به نام خدا

درس مبانی برنامه‌سازی

موتور شبیه‌سازی بازی فوتبال

دانشکده مهندسی کامپیوتر

دانشگاه صنعتی شریف

نیم سال اول ۰۴ - ۰۵

مدرس:

دکتر محمدامین فضلی

تیم طراحی:

پوریا رحمانی

مانی ابراهیمی

متین باقری

مهدی جعفری

محمد فلاحی‌نژاد

علی مقدسی

نیما ملایی

فهرست مطالب

ارتباط با دستیاران آموزشی

سوالات خود در مورد پروژه را می‌توانید در سامانه کوئرا مطرح کنید.
در صورت تاخیر در پاسخ‌دهی، می‌توانید از طریق ایمیل به pooriarahmani84@gmail.com یا از طریق تلگرام به P1oo3r8ia4@ یا MatinBagheri@ پیام دهید.

توضیح کلی پروژه

در این پروژه، یک موتور[1] بازی فوتبال دو بعدی 6 vs 6 در اختیار شما قرار می‌گیرد و چیزی که از شما خواسته‌شده‌است، تکمیل دو مورد از فایل‌های کد درون آن است.
در فاز اول، شما باید کدهای مربوط به داوری بازی (مواردی مانند تشخیص گل شدن یا بیرون رفتن توپ از زمین مسابقه) را در فایل engine/logic/referee.c کامل کنید و در فاز دوم باید منطق بازی بازیکنان را با کامل کردن بدنه توابعی در فایل engine/logic/coach.c بنویسید.

توجه: اجازه تغییر دادن هیچ‌یک از فایل‌های موتور شبیه‌سازی غیر از این دو فایل را ندارید، بعلاوه، در فایل engine/logic/referee.c تنها مجاز به تکمیل توابع مطابق توضیحات پروژه و documentation فراهم شده به صورت کامنت در خود فایل referee.c هستید و در فایل coach.c علاوه بر تکمیل توابع، می‌توانید خودتان هم توابع دلخواهی تعریف کنید و از آن‌ها در **همان فایل** استفاده کنید.
برای دریافت انجین، ابتدا مخزن[2] موتور شبیه‌سازی را از این لینک clone کنید. سپس از CMakeFile موجود برای build کردن استفاده کنید. اگر در فرایند build کردن با مشکلی مواجه شدید، به بخش آخر (ضمیمه) مراجعه کنید و مطابق راهنمایی‌های آن عمل کنید.

معرفی اجزای موتور شبیه‌سازی

این بازی فوتبال، به طور کلی، به جز روند عادی بازی، صرفا out و کرنر و شروع مجدد پس از گل دارد.
در این بخش، صرفا یک توضیح اجمالی از اجزای موتور بازی می‌دهیم. برای مشاهده جزئیات بیشتر می‌توانید کدهای پوشه engine/entities را بخوانید.
همچنین برای مشاهده ثابت‌هایی مثل حداکثر ممکن سرعت شوت یا سرعت حرکت، حداکثر امتیاز

[1] engine
[2] repository

هر توانایی یک بازیکن یا حداکثر ممکن مجموع امتیازات توانایی یک بازیکن، می‌توانید به فایل core/constants.h مراجعه کنید.
برای توپ، آخرین تیمی که توپ را لمس کرده و صاحب فعلی توپ نگه‌داری می‌شود.
برای هر بازیکن، مختصات فعلی، سرعت فعلی و یک آرایه از توانایی‌ها (talent) نگه‌داری می‌شود.
چهار نوع توانایی داریم:

- agility: ضریبی که حداکثر سرعت بازیکن از روی آن تعیین می‌شود
- dribbling: امتیاز قدرت دریبل
- defence: امتیاز قدرت دفاع
- shooting : ضریبی که حداکثر سرعت شوت ممکن برای یک بازیکن را تعیین می‌کند

همچنین برای هر بازیکن state فعلی او نگهداری می‌شود.
چهار نوع state داریم:

- `IDLE`: به عنوان حالت پایه است، در این حالت، بازیکن هیچ کاری نمی‌کند
- `MOVING`: اگر بازیکن در حالت moving باشد، با توجه به منطقی که شما در فاز دوم باید پیاده‌سازی کنید، حرکت می‌کند.
- `INTERCEPTING`: در این حالت، اگر توپ جلوی پای بازیکن باشد، با صاحب توپ (در صورت وجود) رقابت می‌کند و به احتمالی که متناسب با نسبت قدرت defence او به قدرت dribbling صاحب توپ است، ممکن است صاحب توپ شود.
- `SHOOTING`: اگر بازیکن در حالت shooting باشد، مطابق منطقی که در فاز دوم باید پیاده‌سازی کنید، توپ را شوت می‌کند.

