INF01202 – Algoritmos e Programação

Turmas E e F – Prof. Joel Carbonera

Trabalho Prático Final – 2023/1

1. Motivação e Objetivos

O objetivo deste trabalho é exercitar as habilidades e conceitos de programação desenvolvidos ao longo da disciplina pela implementação de um jogo de computador em linguagem C. Um segundo objetivo é exercitar as habilidades de pesquisa, uma vez que o aluno poderá utilizar bibliotecas e funções não trabalhados em aula para implementar certas funcionalidades. Um terceiro objetivo é exercitar a habilidade de desenvolvimento em equipe, uma vez que o trabalho (implementação) deve ser realizado por duplas de alunos.

2. Descrição geral do jogo

O jogo a ser implementado é uma mescla inspirada em "Minecraft" e "Pac-Man", chamado “Pac-mine”. O jogador comanda um minerador espacial aventureiro cujo principal objetivo é minerar em cavernas perigosas em planetas desconhecidos e infestadas por toupeiras mutantes, as quais podem percorrer áreas soterradas que são inicialmente inacessíveis ao jogador. As cavernas são formadas por **áreas livres**, **áreas soterradas** e **paredes indestrutíveis**. Usando sua arma laser mineradora, o aventureiro pode destruir instantaneamente blocos de áreas soterradas com seus disparos, além de neutralizar as assombrosas toupeiras, é claro. No entanto, a tecnologia dessa arma ainda não é suficiente para quebrar paredes indestrutíveis que separam as áreas nas cavernas.

O desafio principal está na mineração. O jogador não consegue ver os minérios, inimigos e power-ups que estão nas áreas soterradas. Contudo, ele possui um sensor que permite identificar a existência de objetos de interesse em uma área de **81 blocos (9X9)** centrada na posição do jogador. O sensor permite detectar toupeiras mutantes, minério de ouro, esmeraldas, além de um power-up que dá poderes temporários ao jogador. Ao coletá-lo, por 3 segundos o jogador passa a poder ver todos os objetos do mapa. Deve-se tomar muito cuidado ao explorar as cavernas em busca de minérios, pois ao mesmo tempo que os tesouros estão espalhados e escondidos, os monstros estão também.

A principal mecânica do jogo é a capacidade de minerar os blocos à frente do jogador com sua arma laser, permitindo criar variados caminhos e estratégias enquanto explora as diversas cavernas. A arma do aventureiro realiza um disparo de um projétil que percorre o caminho à frente do jogador até atingir uma área soterrada, uma parede indestrutível ou uma toupeira. O jogador só pode realizar um disparo por vez, de modo que um novo disparo só pode ser realizado quando o projétil disparado anteriormente atingir um obstáculo. Quando o projétil atinge uma área soterrada, ela se torna livre e, no caso de alvejar uma toupeira, ela é neutralizada. O jogador coleta os tesouros e os power-ups quando passa sobre eles (fica na mesma posição em que eles se encontram). Caso o jogador seja atacado por uma toupeira

(estiver na mesma localização que ela), uma de suas **três vidas** é perdida e tanto ele quanto os adversários remanescentes são reposicionados em seus locais iniciais do mapa, enquanto o resto do mapa permanece inalterado. Se o jogador esgotar todas as três vidas, a partida é perdida definitivamente e o jogo deve ser reiniciado ou um jogo salvo deve ser carregado.

O jogo é estruturado em níveis/fases. A conclusão de cada fase ocorre quando o aventureiro consegue coletar todas as esmeraldas distribuídas no mapa da caverna atual. Além das esmeraldas, no entanto, também existirão minérios de ouro disponíveis para mineração, cuja recompensa por coletar será um bônus na pontuação do jogador. Ao passar de fase, uma nova caverna com uma nova dificuldade é carregada, à medida que se aventura em locais cada vez mais profundos e perigosos e em planetas desconhecidos.

