# تمرین کامپیوتری شماره 2

## عنوان: Distributed File System Over LAN

**درس:** شبکه‌های کامپیوتری

**استاد:** دکتر ناصر یزدانی

**نیمسال دوم سال تحصیلی 04-1403**

# فهرست مطالب

## - اهداف پروژه

- آشنایی با فایل سیستم‌های توزیع شده (DFS)
- آشنایی با کاربرد شبکه در DFS
- آشنایی با Firewall Punching

## - ابزارهای مورد استفاده

این پروژه **کاملاً با زبان C++** پیاده‌سازی خواهد شد. به همین دلیل، از **کتابخانه‌ها و فریم‌ورک‌های معتبر** این زبان برای توسعه استفاده می‌کنیم تا کدنویسی سریع‌تر، بهینه‌تر و ساختاریافته‌تر باشد.

**ابزارهای مورد نیاز:**

- **Qt Framework** برای پیاده‌سازی تمامی بخش‌های پروژه.

## - مقدمه

**"تمام تخم مرغ هایت را در یک سبد نگذار"**، ضرب‌المثل معروفی که به احتمال زیاد تاکنون بارها آن را شنیده‌اید. در دنیای فناوری این ضرب‌المثل سرلوحه کار بسیاری از شرکت‌های بزرگ و معروف است. هنگامی که داده‌های ما به صورت متمرکز در مکانی جمع شوند، احتمال خرابی و از دست رفتن آن‌ها بسیار زیاد می‌شود. برای همین، شرکت‌هایی که خدمات آن‌ها مبتنی بر ذخیره‌ی داده ها است (مانند گوگل، مایکروسافت، فیسبوک و ...) به سیستم‌های ذخیره و بازیابی توزیع شده روی آورده اند که از معروف ترین این Distributed File System ها یا به اختصار DFS ها می‌توان به GFS و Hadoop اشاره کرد. با این حال، اطمینان بیشتر از سلامت داده‌ها تنها مزیت فایل سیستم‌های توزیع شده نیست، با استفاده از این شبکه‌ها، می‌توان گلوگاه‌ها را شناسایی و حذف کرد، داده‌ها را در نزدیکی کاربران ذخیره کرد تا دسترسی آن‌ها به داده‌ها سریع‌تر شود و همچنین حجم داده در بخش کوچک‌تری از شبکه منتقل شود و ...

## - شرح پروژه

فایل سیستم‌های توزیع شده شامل 2 بخش اصلی هستند:

1- **گره مدیر:** این گره فرا داده‌ها (Meta Data) را ذخیره و تصمیم‌گیری‌ها را انجام می‌دهد. کاربر برای نوشتن/خواندن داده‌ها ابتدا به این گره پیام می‌دهد.
2- **گره داده (chunk server):** این گره‌ها وظیفه‌ی ذخیره‌ی فایل‌ها را بر عهده دارند. هر فایل ممکن است به چندین بخش (chunk) شکسته شود و هر چانک درون یک چانک سرور متفاوت ذخیره شود. (در ادامه برای اشاره کردن به این گره، به اختصار به آن سرور می‌گوییم.)

در کنار فایل سیستم، ما به یک کاربر جهت ذخیره/بازیابی داده نیاز داریم.

## - فرآیند کلی ذخیره فایل در یک فایل سیستم توزیع شده

- کاربر به گره مدیر پیامی شامل جزئیاتی درباره کاری که می‌خواهد انجام دهد، می‌فرستد. در ابتدا فرض می‌کنیم که کاربر می‌خواهد داده‌ای (فایلی مانند یک txt) را در این فایل سیستم ذخیره کند، ابتدا یک پیام شامل حجم فایل و نوع فایل و ... (یک سری meta data) برای مدیر ارسال می‌کند؛ سپس مدیر محاسبه می‌کند که با توجه به حجم داده ، چند چانک سرور برای ذخیره‌ی این فایل مورد نیاز است (فرض می‌کنیم که حجم چانک ها ثابت است و مدیر از آن اطلاع دارد.)
- در ادامه مدیر مطابق با الگوریتمی مشخص (مراجعه به بخش توپولوژی و الگوریتم)، آدرس سروری که اولین چانک از داده باید در آن قرار بگیرد را به کاربر بازمی‌گرداند. در کنار این موارد، به کاربر می‌گوید که هر فایل را به چند قسمت تقسیم کند.
- کاربر ابتدا فایل را به تعداد بخش‌هایی که مدیر اطلاع داده تقسیم می‌کند و سپس شروع به ارسال بخش اول به سرور اول میکند. پس از اتمام ارسال چانک اول، سرور اول به کاربر می‌گوید که چانک بعدی چه کسی است (آدرس چانک سرور بعدی را ارسال می‌کند) همچنین این آدرس را در انتهای چانکی که دریافت کرده ذخیره می‌کند.
- کاربر به همین ترتیب بخش‌های دیگر فایل را به چانک سرورها ارسال می‌کند تا این فرآیند تمام شود.
- لازم به ذکر است که مدیر شبکه، اطلاعات فایل به همراه آدرس سرور اول که ابتدای فایل در آن قرار دارد را نگه می‌دارد تا در هنگام بازیابی اطلاعات، از آن‌ها استفاده کند.

