تمرین کامپیوتری شماره 1

عنوان: Socket Programming

درس: شبکه‌های کامپیوتری

استاد راهنما: دکتر ناصر یزدانی

نیمسال دوم سال تحصیلی 04-1403

فهرست مطالب

اهداف پروژه 1

ابزارهای مورد استفاده 1

مقدمه 1

شرح بازی 1

ویژگی‌های بازی 2

راهنمای اجرای پروژه 2

نوشتن برنامه 2

راهنمای پیاده‌سازی 3

- ❖ ایجاد کلاس‌های اختصاصی برای شبکه 3
- ❖ سریال‌سازی داده‌ها و مدیریت اشیای بازی 3
- ❖ همگام‌سازی و مدیریت رویدادها 3
- ❖ مدیریت حرکت بازیکنان 4
- ❖ جداسازی منطق شبکه و بازی 4

سوالات تئوری 4

نصب فریمورک Qt 4

جمع بندی و نکات پایانی 7

معیارهای ارزیابی 7

- اهداف پروژه

- یادگیری مبانی برنامه‌نویسی سوکت
- پیاده‌سازی و مقایسه پروتکل‌های UDP و TCP در یک برنامه کاربردی
- همگام‌سازی وضعیت بازی در شبکه برای ایجاد تجربه چندنفره
- درک تفاوت‌های نظری بین پروتکل‌های شبکه و کاربرد آن‌ها در بازی‌ها

- 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 سریع، بهینه و ساختاریافته باشد.

ابزارهای مورد نیاز:

- **Qt Framework** برای طراحی رابط گرافیکی (GUI) و مدیریت رویدادها

- مقدمه

در عصر حاضر، ارتباطات شبکه‌ها و تبادل داده‌ها بین دستگاه‌های مختلف به یکی از ارکان اصلی توسعه نرم‌افزار تبدیل شده است. برنامه‌نویسی سوکت (Socket Programming) به عنوان یکی از پایه‌های اساسی در این حوزه، امکان برقراری ارتباط بین دو یا چند دستگاه را در یک شبکه فراهم می‌کند. این تکنولوژی در برنامه‌های کاربردی مختلفی از جمله بازی‌های آنلاین، چت‌های تحت شبکه، انتقال فایل و بسیاری از برنامه‌های دیگر مورد استفاده قرار می‌گیرد.

در این پروژه، هدف اصلی ما توسعه یک بازی ساده با استفاده از زبان برنامه‌نویسی C++ و فریمورک Qt است که قابلیت اجرای دو نفره را از طریق شبکه فراهم کند. برای رسیدن به این هدف، از دو پروتکل شبکه، یعنی TCP (Transmission Control Protocol) و UDP (User Datagram Protocol) استفاده خواهیم کرد. هر یک از این پروتکل‌ها ویژگی‌ها و کاربردهای خاص خود را دارند که در ادامه به بررسی آن‌ها خواهیم پرداخت.

- شرح بازی

بازی طراحی‌شده نسخه‌ای ساده‌شده از Bomberman است که در آن بازیکنان شخصیت‌هایی را کنترل می‌کنند، بمب‌هایی برای تخریب موانع قرار می‌دهند و با یکدیگر رقابت می‌کنند. کد ارائه‌شده نسخه‌ای دوبعدی و مینیمال از این بازی است که با استفاده از C++ و Qt6 توسعه یافته و شامل دو بازیکن در یک صفحه نمایش واحد می‌باشد.

- ویژگی‌های بازی

1. **جان بازیکنان:** هر بازیکن دارای ۳ جان است.
2. **حرکت بازیکنان:** بازیکنان می‌توانند با کلیدهای W، S، D و A حرکت کنند.
3. **قرار دادن بمب:** هر بازیکن می‌تواند با فشردن کلید SPACE یک بمب در محل خود قرار دهد.
4. **مکانیزم انفجار:** بمب‌ها پس از مدتی منفجر شده و اگر بازیکنی در محدوده انفجار باشد، یک جان خود را از دست می‌دهد.
5. **شرط پیروزی:** بازی زمانی به پایان می‌رسد که جان یکی از بازیکنان به صفر برسد.