درنهایت، در struct scene اطلاعات کلی مربوط به بازی نگهداری می‌شود که می‌توانید برای اطلاع از جزئیات آن game/scene.c را مشاهده کنید تا بدانید برای پیاده‌سازی منطق بازی بازیکنان چگونه می‌توانید از اطلاعات بازی استفاده کنید.

فاز اول (1.6 نمره)

در این فاز، باید توابع درون فایل engine/logic/referee.c را پیاده‌سازی کنید. اگرچه جزئیات مواردی که باید پیاده‌سازی کنید و نکاتی که باید به آن‌ها توجه کنید به صورت کامنت در referee.c برای شما گذاشته شده است، در اینجا هم به توضیح مواردی که باید پیاده‌سازی کنید می‌پردازیم:

تابع goal

در این تابع باید بررسی کنید که آیا توپ گل شده است یا نه و براساس اینکه تیم سمت راست گل زده، تیم سمت چپ گل زده یا هیچ تیمی گل نزده است، مقدار مناسب را خروجی دهید.
شروطی که باید برای گل‌شدن توپ برقرار باشد به‌طور کامل در کامنت‌های بالای تابع برای شما مشخص شده‌است.

تابع out

در این تابع باید بررسی کنید آیا توپ به بیرون از زمین مسابقه رفته‌است یا خیر، اگر بیرون رفته، عبارتی که در بدنه تابع comment شده را چاپ کنید و مقدار درست را برگردانید.
توجه کنید گل هم، نوعی out محسوب می‌شود!

تابع referee

در این تابع باید وقوع رخداد goal یا out را شناسایی کرده خروجی دهید.

توابع مربوط به verify

در چهار تابع بعدی، باید مجاز بودن سرعت شوت بازیکنان، سرعت حرکت بازیکنان، مقادیر امتیازات talent بازیکنان و درستی state بازیکنان را بررسی کنید و در صورت وجود ایراد، پیام خطای کامنت شده در تابع را چاپ کنید.
همچنین اگر بازیکن یا توپ سرعت بیشتری از حد مجاز داشتند، سرعتشان را به بیشینه سرعت مجاز کاهش دهید.
بیشینه سرعت مجاز حرکت هر بازیکن (یا توپ)، ضریبی از بیشترین سرعت ممکن است که این ضریب با agility (یا shooting) هر بازیکن متناسب است.
در مورد state هم تنها حالت غیرمجازی که باید بررسی کنید بودن در حالت shooting در زمانی است که بازیکن صاحب توپ نیست.
بعلاوه در تابع مربوط به shoot باید بررسی شود موقع شروع بازی، جهت پاس به سمت زمین حریف نباشد.
توجه۱: غیر از پیاده‌سازی توابع، تنها می‌توانید توابع جدیدی تعریف کرده از آن‌ها در پیاده‌سازی توابع اصلی استفاده کنید.
توجه ۲: داوری پیاده‌سازی‌های شما هم با بررسی برنامه‌ای که نوشتید و هم با اجرای موتور با یک تیم از بازیکنانی که به صورت تصادفی بازی می‌کنند و قوانین مربوط به حرکت و شوت را نقض می‌کنند

انجام می‌شود.

فاز دوم (2.4 نمره)

در این فاز شما باید توابع فایل engine/logic/coach.c را پیاده‌سازی کنید.
در این فایل، به‌ازای هر شش بازیکن از دو تیم، سه تابع وجود دارد. یکی مربوط به تابع منطق shoot، یکی مربوط به تابع منطق move و دیگری مربوط به تابع منطق تغییر state بازیکن.
engine به‌گونه‌ای پیاده‌سازی شده‌است که در هر واحد زمانی، ابتدا تابع منطق تغییر state برای تمام بازیکنان صدا زده می‌شود و سپس بر اساس state فعلی، تابع مناسب صدا زده می‌شود. مثلا در وضعیت shooting تابع منطق شوت صدا زده می‌شود.

