریس وارون میرسیم.

اما چند نکته مهم وجود دارد که اگر ماتریس تهی باشد، وارون آن هم تهی است و اگر ماتریس مربعی نباشد دیگر وارونی وجود ندارد و اگر دترمینان هم صفر شود باز وارونی وجود ندارد.



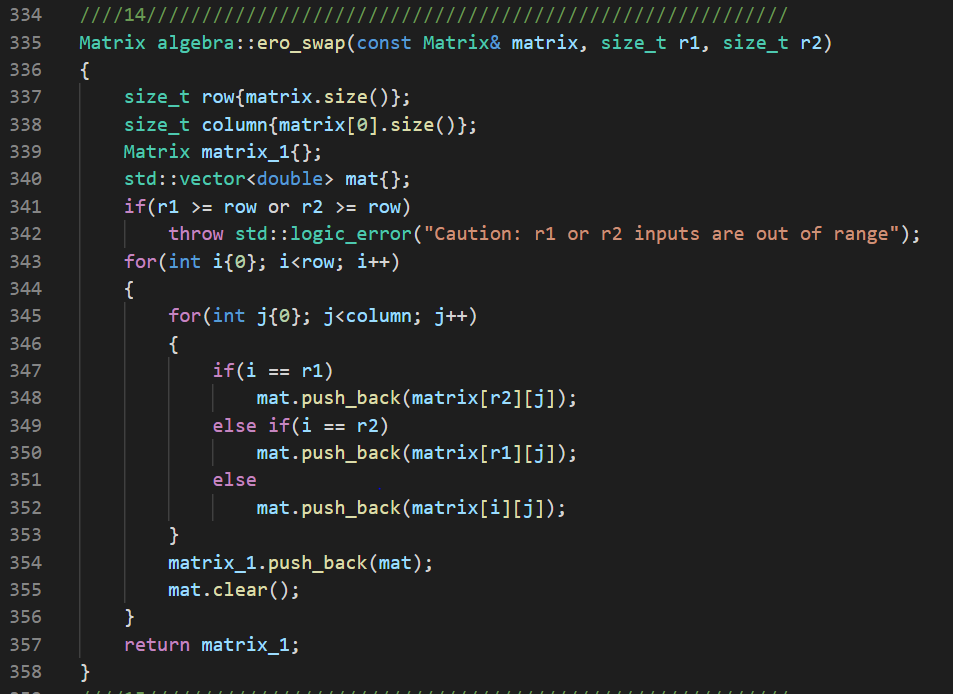
1. Concatenate:

باز هم بررسی میکنیم که ماتریس‌ها تهی نباشند و سپس با توجه به اینکه کاربر axis =0 or 1 وارد کرده است دو ماتریس را به هم وصل میکنیم.

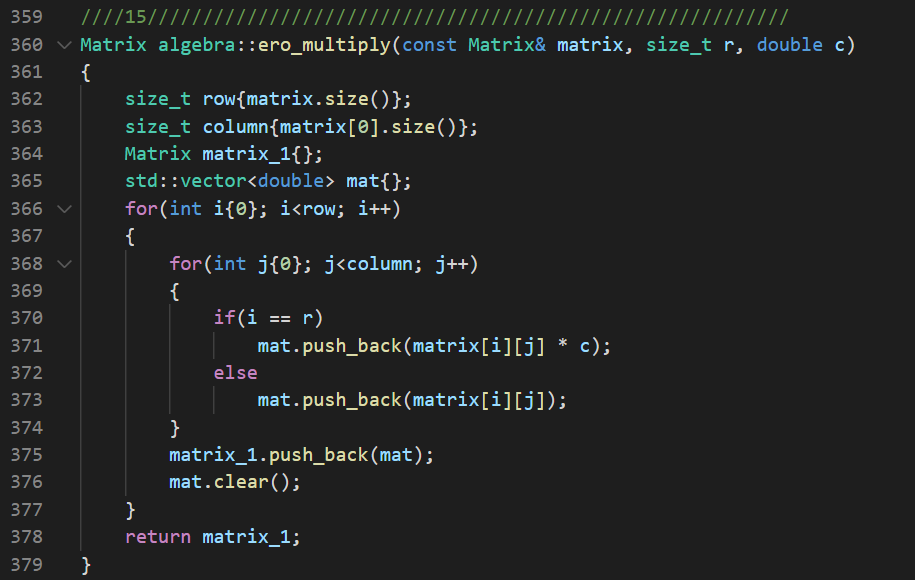
وقتی دو ماتریس را پشت هم وصل کنیم پس تعداد سطرها افزایش میابد و باید ستون‌های دو ماتریس برابر باشند و وقتی دو ماتریس کنار هم وصل کنیم پس تعداد ستون‌ها افزایش میابد و باید سطرهای دو ماتریس با هم برابر باشند تا این عمل اتصال انجام شود.