- راهنمای اجرای پروژه

پس از نصب Qt (به بخش نصب Qt مراجعه کنید)، پروژه را در Qt Creator باز کنید یا از خط فرمان (command line) برای ساخت آن استفاده کنید. سپس فایل اجرایی را راه‌اندازی کنید تا نسخه محلی (local) بازی را مشاهده کنید.

توجه: کد ارائه‌شده در حال حاضر از اجرای چندنفره تحت شبکه پشتیبانی نمی‌کند و مسئولیت شما اضافه کردن این قابلیت است.

- نوشتن برنامه

وظیفه شما در این پروژه، افزودن قابلیت اجرای چندنفره از طریق شبکه به کدی است که در اختیار شما قرار داده شده است. برای این منظور، لازم است که ارتباط میان بازیکنان را یک‌بار با استفاده از پروتکل **UDP** و یک‌بار با پروتکل **TCP** پیاده‌سازی کنید. این پیاده‌سازی باید با استفاده از کتابخانه **QSockets** در فریم‌ورک Qt انجام شود.

همچنین، بخش شبکه (Networking) باید در یک **thread مجزا** از منطق اصلی بازی پیاده‌سازی شود تا از تأثیرگذاری عملیات شبکه بر عملکرد بازی (مانند تأخیر یا فریز شدن) جلوگیری شود و تجربه کاربری بهتری فراهم گردد.

در این پیاده‌سازی، باید دو نمونه از برنامه را اجرا کنید:

- یک نمونه به عنوان **بازیکن اول**
- نمونه دیگر به عنوان **بازیکن دوم**

سپس، باید **ارتباط بین این دو بازیکن را از طریق پروتکل‌های تعیین‌شده برقرار کنید.**

نکات مهم:

1. در روش **TCP**، اتصال باید **پایدار و بدون از دست رفتن داده‌ها** باشد.
2. در روش **UDP**، ارسال و دریافت داده‌ها باید **با در نظر گرفتن احتمال از دست رفتن بسته‌ها** مدیریت شود.

3. برنامه باید به گونه‌ای طراحی شود که هر بازیکن بتواند **داده‌های ارسال‌شده توسط بازیکن دیگر را دریافت کرده و به‌درستی پردازش کند.**
4. برای تست عملکرد، می‌توانید برنامه را **روی یک سیستم با دو نمونه‌ی جداگانه** اجرا کنید یا **از دو دستگاه مختلف در یک شبکه استفاده کنید.**

در نهایت، اطمینان حاصل کنید که هر دو روش به درستی اجرا شده و ارتباط میان بازیکنان **پایدار و بدون خطا** برقرار می‌شود.

- راهنمای پیاده‌سازی

❖ ایجاد کلاس‌های اختصاصی برای شبکه

- TCPManager:
  - مدیریت ارتباطات مطمئن (اتصال، قطع ارتباط و خطاها)
  - مدیریت رویدادهای حیاتی بازی (ثبت بازیکنان، قرار دادن بمب و مرگ بازیکنان)
  - استفاده از QTcpSocket برای ارتباطات کلاینت‌ها و QTcpServer برای میزبانی سرور
- UDPManager:
  - مدیریت به‌روزرسانی‌های لحظه‌ای (موقعیت بازیکنان و وضعیت حرکت)
  - استفاده از QUdpSocket برای ارتباطات سریع و بدون اتصال

❖ سریال‌سازی داده‌ها و مدیریت اشیای بازی

- **فرمت‌های داده:** تبدیل وضعیت بازی (موقعیت بازیکنان، سلامتی و بمب‌ها) به فرمت‌های ساخت‌یافته مانند JSON یا QByteArray
- استفاده از MapLoader برای بارگذاری و مدیریت نقشه‌های بازی

❖ همگام‌سازی و مدیریت رویدادها

- ارسال رویدادهای ورود و خروج بازیکنان
- انتقال اطلاعات حیاتی مانند قرار دادن بمب، تغییر سلامتی یا مرگ بازیکنان
- نمایش وضعیت سلامتی بازیکنان و تخریب دیوارها باید همیشه با سرور همگام باشد
- پخش اطلاعات ورودی حرکت بازیکنان فشردن کلیدهای (W/A/S/D)

❖ مدیریت حرکت بازیکنان

- استفاده از کلاس Player برای مدیریت منطق حرکت و کنترل‌های مرتبط
- افزودن مکانیزم‌های اصلاح وضعیت (Reconciliation) برای همگام‌سازی وضعیت بازی در صورت از دست رفتن بسته‌ها
- پیاده‌سازی پیش‌بینی حرکت برای کاهش تأخیر

❖ جداسازی منطق شبکه و بازی

- از سیگنال‌ها (Signals) و اسلات‌های Qt (Slots) برای جداسازی منطق شبکه از اشیای بازی استفاده کنید.