توابع shooting_logic

مواردی که باید در پیاده‌سازی منطق شوت بازیکنان رعایت کنید به صورت زیر است:

- سرعت شوت، بیش از سرعت مجاز نشود.
- .در هنگام شروع بازی، پاس شروع‌کننده، نباید به سمت زمین حریف باشد
- در هنگام پرتاب out توپ نباید از بیرون زمین به سمت بیرون زمین پاس داده شود.(انتخاب و تعیین موقعیت پرتاب کننده out توسط موتور شبیه‌سازی صورت می‌گیرد بطوریکه نزدیک‌ترین بازیکن به توپ، پشت توپ قرار می‌گیرد.)

توابع movment_logic

مواردی که باید در مورد منطق حرکت رعایت کنید به صورت زیر است:

- بازیکن هیچوقت به بیرون از زمین بازی نرود یا اگر به دلیلی مثل پرتاب out به بیرون از زمین بازی رفت، سریعا به زمین بازی برگردد .
- سرعت بازیکن از بیشینه سرعت مجازش بیشتر نشود. همانطور که پیش تر اشاره شد، بیشینه سرعت مجاز، از بیشینه سرعت ممکن حرکت، باتوجه به توانایی agility بازیکن به صورت خطی scale می‌شود.
- هیچگاه چند بازیکن در یک نقطه تجمع نکنند.

توابع change_state_logic

مواردی که برای این توابع باید رعایت کنید:

- اگر بازیکنی مالک توپ نیست، نباید در وضعیت shooting باشد.
- اگر بازیکنی مالک توپ هست، نباید در وضعیت intercept باشد.
- انتظار می‌رود اگر توپ بدون بازیکن به سمت بازیکنی آمد (به‌خصوص از جانب تیم حریف) بتواند آن را بگیرد.

موقعیت اولیه و امتیازات توانایی

در این فایل، باید امتیازهای talent بازیکن‌ها و موقعیت اولیه بازیکنان در زمین را نیز برای دو تیم مشخص کنید.
برای موقعیت اولیه، انتظار می‌رود:

- هیچ بازیکنی در زمین تیم حریف نباشد .
- هیچ بازیکنی با دایره وسط زمین اشتراک نداشته باشد (به‌جز بازیکن شروع‌کننده بازی که خود موتور شبیه‌ساز به صورت خودکار او را در وسط زمین می‌گذارد) .
- هیچ دو بازیکنی دقیقا در یک مکان نباشند.

برای امتیازات talent بازیکنان انتظار می‌رود:

- امتیاز هر talent عددی در بازه ۱ تا بیشینه مجاز برای هر talent باشد.
- مجموع امتیازات talent ها کمتر یا مساوی بیشینه مجموع امتیازات مجاز باشد .

توجه۱: در توابع منطق شوت و منطق حرکت بازیکن، تنها مجازید به‌ترتیب بردار سرعت حرکت توپ و بردار سرعت حرکت بازیکن را تغییر دهید در تابع منطق تغییر state بازیکن، تنها مجازید متغیر state بازیکن را تغییر دهید و مجاز به تغییر مستقیم متغیرهای دیگر یا استفاده از توابعی از engine مثل توابع فایل scene.c نیستید.
توجه۲: علاوه بر موارد بالا، انتظار می‌رود بازیکن هیچگاه گل‌به‌خودی نزند!!

بخش امتیازی (0.5 نمره)

سه مورد به عنوان بخش امتیازی درنظر گرفته شده است:

- **پیاده‌سازی دروازه‌بان**: برای دروازه‌بان انتظار می‌رود همواره در فاصله معین و کمی از دروازه تیمش حضور داشته باشد و زمانی که توپ در راستای چارچوب دروازه‌اش و در نیمه خودی است، دقیقا روبروی توپ باشد و در صورت برخورد توپ با او، بتواند توپ را بگیرد.

- **پیاده‌سازی حداقل یک بازیکن به عنوان مدافع:** برای مدافع، انتظار می‌رود همواره در نیمه زمین خودی باقی بماند و اگر توپ در نیمه خودی بود، همواره تلاش کند تا میان مرکز توپ و مرکز دروازه بایستد و اگر توپ جلوی پای او رسید، تلاش کند توپ را بگیرد.
همچنین انتظار می‌رود تخصیص امتیازات talent به مدافع به صورت منطقی انجام شود، مثلا قدرت defence بالا باشد و ...