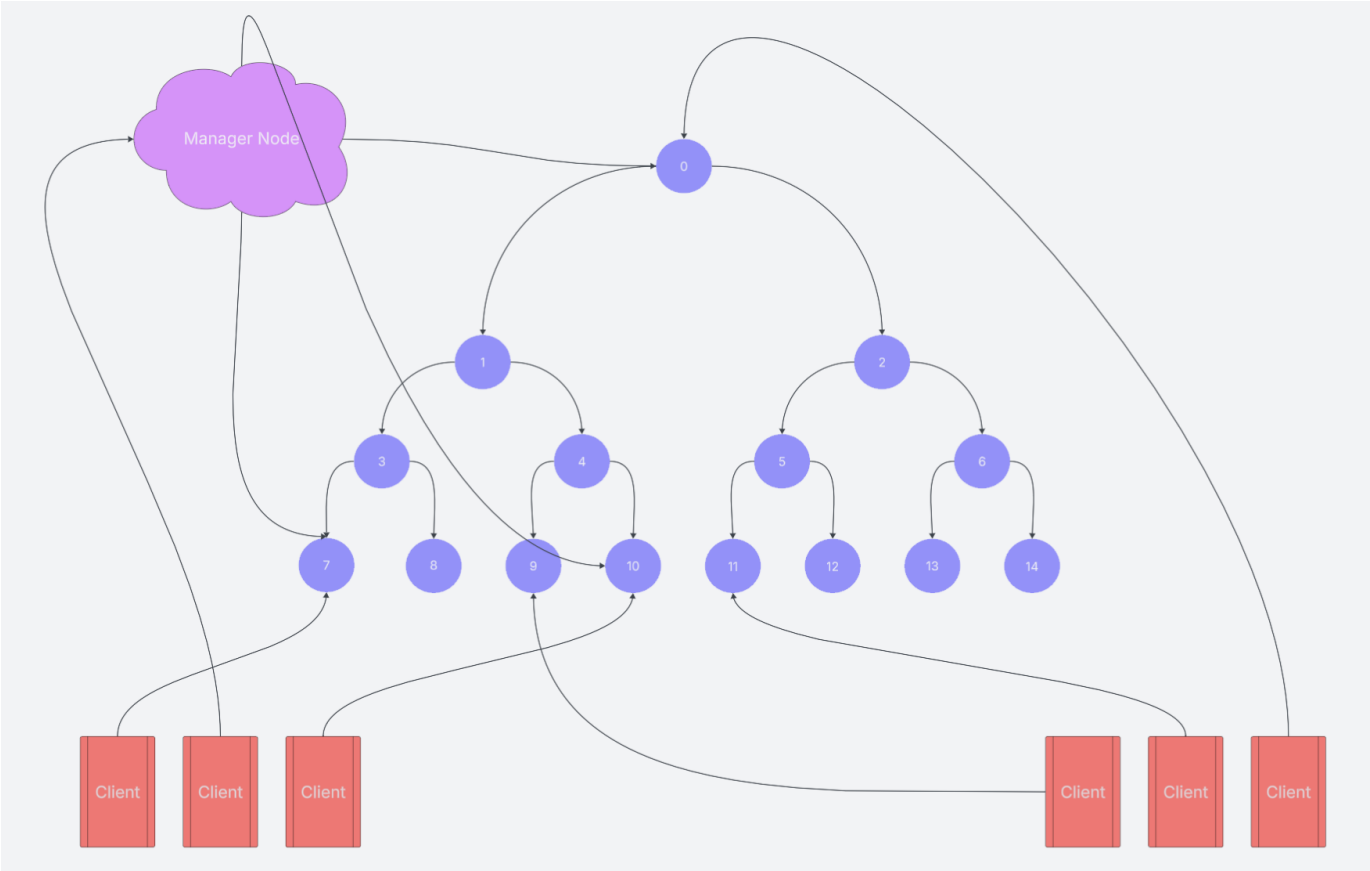
## - فرآیند کلی بازیابی/خواندن فایل از یک فایل سیستم توزیع شده