- سوالات تئوری

به سوالات زیر پاسخ دهید:

1 - سوکت چیست و چه نقشی در برقراری ارتباط شبکه‌ای بین دستگاه‌ها ایفا می‌کند؟

2 - پروتکل‌های TCP و UDP از نظر مدیریت اتصال و تضمین تحویل داده چه تفاوت‌هایی دارند؟

3 - در چه شرایطی TCP گزینه مناسب‌تری نسبت به UDP است و برعکس، در چه شرایطی UDP برتری دارد؟

4 - چرا UDP یک انتخاب رایج برای بازی‌های چندنفره بلادرنگ مانند Bomberman است؟

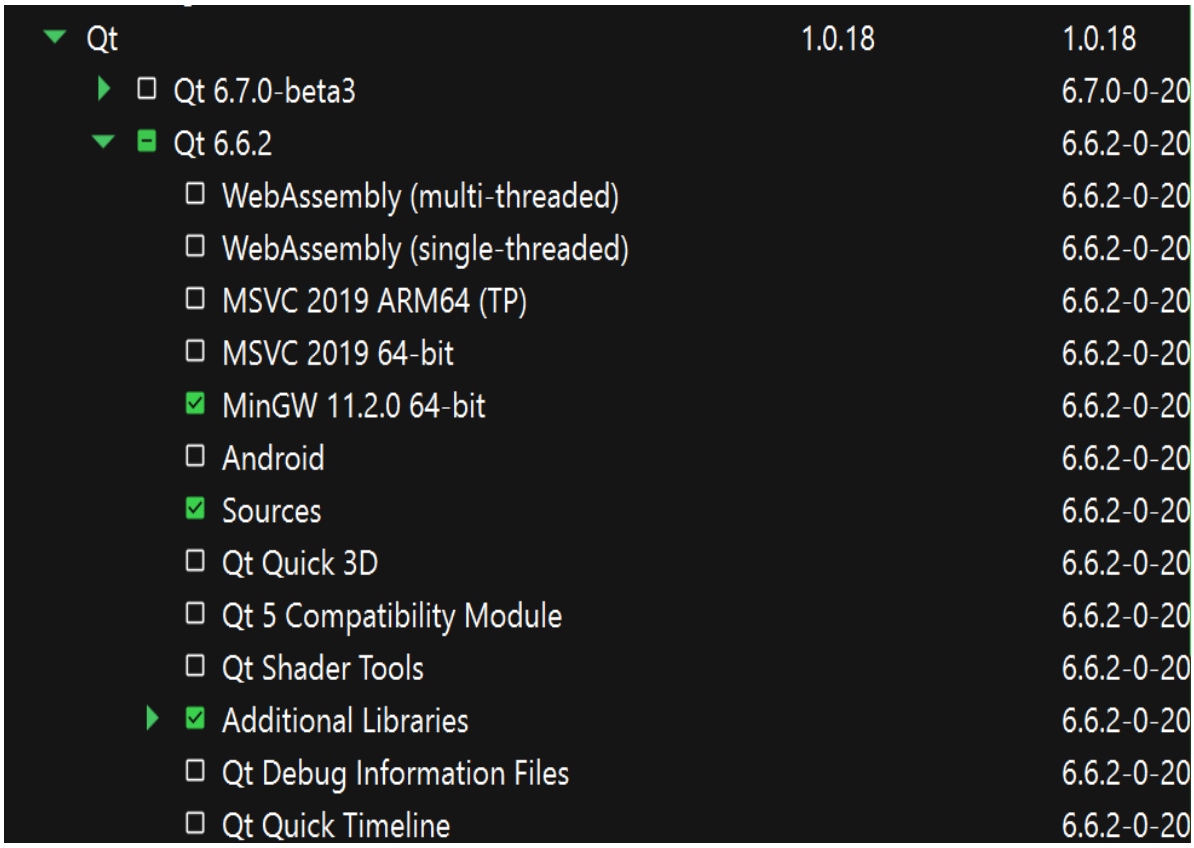
توجه: این سؤالات برای **بررسی میزان درک شما از مفاهیم اصلی پروژه طراحی شده‌اند**. برای هر موضوع زمان کافی برای تحقیق و بررسی اختصاص دهید. پاسخ‌های شما بر اساس دقت و وضوح بیان ارزیابی خواهند شد؛ همچنین لازم است برای هر سوال مثالی بر اساس مراحل انجام تمرین توسط خودتان ذکر شود.

