مستند پروژه بازی مار

مقدمه

این پروژه یک بازی مار ساده را پیاده‌سازی می‌کند که در آن بازیکن می‌تواند مار خود را با استفاده از کلیدهای جهت‌نمای صفحه‌کلید حرکت دهد. بازی شامل مار بازیکن، سه مار حریف و تعدادی غذا است. قوانین بازی شامل برخورد مارها، رشد مارها با خوردن غذا و تشخیص برنده یا بازنده بودن بازی است.

کلاس‌ها و توابع

کلاس Point

این کلاس شامل ساختار ساده‌ای برای ذخیره مختصات x و y است. نقاط مختلف در بازی.

کلاس Snake

این کلاس مسئول مدیریت حرکت و وضعیت مارها است.

متغیرها:

- deque body: یک دک برای ذخیره بخش‌های بدن مار.
- Direction dir: جهت حرکت فعلی مار.
- int length: طول فعلی مار.
- SDL_Color color: رنگ مار.

توابع:

- Snake(int startX, int startY, SDL_Color snakeColor): سازنده کلاس که مختصات اولیه و رنگ مار را دریافت می‌کند و مار را مقداردهی اولیه می‌کند.
- void move(): مار را در جهت فعلی حرکت می‌دهد. اگر جهت NONE باشد، مار حرکت نمی‌کند.
- void grow(): طول مار را یک واحد افزایش می‌دهد.
- bool checkCollision(Point p): بررسی می‌کند که آیا نقطه داده شده با بخش‌های بدن مار برخورد دارد یا خیر.
- bool checkSelfCollision(): بررسی می‌کند که آیا سر مار با بدنه خودش برخورد کرده است یا خیر.
- bool isOutOfBounds(): بررسی می‌کند که آیا سر مار از مرزهای صفحه بازی خارج شده است یا خیر.
- void setDirection(Direction newDirection): جهت حرکت مار را تنظیم می‌کند.
- void render(SDL_Renderer* renderer): مار را روی صفحه نمایش رسم می‌کند.
- Point getHeadPosition(): موقعیت سر مار را بازمی‌گرداند.
- int getLength(): طول مار را بازمی‌گرداند.

کلاس Food

.این کلاس مسئول مدیریت غذاها در بازی است

:متغیرها

- Point position: موقعیت فعلی غذا.
- int screenWidth, screenHeight: ابعاد صفحه بازی.

:توابع

- Food(int screenWidth, int screenHeight): سازنده کلاس که ابعاد صفحه بازی را دریافت می‌کند و غذا را مقداردهی اولیه می‌کند.
- void generate(): موقعیت جدیدی برای غذا به صورت تصادفی در محدوده صفحه بازی تولید می‌کند.
- void render(SDL_Renderer* renderer): غذا را روی صفحه نمایش رسم می‌کند.
- Point getPosition(): موقعیت فعلی غذا را بازمی‌گرداند.

کلاس Game

.این کلاس مسئول مدیریت کل بازی است

:متغیرها

- SDL_Window* window: پنجره اصلی بازی.
- SDL_Renderer* renderer: رندرر برای رسم اجزای بازی.
- bool isRunning: وضعیت اجرای بازی (در حال اجرا یا متوقف).
- bool gameWon: وضعیت برنده یا بازنده بودن بازی.
- Snake player: شیء مار بازیکن.
- std::vector enemies: لیستی از مارهای حریف.
- std::vector foods: لیستی از غذاها.

:توابع

- Game(): سازنده کلاس که بازی را مقداردهی اولیه می‌کند و مار بازیکن، مارهای حریف و غذاها را ایجاد می‌کند.
- ~Game(): مخرب کلاس که منابع SDL را آزاد می‌کند.
- void run(): حلقه اصلی بازی که شامل مدیریت رویدادها، به‌روزرسانی وضعیت بازی، رسم اجزای بازی و بررسی برخوردها است.
- void handleEvents(): مدیریت رویدادهای ورودی از جمله کلیدهای فشرده شده و رویداد خروج از بازی.
- void update(): به‌روزرسانی وضعیت بازی شامل حرکت مارها و بررسی برخورد مار با غذا.
- void render(): رسم اجزای بازی شامل مار بازیکن، مارهای حریف و غذاها روی صفحه نمایش.
- void checkCollisions(): بررسی برخورد مار بازیکن با مارهای حریف، برخورد با خود و برخورد با مرزهای صفحه بازی.

- void moveEnemies(): حرکت مارهای حریف به صورت تصادفی در چهار جهت اصلی.

نحوه اجرای بازی

1. بر روی سیستم شما نصب شده است SDL2 مطمئن شوید که کتابخانه.
2. پروژه را کامپایل کنید.
3. برنامه را اجرا کنید.
4. از کلیدهای جهت‌نمای صفحه‌کلید برای حرکت مار بازیکن استفاده کنید.
5. مار بازیکن با خوردن غذاها رشد می‌کند.
6. بازی تا زمانی ادامه دارد که مار بازیکن زنده باشد و تمامی مارهای حریف از بین بروند یا مار بازیکن برخورد کند.

نتیجه‌گیری

این پروژه به صورت ساده اما کامل قوانین بازی مار را پیاده‌سازی می‌کند. کلاس‌های Game و Food، Snake هر کدام وظایف مشخصی دارند که با همکاری یکدیگر بازی را مدیریت می‌کنند. این پروژه می‌تواند به عنوان یک پایه مناسب برای توسعه بازی‌های پیچیده‌تر و پیشرفته‌تر مورد استفاده قرار گیرد.