- کاربر اطلاعات فایلی که می‌خواهد بخواند را به مدیر شبکه ارسال می‌کند.
- مدیر شبکه با بررسی اطلاعات، آدرس چانک سرور اول را به کاربر اطلاع می‌دهد.
- کاربر به چانک سرور اول درخواست داده و چانک اول داده را دریافت می‌کند.
- در انتهای بخش اول داده، آدرس چانک سرور بعدی وجود دارد. که ادامه ی داده در آن است را به کاربر اطلاع می‌دهد و به همین شکل کاربر تا دریافت کل فایل فرآیند دریافت چانک را تکرار می‌کند.

## - توپولوژی و الگوریتم

فایل سیستم های توزیع شده، از توپولوژی‌های متفاوتی برای ارتباط میان چانک سرورها استفاده می‌کنند، همچنین الگوریتم‌های خاصی را برای پیدا کردن نود بعدی برای دریافت بخش بعدی داده ، استفاده می‌کنند. در این تمرین، طراحی ما به شکل زیر است:

- در ابتدا لازم به ذکر است که سعی کنید حداقل از 2 لپ‌تاب (یا 2 ماشین مجازی) برای پیاده‌سازی تمرین استفاده کنید.
- در لپ تاپ اول، کلاینت (کاربر) و منیجر (گره مدیر) حضور دارند.
- در لپ تاپ دوم، **15 چانک سرور به شکل یک درخت دو دویی** به یکدیگر متصل می‌شوند.
- لازم نیست میان چانک سرورها ارتباط واقعی برقرار باشد. (چانک ها به یک دیگر ارتباط مستقیم ندارند)




*تصویر 1، توپولوژی شبکه*

- دقت شود که هر چانک سرور به یک آدرس در فایل سیستم دستگاه مپ (Map) می‌شود و داده‌های خود را در آن پوشه (Folder) ذخیره می‌نماید. به عنوان مثال، چانک سرور اول به آدرس D:/CHUNK-1 و چانک سرور دوم به ادرس D:/CHUNK-2 مپ می‌شوند.
- الگوریتم پیدا کردن نود بعدی، الگوریتم DFS (Depth-first search) خواهد بود. (دقت شود که شماره گذاری نودها دقیقا به صورت تصویر بالا باشد، اسم شماره گذاری BFS (Breadth-First Search) است اما الگوریتم شما باید DFS کار کند)
- پورت های چانک سرورها را طوری تنظیم کنید که مطابق با شماره‌ی نود در درخت باشد.
- هر دو لپ تاپ روی یک شبکه‌ی محلی حضور داشته باشند.
- داده‌ها، فایل های نوشتاری هستند که در اختیار شما قرار می‌گیرند.
- با توجه به قسمت "افزودن نویز به داده های ارسالی" و نویز 1 درصد، پیشنهاد می شود که داده‌ها را در چانک‌های 8 **کیلوبایتی** ذخیره کنید و طبق آن مقدار fec_length را بدست آورید.
- طراحی Metadata و داده‌هایی که هر گره نگه می‌دارد، بر عهده‌ی شماست.
- توجه کنید که ممکن است هنگام تقسیم کردن داده، حجم داده کمتر از مقدار مشخص شده شود و در آخر چانک، شما باید آدرس سرور را ذخیره کنید. برای انجام این کار، می توانید، داده را در سرور wrap کنید.
- توجه کنید که ارتباطات بین گره ها در **تصویر 1**، به عنوان مثال آورده شده‌اند و تنها مسیری احتمالی از داده را نشان می‌دهند. علاوه بر آن، تنها یک client داریم.

## - افزودن نویز به داده های ارسالی

در دنیای واقعی، همواره نویز و احتمال خرابی داده‌ها وجود دارد. ممکن است در هنگام انتقال، داده‌ها خراب شوند، یا در هنگام نگهداری به دلایلی بخش‌هایی از داده‌ها آسیب ببینند و از دست بروند.