- نصب فریمورک Qt

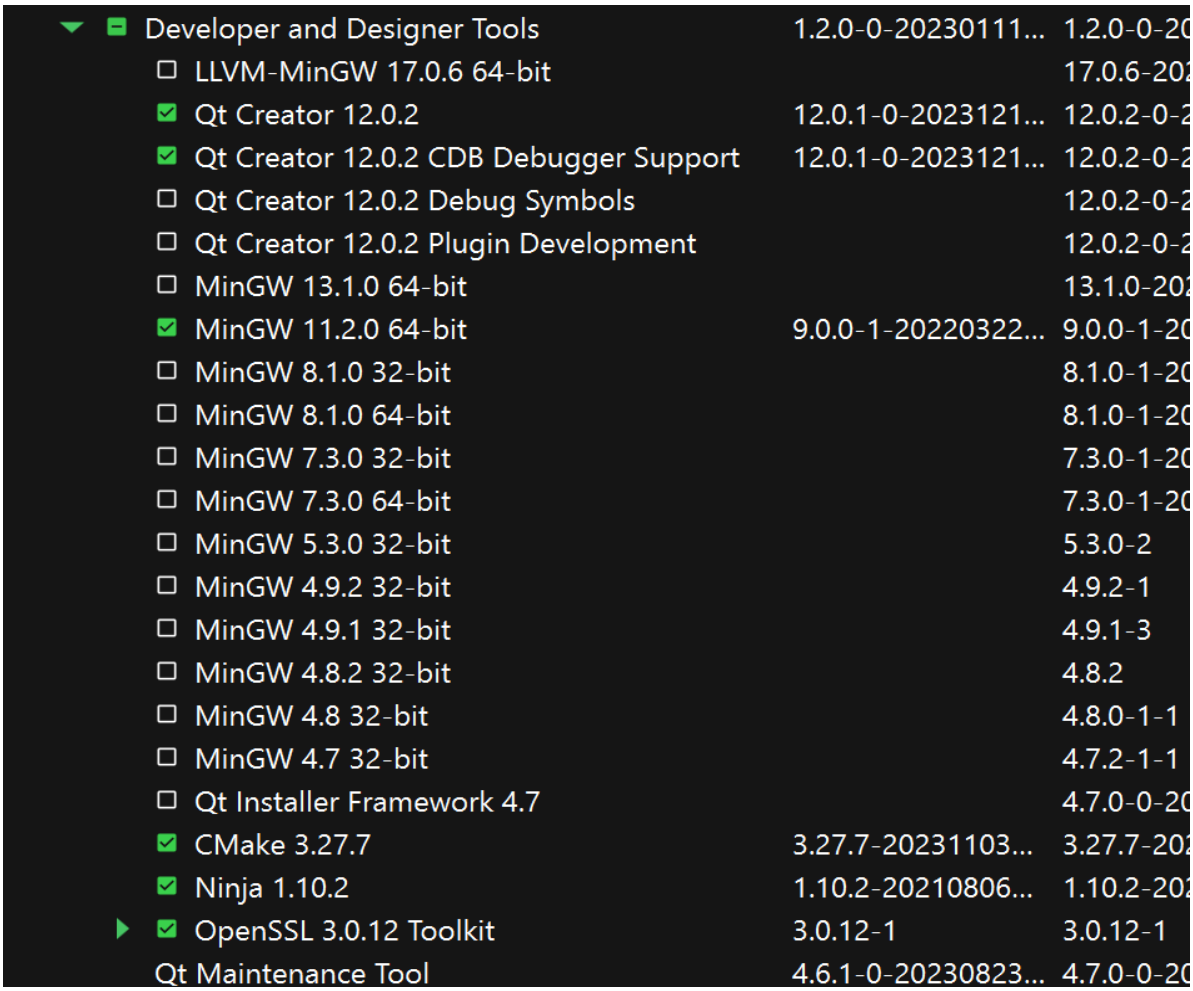
برای نصب این ابزار با مراجعه به مستندات رسمی Qt گام‌های مربوط به ایجاد حساب کاربری و دانلود نصب‌کننده آنلاین را انجام دهید (دقت نمایید که حتما نسخه open source را از این لینک (دانلود نصب کننده ی Qt) تهیه نمایید در غیر این صورت مجبور به دانلود و نصب مجدد خواهید شد) و سپس مطابق با تصویر زیر پکیج‌های مورد نیاز را انتخاب و نصب کنید. دقت شود در تصاویر زیر فقط پکیج‌هایی که تیک سبز دارند انتخاب شوند.

دقت کنید که برای دانلود پایدار و بدون مشکل حتما از ابزارهای گذر از تحریم مانند شکن استفاده نمایید. هنگام نصب با نسخه‌های متفاوتی از Qt مواجه خواهید شد، پیشنهاد ما برای نصب روی ویندوز نسخه 6.6.3 و برای لینوکس در صورت عدم وجود نسخه 6.6.3،

نسخه ی 6.7.3 است. پس از انجام تمرین اول، این فریمورک را از سیستم خود حذف نکنید زیرا برای تمرین 3 نیز به آن احتیاج خواهید داشت.



1- پکیج‌های مورد نیاز بخش اول



2-پکیج‌های مورد نیاز بخش دوم

برای ویندوز حتما جدید ترین نسخه‌ی **mingw** را تیک بزنید، در لینوکس نیز ابزار **g++** , **gcc** را نصب کنید:

```
sudo apt update
sudo apt install build-essential
g++ --version
```

- جمع بندی و نکات پایانی

معیارهای ارزیابی

- ✓ **صحت عملکرد:** هر دو پیاده‌سازی (TCP) و (UDP) باید به درستی کار کنند.
- ✓ **تجربه کاربری روان:** بازی باید برای هر دو بازیکن پاسخ‌گو و بدون تأخیر محسوس باشد.
- ✓ **اجرای صحیح در شبکه محلی:** بازی باید به‌صورت چندنفره در یک شبکه محلی (LAN) بدون مشکل اجرا شود.
- ✓ **دقت در پاسخ‌های نظری:** پاسخ‌ها به سؤالات تئوری باید صحیح، دقیق و همراه با تحلیل مناسب باشند.

- **ددلاین: 1403/12/28 (زمان تحویل متعاقبا اعلام خواهد شد)**
- تمرین‌های کامپیوتری به صورت انفرادی یا در گروه‌های ۲ نفره انجام می‌شوند. (حضور تمام اعضای گروه در جلسه تحویل الزامی است)
- هر ۲ نفر می بایست کار را تقسیم کنند. همچنین از Git برای ساختن branch و تقسیم issue ها استفاده نمایید. (با استفاده از commitها و تعیین issueها میزان مشارکت هر نفر مشخص می‌شود). بعد از انجام این کار کدها را در یک repository به نام CN_CA_1 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 خود را در قالب یک فایل txt در محل پاسخ تمرین قرار دهید. (از فرستادن فایل به صورت زیپ خودداری نمایید.) اکانت دستیاران آموزشی این تمرین را به Repo خودتان به عنوان Maintainer به پروژه اضافه کنید.

Github accounts:

- @hesamhrf
- @inaijin
- @AliDarabi9
- @TheSohrabX

- برای پیاده سازی این تمرین از C++ استفاده کنید.
- دقت کنید گزارش نهایی شما می‌بایست همانند 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 به خرج دهید.
- برای هر قسمت کد، گزارش دقیق و شفاف بنویسید. کدهای ضمیمه شده بدون گزارش مربوطه نمره‌ای نخواهند داشت.
- هدف این تمرین یادگیری شماست. لطفا تمرین را خودتان انجام دهید. در صورت مشاهده‌ی شباهت میان کدهای دو گروه، مطابقت سیاست درس با گروه متقلب و تقلب‌دهنده 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.