Moving Car:

```
#include <SDL2/SDL.h>

#include <math.h>


// Window dimensions

const int WINDOW_WIDTH = 800;

const int WINDOW_HEIGHT = 600;


// Car dimensions

const int CAR_WIDTH = 200;

const int CAR_HEIGHT = 100;

const int WHEEL_RADIUS = 25;

const int HEADLIGHT_WIDTH = 20;

const int HEADLIGHT_HEIGHT = 15;


// Function to draw a filled rectangle

void drawFilledRect(SDL_Renderer* renderer, int x, int y, int width, int height, SDL_Color color) {

   SDL_Rect rect = { x, y, width, height };

   SDL_SetRenderDrawColor(renderer, color.r, color.g, color.b, color.a);

   SDL_RenderFillRect(renderer, &rect);

}


// Function to draw a filled circle

void drawFilledCircle(SDL_Renderer* renderer, int centerX, int centerY, int radius, SDL_Color color) {

   for (int y = -radius; y <= radius; y++) {

      for (int x = -radius; x <= radius; x++) {

         if (x * x + y * y <= radius * radius) {

            SDL_SetRenderDrawColor(renderer, color.r, color.g, color.b, color.a);

            SDL_RenderDrawPoint(renderer, centerX + x, centerY + y);
            }
        }
    }
}

int main() {
    SDL_Init(SDL_INIT_VIDEO);

    // Create a window
    SDL_Window* window = SDL_CreateWindow("Car", SDL_WINDOWPOS_UNDEFINED,
SDL_WINDOWPOS_UNDEFINED,
                                          WINDOW_WIDTH, WINDOW_HEIGHT, SDL_WINDOW_SHOWN);
    if (!window) {
        SDL_Log("Failed to create window: %s", SDL_GetError());
        return 1;
    }

    // Create a renderer
    SDL_Renderer* renderer = SDL_CreateRenderer(window, -1, SDL_RENDERER_ACCELERATED);
    if (!renderer) {
        SDL_Log("Failed to create renderer: %s", SDL_GetError());
        return 1;
    }

    // Set the background color
    SDL_SetRenderDrawColor(renderer, 255, 255, 255, 255);
    SDL_RenderClear(renderer);

    // Set the initial x-coordinate of the car
```

```
int carX = 0;

// Animation loop
int running = 1;
while (running) {
    SDL_Event event;
    while (SDL_PollEvent(&event)) {
        if (event.type == SDL_QUIT)
            running = 0;
    }

    // Clear the renderer
    SDL_SetRenderDrawColor(renderer, 255, 255, 255, 255);
    SDL_RenderClear(renderer);

    // Update the x-coordinate of the car
    carX++;
    if (carX > WINDOW_WIDTH)
        carX = -CAR_WIDTH;

    // Calculate the rotation angle of the wheels based on the car's movement
    double rotationAngle = (double)carX / WINDOW_WIDTH * 2 * M_PI;

    // Draw the car body (large rectangle)
    SDL_Color carColor = { 255, 0, 0, 255 };
    drawFilledRect(renderer, carX, WINDOW_HEIGHT / 2, CAR_WIDTH, CAR_HEIGHT, carColor);

    // Draw the smaller rectangle on top of the car body
    SDL_Color topColor = { 255, 0, 0, 255 };
```

```
drawFilledRect(renderer, carX + CAR_WIDTH / 4, WINDOW_HEIGHT / 2 - CAR_HEIGHT / 2,
        CAR_WIDTH / 2, CAR_HEIGHT / 2, topColor);

// Draw the wheels (circles) with rotation
SDL_Color wheelColor = { 0, 0, 0, 255 };
int wheelY = WINDOW_HEIGHT / 2 + CAR_HEIGHT / 2 + WHEEL_RADIUS * 2;
drawFilledCircle(renderer, carX + WHEEL_RADIUS,
         wheelY, WHEEL_RADIUS, wheelColor);
drawFilledCircle(renderer, carX + CAR_WIDTH - WHEEL_RADIUS,
         wheelY, WHEEL_RADIUS, wheelColor);

// Calculate the position of the wheels' center
int wheelCenterX = carX + WHEEL_RADIUS;
int wheelCenterY = wheelY;

// Calculate the position of the wheels' outer points after rotation
int wheelOuterX1 = wheelCenterX + cos(rotationAngle) * WHEEL_RADIUS;
int wheelOuterY1 = wheelCenterY + sin(rotationAngle) * WHEEL_RADIUS;
int wheelOuterX2 = wheelCenterX - cos(rotationAngle) * WHEEL_RADIUS;
int wheelOuterY2 = wheelCenterY - sin(rotationAngle) * WHEEL_RADIUS;

// Draw lines to represent rotation of the wheels
SDL_SetRenderDrawColor(renderer, 0, 0, 0, 255);
SDL_RenderDrawLine(renderer, wheelCenterX, wheelCenterY, wheelOuterX1, wheelOuterY1);
SDL_RenderDrawLine(renderer, wheelCenterX, wheelCenterY, wheelOuterX2, wheelOuterY2);

// Draw the headlight (green rectangle)
SDL_Color headlightColor = { 0, 255, 0, 255 };
drawFilledRect(renderer, carX + CAR_WIDTH / 2 - HEADLIGHT_WIDTH / 2,
                    WINDOW_HEIGHT / 2 - CAR_HEIGHT / 2 - HEADLIGHT_HEIGHT,
                    HEADLIGHT_WIDTH, HEADLIGHT_HEIGHT, headlightColor);

        // Update the screen
        SDL_RenderPresent(renderer);

        // Delay for a short duration to control the animation speed
        SDL_Delay(10);
    }

    // Clean up resources
    SDL_DestroyRenderer(renderer);
    SDL_DestroyWindow(window);
    SDL_Quit();

    return 0;
}
```