- به منظور شبیه سازی فرایند انتقال داده در دنیای واقعی، شما باید به صورت مصنوعی نویز را به داده اضافه کنید. از جایی که تولید نویز موضوع اصلی این تمرین نیست، جهت تولید نویز کافی است عددی تصادفی (Random) تولید کرده و طبق آن با احتمال p هر بیت را معکوس کنید. برای راحتی کار می‌توانید یک رشته از اعداد صفر و یک تولید کرده و آن را با دیتای اصلی XOR کنید. توجه داشته باشید که نویز تولید شده نباید بیشتر از مقدار مشخصی باشد تا تمام داده‌ها غیر قابل بازیابی نباشند و خراب بودن بسته‌هایی که فرستاده می‌شوند موردی استثنایی محسوب شود. (این مقادیر مشخص، در ادامه توضیح داده خواهند شد.) این نکته را در نظر داشته باشید که اضافه‌کردن نویز، قدم آخر قبل از ارسال اطلاعات است و الگوریتم بازیابی باید قبل از آن پیاده شده باشد.
- برای بازیابی اطلاعات از دست رفته هنگام انتقال داده، فرستنده و گیرنده باید با الگوریتمی داده ها را رمزگذاری (Encode) رمزگشایی (Decode) کنند. در برخی موارد ممکن است نویز به اندازه‌ای باشد که بازیابی اطلاعات ممکن نباشد، در این صورت گیرنده پیامی به فرستنده ارسال می‌کند تا بسته‌های مذکور را دوباره دریافت کند. در این تمرین اتصال TCP مورد بحث نیست، در نتیجه نیازی برای دریافت دوباره پیام در صورت خراب بودن اطلاعات نیست و کافی است این پدیده را تشخیص داده و پیامی بر مبنای Corrupt بودن این قسمت از پیام نمایش دهید.
- الگوریتم reed-solomon که در این تمرین استفاده خواهید کرد، به منظور تصحیح خطا (noise correction) در ذخیره سازی داده های کلان استفاده می‌شود. خلاصه ای از روش کار این الگوریتم این است که تعداد بیت parity به عنوان داده های اضافی به پیام اضافه می‌شوند. این اطلاعات اضافی بر مبنای تئوری میدان‌ها کار می‌کنند که در درس مبانی امنیت شبکه با آن آشنا خواهید شد. الگوریتم مذکور، به مقدار t بیت اضافی به پیام اضافه می‌کند و با استفاده از این اطلاعات، می‌تواند تا حداکثر t خطا را تشخیص دهد و t/2 خطا را تصحیح کند. باید توجه داشته باشید که نویز تولید شده بیشتر از t نباشد، در غیر این صورت تشخیص خطا غیرممکن خواهد بود.
- برای پیاده سازی الگوریتم reed-solomon می‌توانید از کتابخانه ی موجود در این لینک استفاده کنید. توضیحات مربوط به نحوه کارکرد الگوریتم در صفحه گیت‌هاب به همراه مثال آمده است. توجه داشته باشید که پارامتر t با نام fec_length نشان داده شده است.
- در پیاده سازی این قسمت، ترتیب عملیات لازم به این صورت است که ماژول انکودر کتابخانه را ابتدا با توجه مقدار داده ای که هر بار قبل از فرستادن انکود می‌کنید و مقدار احتمال خرابی هر بیت هنگام انداختن نویز، تنظیم کنید و مقدار t را به اندازه ای که به نظرتان کافی است قرار دهید. سپس داده را با یک تابع نویز و احتمال خرابی که انتخاب کردید XOR کنید (مقدار واقعی نویز معمولا بین $10^{-5}$ و $10^{-6}$ است اما برای مشاهده اثر نویز در این تمرین حدود **یک درصد نویز** بر روی داده اعمال کنید). سپس ارتباط بین دو دستگاه را برقرار کرده و در مقصد پیام را دیکود کنید. در بخش دیکود کردن پیام مناسبی برای نمایش تعداد بیت های خراب و اینکه آیا الگوریتم توانسته آن را بازیابی کند، نمایش دهید و اطلاعات را در جای صحیح ذخیره کنید. در صورت خراب بودن اطلاعات (همانطور که ذکر شد، باید پدیده استثنایی باشد)، آن چانک را به عنوان چانک خراب نشانه گذاری کنید و هنگام درخواست آن بخش توسط گیرنده پیام chunk corrupt را مخابره کنید.
- توجه کنید که در فرآیند ارسال و ذخیره سازی، حداقل 1 چانک خراب باید تولید شود!

### - ارتباط P2P میان دو دستگاه

با توجه به این که در این تمرین باید از دو لپ‌تاب استفاده کنید، می‌خواهیم با یکی از موانع ایجاد ارتباط P2P و راه حل رفع آن آشنا شویم. دیوار آتش یا فایروال، ابزاری برای ارتقای سطح امنیت در دستگاه‌های متصل به شبکه است. یکی از کاربردهای فایروال، جلوگیری از ورود بسته‌های داده در بستر شبکه به سیستم کاربر است بدین شکل که فقط در صورتی بسته حق ورود از سمت شبکه به دستگاه را دارد که کاربر درخواستی ارسال کرده باشد و این بسته پاسخ به آن درخواست باشد!

به همین جهت است که برای ارتباط با سرور، ابتدا کاربر درخواست می‌دهد و هنگامی که پاسخ درخواست به فایروال می‌رسد، با توجه به اینکه این بسته یک پاسخ به درخواست کاربر است، فایروال اجازه‌ی عبور بسته را می‌دهد، در غیر این صورت بسته اجازه ورود نخواهد داشت. با توجه به این موضوع، ارتباط P2P در هنگامی که فایروال سیستم روشن است، بنظر غیر ممکن می‌رسد. به مثال زیر توجه کنید:

کاربر A به کاربر B داده ای ارسال می‌کند. فایروال در سیستم کاربر B این بسته را دریافت کرده و بررسی می‌کند که آیا این یک پاسخ به درخواستی از سمت کاربر B هست یا خیر. چون این شرایط برقرار نیست، بسته اجازه ی ورود به سیستم کاربر B را نخواهد داشت. همین مساله از سمت کاربر A نیز وجود خواهد داشت و در نتیجه می‌توان گفت فایروال اجازه ی برقراری ارتباط P2P را نمی‌دهد.

با این حال ما می‌دانیم که ارتباط P2P در دنیای واقعی وجود دارد، چگونه؟

یکی از راه های موجود استفاده از عملیات فایروال پانچینگ (Firewall Punching) است. در این روش میتوان بر بستر TCP و UDP اقدام به ایجاد ارتباط P2P کرد، در این تمرین می‌خواهیم تا با استفاده از UDP عملیات فایروال پانچینگ را انجام دهید تا دو لپ‌تاب بتوانند در صورت روشن بودن فایروال، اقدام به ارسال و دریافت داده از کاربر به چانک سرورها و برعکس نمایند.

برای اطلاعات بیشتر در مورد نحوه کارکرد firewall punching یا hole punching می‌توانید از لینکهای زیر استفاده کنید:

- لینک 1
- لینک 2

## - سوالات تئوری

در گزارش خود به سوالات زیر پاسخ دهید:

1- نویزها و خرابی‌های ایجاد شده در زمان انتقال چگونه در شبکه رفع می‌شوند؟
2- پلتفرم‌های استریم ویدیو به دلیل جنس کاربرد، از ارتباط بر بستر UDP استفاده می‌کنند. آن‌ها این خرابی‌ها را چگونه مدیریت می‌کنند؟
3- نویزها و خرابی‌های زمان استقرار (اطلاعات در جایی ذخیره شده‌اند اما ممکن است خراب شوند) چگونه رفع می‌شوند و فایل سیستم های توزیع شده چه مزیتی در این مورد دارند؟
4- با توجه به پارامتر هایی که برای نویز و دینویز اطلاعات ذخیره کردید، به صورت میانگین چند درصد از اطلاعات فرستاده شده غیر قابل بازیابی خواهند بود؟ چند درصد از خطاها غیر قابل تشخیص است؟
5- آیا فایل سیستم های توزیع شده باعث جلوگیری از هر نوع خرابی در داده‌ها می‌شوند؟ کاربرد رپلیکاها (Replica) را در این سیستم‌ها توضیح دهید.
6- چگونه می‌توان با استفاده از فایل سیستم های توزیع شده گلوگاه‌ها را حذف کرد؟
7- CDN ها (Content Delivery Network) یا شبکه های تحویل محتوا را با فایل سیستم های توزیع شده مقایسه کنید. این شبکه ها چگونه می‌توانند مدت زمان دسترسی کاربر به محتوا را کاهش دهند و فایل سیستم های توزیع شده چه نقشی در این بین دارند؟

**توجه:** این سؤالات برای **بررسی میزان درک شما از مفاهیم اصلی پروژه طراحی شده‌اند**. برای هر موضوع زمان کافی برای تحقیق و بررسی اختصاص دهید. پاسخ‌های شما بر اساس دقت، و وضوح بیان ارزیابی خواهند شد.