- **پاس‌کاری کردن بازیکنان:** باید در طول اجرای بازی در زمان تحویل، (در روند بازی و نه در هنگام شروع مجدد) حداقل یکبار صاحب توپ به بازیکنی که به دروازه حریف نزدیک‌تر است و در مسیر پاس مانعی نیست، پاس بدهد.

ضمیمه: روش دوم دریافت وابستگی‌ها

برای اجرای موتور شبیه‌سازی، طبق راهنمای زیر عمل کنید:

اجرا روی لینوکس (دبیان و اوبونتو)

گرافیک این موتور شبیه‌سازی، با کمک کتابخانه SDL2 نوشته شده‌است، بنابراین شما به این کتابخانه (شامل SDL_ttf و SDL_image) نیاز دارید.
بنابراین ابتدا دستور زیر را اجرا کنید:

```
sudo apt update
sudo apt install libsdl2-dev libsdl2-image-dev libsdl2-ttf-dev
```

اجرا کنید.
حال، اگر در پوشه clone شده، پوشه‌ای به‌نام build وجود نداشت، پوشه build را در آن بسازید و وارد آن شوید. سپس دستور زیر را اجرا کنید:

```
cmake -G Ninja ..
```

در نهایت باید در پوشه build فایلی به نام build.ninja مشاهده کنید.
حالا در پوشه build دستور زیر را اجرا کنید:

```
ninja
```

اگر اجرا موفقیت‌آمیز باشد، در پوشه build/bin فایل باینری موتور شبیه‌سازی با نام soccerengine را ببینید که می‌توانید آن را اجرا کنید.

اجرا روی ویندوز

برای اجرا روی ویندوز، به یک محیط ایزوله لینوکسی نیاز دارید. اگر WSL دارید، می‌توانید با همان کار کنید و نیازی به MSYS2 ندارید.
در غیر این صورت، می‌توانید MSYS2 را از این لینک دانلود کنید. (توجه کنید حداقل به ویندوز ۱۰ شصت‌وچهار بیتی نیاز دارید)
پس از نصب MSYS2 از منوی استارت، MSYS2 MINGW64 را باز کنید. یک صفحه CLI باز می‌شود.
برای بروزرسانی پایگاه داده بسته‌ها دستور زیر را در کامندلاین باز شده بنویسید:

```
pacman -Syu
```

صفحه MSYS2 MINGW64 پس از پایان این دستور بسته می‌شود. دوباره آن را از منوی استارت باز کنید و با دستور زیر، بسته‌ها را بروزرسانی کنید:

```
pacman -Su
```

حال، باید gcc, cmake و کتابخانه‌های مربوط به SDL2 (شامل SDL2_image و SDL2_ttf را نصب کنید.) بنابراین دستور زیر را اجرا کنید:

```
pacman -S --needed \
mingw-w64-x86_64-gcc \
mingw-w64-x86_64-cmake \
mingw-w64-x86_64-SDL2 \
mingw-w64-x86_64-SDL2_image \
mingw-w64-x86_64-SDL2_ttf \
msys/vim
```

حالا نوبت به ساخت پوشه build و کامپایل و اجرای موتور شبیه‌سازی می‌رسد.
فرض می‌کنیم پروژه در درایو D و در مسیر D:\X\Y\Z قرار دارد. ابتدا باید از MSYS2 MINGW64 به این پوشه بروید. بنابراین بنویسید:

```
cd "/d/X/Y/Z"
```

توجه کنید مسیر را در double-quote و با / بنویسید.
حالا پوشه build را با دستور زیر در پوشه پروژه بسازید تا پروژه با CMake ساخته شود:

```
cmake -S . -B build -G "MinGW Makefiles"
```

درنهایت، در همان پوشه پروژه، با دستور

```
cmake --build build
```

پروژه را کامپایل می‌کنیم.
درنهایت اگر همه مراحل را به‌درستی انجام داده باشید، باید در پوشه build/bin یک فایل به‌نام soccerengine.exe ببینید که می‌توانید آن را اجرا کنید.

اگر پیش از اینکه هیچ کدی بنویسید، موتور شبیه‌سازی را اجرا کنید، باید صفحه‌ای شبیه تصویر زیر را ببینید: