Trabalho Prático

Bruna Pereira Serrano da Mata [13733579]
Gabriel da Costa Merlin [12544420]
Pedro Augusto Monteiro Delgado [13672766]

Dezembro, 2023

1 Links

O vídeo de apresentação do grupo pode ser acessado pelo link: https://youtu.be/v17hZkKJ6SI. Já o projeto do grupo, no GitHub, pode ser acessado pelo link: https://github.com/gabrielcmerlin/jogoFloquinho.

2 Introdução

O jogo criado pelo grupo, chamado "Ajude o Floquinho!", foi inspirado na Campanha USP do Agasalho, um grupo de extensão da USP São Carlos. O personagem principal, o filhote de pinguim Floquinho, deve coletar roupas encontradas no cenário do jogo e colocá-las na caixa de arrecadação. Porém, está chovendo e as roupas não podem molhar! O jogador deve, então, ajudar o Floquinho a desviar das gotas de chuva, enquanto leva as roupas de um lado para o outro do cenário.

Figura 1: Tela inicial do jogo

3 Manual

Antes de mostrar como instalar o jogo, é necessário dizer que, devido à biblioteca gráfica utilizada, ele só funciona em sistemas Unix. Testamos o jogo em 1 Ubuntu 20.04, 1 Ubuntu 22.04 e 3 WSLs de Ubuntu 22.04, sua instalação funcionou em todos os Ubuntus nativos e em apenas 1 WSL. Logo, não recomendamos o uso de WSL para executar o jogo.

3.1 Dependências

O jogo foi feito utilizando a linguagem C++, sendo necessária a instalação de algum compilador dessa linguagem. Em nosso Makefile, utilizamos o compilador G++ 9.4.0, que pode ser instalado usando:

sudo apt update
sudo apt install build-essential

Para a parte visual do jogo, utilizamos a biblioteca SDL para C++, que pode ser instalada usando:

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

3.2 Compilação e execução

Devido ao uso de classes durante a programação, nosso jogo possui diversos arquivos. Por isso, utilizamos um Makefile para facilitar toda a compilação. Portanto, para compilar e executar o jogo, basta digitar no terminal:

make all
make run

3.3 Como jogar

Para jogar, são necessários os seguintes controles: setas (direita e esquerda), barra de espaço e enter. Assim que o jogo é aberto, é mostrada a opção para o jogador acessar o tutorial do jogo, no qual é explicado como jogar.

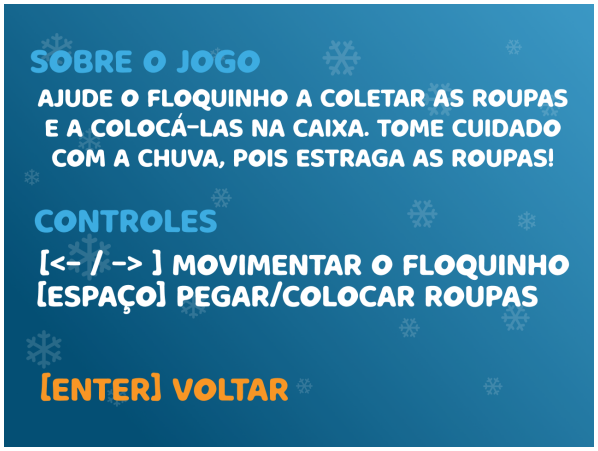

Figura 2: Tela de tutorial do jogo

Durante o andamento do jogo, o jogador deve movimentar o Floquinho horizontalmente pela tela, enquanto desvia das gotas de chuva que caem verticalmente. Quando uma gota toca o personagem, a rodada acaba. Então, o jogador pode reiniciar ou encerrar o jogo.

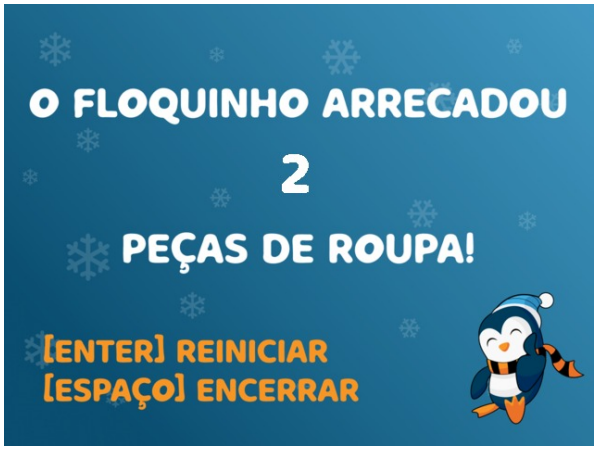

Figura 3: Tela final do jogo

Além disso, o jogador deve coletar as roupas que aparecem na parte à direita da tela, apertando espaço enquanto o Floquinho estiver próximo da pilha de roupas, e levá-las para a caixa que fica na parte à esquerda da tela, apertando espaço novamente enquanto o Floquinho estiver próximo da caixa. A cada vez que esse processo é concluído, o jogador recebe 1 ponto na pontuação, mostrada no canto superior esquerdo da tela.

Figura 4: Tela principal do jogo

4 Implementação

Durante a concepção do jogo, escolhemos retratar uma das principais tarefas da Campanha USP do Agasalho: coletar roupas. Um dos problemas enfrentados durante a arrecadação é a chuva, que pode estragar as roupas, tornando-as impróprias para a doação. É importante ressaltar que a chuva é um fenômeno da natureza, que apesar de ser, em partes, previsível, é incontrolável por parte do ser humano.

Para transformar essas ideias em código, utilizamos os conceitos de *threads* e semáforos. Um dos objetivos do uso de *threads* era separar a ocorrência da chuva das ações do Floquinho, ou seja, o jogador não tem acesso à seção do programa dedicada à chuva. Contudo, para o funcionamento do jogo, é necessário acessar a posição da gota de chuva, sendo esta a região crítica do programa, conforme explicaremos adiante.

4.1 Threads

Conforme apresentado, foi necessário a implementação de threads em nosso jogo. Criamos, então, 2 threads, responsáveis por movimentar as gotas de chuva que caem verticalmente na tela e por verificar se há contato entre a chuva e o Floquinho (o que representa o fim do jogo).

A primeira thread *t_chuva* foi implementada para gerar aleatoriamente a posição horizontal da gota de chuva e movê-la verticalmente pela tela (de cima para baixo). Dessa forma, conseguimos garantir que o obstáculo do jogo não seja interferido pelas interações do usuário com o programa, garantindo maior segurança e manutenibilidade ao código.

Já a segunda thread *t_contato* compara as posições horizontal e vertical do Floquinho e da gota, de modo a verificar se há colisão. Em caso de colisão, o jogo é encerrado.

4.2 Semáforos

Como explicado acima, a movimentação da chuva e a verificação de colisão estão cada uma a cargo de uma *thread* separada, o que faz com que as variáveis de posição da gota da chuva nos eixos X e Y sejam regiões críticas, já que a *thread* de movimentação altera essas variáveis, enquanto a de colisão as lê para verificar se o jogo deve acabar ou não.

Dessa forma, como há a concorrência por acesso/modificação das variáveis da posição da gota por parte das duas *threads* citadas, mostrou-se necessário o uso de semáforos, que garantem a sincronização

de tais acessos/modificações. Ou seja, quando uma das *threads* deseja acessar a posição da gota, ela sinaliza isso por meio do semáforo, de modo que a outra *thread* não faça leituras incorretas, garantindo a lógica proposta para o jogo.

5 Conclusão

Durante a construção do jogo, isto é, trabalhando na prática dos conceitos estudados em sala, foi possível compreender com mais clareza os conceitos de *threads*, região crítica, semáforos, paralelismo e concorrência, de forma que o seu uso foi feito de forma proveitosa para o bom funcionamento do jogo como um todo, como explicado nos tópicos anteriores.

Entretanto, enfrentamos algumas dificuldades durante a implementação do projeto, que serão citadas a seguir, juntamente do que foi feito para solucioná-las.

- **Uso de sistema operacional específico para rodar as bibliotecas de interface gráfica e de *threads*:** Foi necessário estabelecer uma divisão mais específica das tarefas entre nós para abranger os que conseguiam rodar a aplicação.
- **Inserção do conceito de *threads* na ideia inicial do projeto:** Pela nossa falta de experiência com implementações utilizando *threads*, inicialmente foi necessário uma maior discussão para pensar na inserção das *threads* dentro do jogo, felizmente, foi possível obter uma solução conveniente que alinham com os requisitos iniciais.
- **Debuggar código utilizando *threads*:** Devido ao paralelismo e imprevisibilidade de escalonamento, precisamos realizar um maior esforço para adequar à forma de compreender a implementação para conseguir debugar a aplicação, alinhando aos ensinamentos transmitidos pela professora em aula.

Dessa forma, conclui-se que os desafios propostos pelo projeto foram importantes para a evolução do nosso conhecimento e experiência, além de ser uma ótima maneira de complementar a teoria vista em sala.