## - جمع بندی و نکات پایانی

### معیارهای ارزیابی

- ✓ **صحت عملکرد:** ارسال و دریافت فایل به درستی و منطبق بر انتظارات انجام شوند.
- ✓ **پیاده‌سازی نویز:** فرآیند ایجاد و رفع نویز به همراه مواردی که قادر به حذف آن نیستیم به درستی پیاده شده باشد.
- ✓ **پیاده‌سازی فایروال پانچینگ:** در هنگام روشن بودن فایروال هر دو دستگاه، ارتباط به درستی برقرار شود.
- ✓ **اجرای صحیح در شبکه محلی:** اجرای پروژه بر روی دو لپ‌تاب و ذخیره فایل از لپ‌تابی در لپ‌تابی دیگر.
- ✓ **نام گذاری صحیح فایل ها و فولدر ها:** چانک سرورها باید فولدرها و فایل‌ها را با دقت نام‌گذاری کنند.
- ✓ **دقت در پاسخ‌های نظری:** پاسخ‌ها به سؤالات تئوری باید صحیح، دقیق و همراه با تحلیل مناسب باشند.

## مهلت تحویل: 1404/02/25

- تمرین‌های کامپیوتری به صورت انفرادی یا در گروه‌های 2 نفره انجام می‌شوند. (حضور تمام اعضای گروه در جلسه تحویل الزامی است)
- هر ۲ نفر باید می بایست کارها را تقسیم کنند. همچنین از Git برای ساختن branch و تقسیم issue ها استفاده نمایید. (با استفاده از commit ها و تعیین issue ها میزان مشارکت هر نفر مشخص می‌شود). بعد از انجام این کار کدها را در یک repository به نام CN_CA_2 در اکانت‌های GitHub خود قرار دهید (به صورت private). همچنین در یک فایل README.md می‌توانید report و داکیومنت خود را کامل کنید و در کنار repository قرار دهید.
- در نهایت لینک این repository و آدرس هش آخرین کامیت را در قالب یک فایل txt در محل پاسخ تمرین قرار دهید. (از فرستادن فایل به صورت زیپ جدا خودداری نمایید.) اکانت دستیاران این تمرین را به Repo خودتان به عنوان Maintainer به پروژه اضافه کنید.

**Github accounts:**

- @ArianFiroozi
- @baharvand79
- @Armi-B
- @Ali-Abedini-2001
- @TheSohrabX

- برای پیاده سازی این تمرین از C++ و Qt Framework استفاده کنید.
- دقت کنید گزارش نهایی شما باید می‌بایست مانند یک Document باشد و شامل توضیح کد و ساختار کد، همچنین نتیجه نهایی اجرای کد و اسکرین شات‌های دقیق از تمام مراحل باشد. (در فایل Readme.md کنار فایل های اصلی خود و در Repo مربوطه قرار دهید.) این نکته حائز اهمیت است که فایل PDF به هیچ عنوان مورد پذیرش قرار نخواهد گرفت ولی لازم است لینک Repository خود را در جایگاه تعریف شده برای این تمرین در ایلرن قرار دهید.
- **ساختار صحیح و تمیزی کد برنامه**، بخشی از نمره‌ی این پروژه شما خواهد بود. بنابراین در طراحی ساختار برنامه دقت کنید.

- برای هر قسمت کد، گزارش دقیق و شفاف بنویسید. کدهای ضمیمه شده بدون گزارش مربوط به آن، نمره‌ای نخواهند داشت.
- هدف این تمرین یادگیری شماست. لطفا تمرین را خودتان انجام دهید. در صورت مشاهده‌ی شباهت بین کدهای دو گروه، مطابقت سیاست درس با گروه متقلب و تقلب دهنده برخورد خواهد شد. همچنین توجه داشته باشید استفاده از ابزارهای AI به هیچ عنوان توجیهی برای شباهت کدهای تحویل داده شده توسط گروه های مختلف نمی باشد بنابراین از این ابزارها صرفا برای یادگیری، درک بهتر تمرین و اصلاح کدهایی که کاملا توسط خودتان نوشته شده است استفاده کنید.
- برای آگاهی از قوانین ارسال با تاخیر، گریس و پرسش سوالات خود در رابطه با این تمرین، به صفحه Piazza درس مراجعه نمایید.

موفق باشید.