1. Ero\_swap:

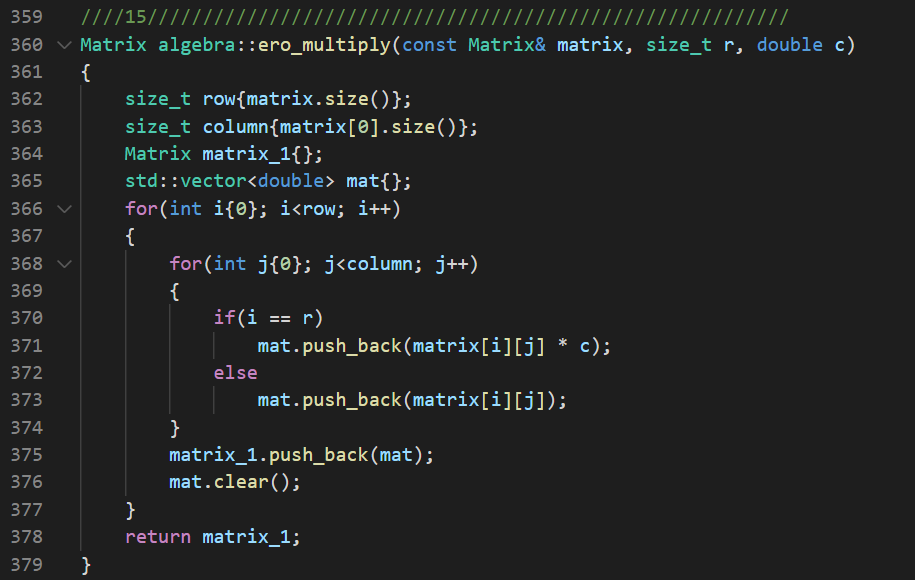
در این تابع دو سطر را با هم جابه‌جا میکنیم فقط باید بررسی شود که عدد سطری که وارد میشود از تعداد سطرهای ماتریس بیشتر نباشد و هنگامی که ماتریس ورودی را در ماتریس دیگری میریزیم اگر به آن سطرهای موردنظر برسیم باید جابه‌جا این عملیات انجام شود.



1. Ero\_multiply:

اگر به سطر موردنظر رسیده باشیم تمام درایه‌های آن سطر را در یک عدد اسکالر ضرب میکنیم.

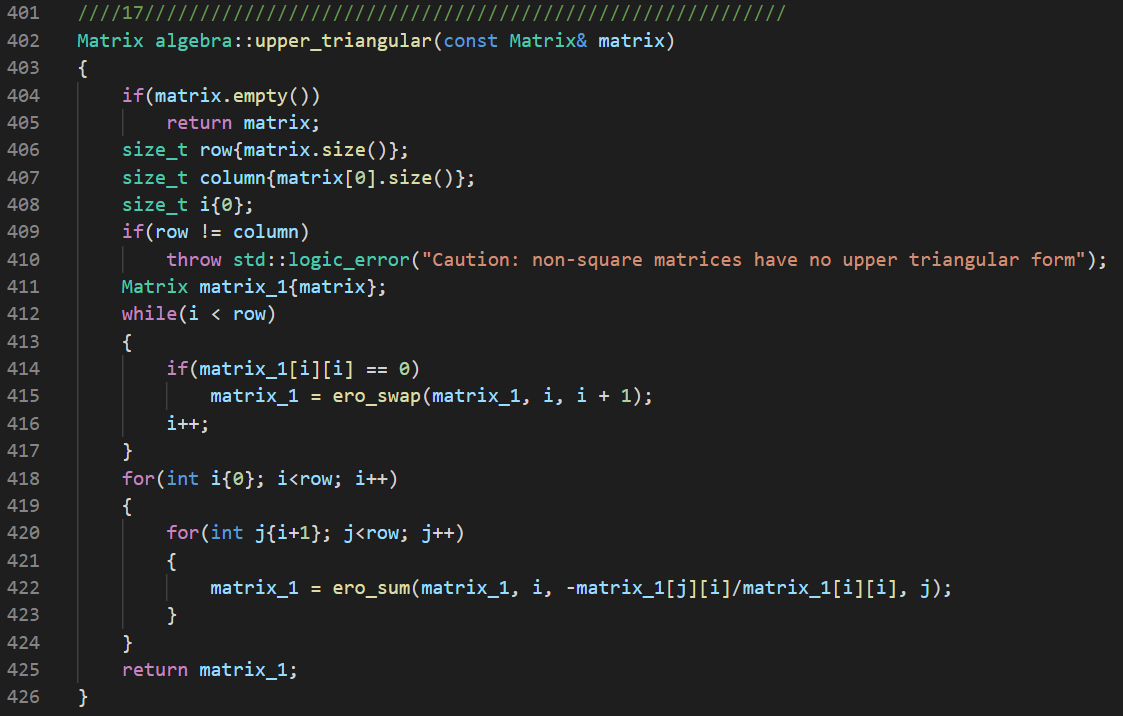
1. Ero\_sum:

یک سطر را در عدد اسکالری ضرب کرده و به سزری دیگر اضافه میکنیم.در کد هم به همین روش انجام دادم و باقی درایه‌ها خودشان قرار میگیرند و فقط اگر به سطر موردنظر برسیم این الگوریتم را پیاده سازی میکنیم.

1. Upper\_triangular:

اگر ماتریس مربعی نباشد نمیتوان ماتریس بالا مثلثی بدست آورد و از تابع ero\_sum استفاده میکنیم تا درایه‌های موردنظر را صفر کنیم. سطر اول را در ضریبی از درایه اول سطر اول و درایه اول سطرهای بعدی ضرب میکنیم و با همان سطر جمع میکنیم.

نکته مهم این است که اگر یکی از درایه‌های روی قطر صفر باشد آنگاه مخرج ضریب صفر میشود و برای این که به این مشکل برخورد نکنیم این سطر را با سطر زیر آن جابه‌جا میکنیم.



با تشکر