3. Requisitos mínimos de implementação

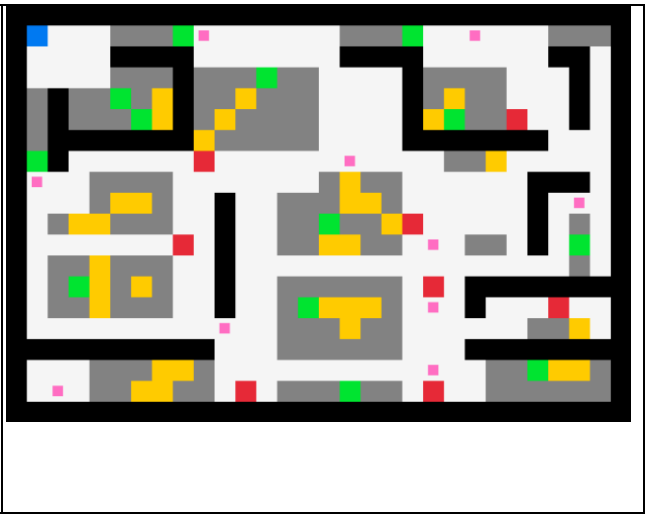
A implementação deverá contemplar os seguintes requisitos mínimos:

- Os elementos visuais do jogo devem ser implementados e exibidos em modo gráfico (exemplo: representando elementos com formas geométricas de cores diferentes);
- O jogo não deve ter delay, ou seja, ao ser disparada uma ação do jogador, o jogo deve responder imediatamente. Por exemplo: se o jogador movimentar o(a) minerador(a) para direita, ele deve imediatamente ir para a direita;
- O mapa da caverna deverá ser carregado a partir de um arquivo texto nomeado (“mapa1.txt” “mapa2.txt”, “mapa3.txt”, etc), onde o valor numérico indica o nível correspondente do mapa de cada caverna;
- O arquivo de texto deve estruturar o mapa no formato de no mínimo 20 linhas por 30 colunas (20x30) de blocos, os quais descrevem a configuração do labirinto;
- No arquivo texto, os artefatos do jogo devem ser representados com os seguintes caracteres:

J	Posição inicial do jogador
T	Posição inicial de uma toupeira
O	Minério de ouro
E	Esmeralda
#	Parede Indestrutível
S	Área soterrada
A	Power-up
<espaço>	Área livre

A figura a seguir demonstra um exemplo de arquivo de mapa e sua respectiva representação visual na tela usando formas geométricas simples com diferentes cores.

```
################################
#J    SSSEA        SSSE   A    SSS#
#     ####         ####        ##  #
#     SSS#SSSESS      #SSSS     #  #
#S#SSESO#SSOSSS      #SOSS     #  #
#S#SSSEO#SOSSSS      #OESST    #  #
#S########OSSSSS     #######      #
#E#        T        A     SSO       #
#A   SSSS          SOSS         ###  #
#    SOOS   #    SSSOOS          #  A  #
#  SOOSSS   #    SSESSOT         #  S  #
#          T #   SSOOSS A SS     #  E  #
#  SSOSSS   #                          S  #
#  SEOSOS   #    SSSSSS T  ########  #
#  SSOSSS   #    SEOOOS A #     T    #
#            A   SSSOSS           SSO  #
##########      SSSSSS     ########  #
#    SSSOOS              A   SSEOOS#
#  A SSOOSS T SSSESS T    SSS SSS#
################################
```



À esquerda, uma representação do arquivo de mapa, em caracteres, e à direita uma representação visual do mapa, em que cada elemento está representado por uma cor específica.

- Todos os mapas devem ser delimitados por paredes indestrutíveis nas primeiras e últimas linhas e colunas;
- A interação do jogador com o jogo deve ser realizada com as seguintes teclas e suas respectivas ações:

TAB	*Acessa o menu superior e espera o jogador escolher*
Setas (←, →, ↓ e ↑) ou teclas (A/a,D/d,S/s,W/w)	*Move o jogador uma posição na direção indicada*
G/g	Realiza um disparo à frente do jogador

- Cada nível poderá ter no máximo 200 toupeiras e um número qualquer de ouro, esmeraldas e power-ups. Vale ressaltar que, a fim de aumentar a dificuldade conforme o jogo progride, é interessante aumentar o número de esmeraldas e de toupeiras, além de diminuir a quantidade de power-ups e eventualmente construir áreas mais estreitas com paredes indestrutíveis para limitar a movimentação do jogador.
- Os movimentos do(a) minerador(a) devem respeitar a configuração do labirinto, impedindo ele possa se mover através de paredes indestrutíveis e áreas soterradas. As toupeiras, por outro lado, só não podem ser mover através de paredes indestrutíveis. Ou seja, elas podem se mover também por áreas soterradas (sem alterar o estado delas).
- As toupeiras devem se movimentar da seguinte maneira: no início do nível, cada toupeira deve seguir em uma direção aleatória. A cada unidade de tempo ela deve tentar se mover uma unidade nesta direção. A toupeira seleciona outra direção aleatória a cada 5 passos, ou sempre que ela não puder se mover na direção anterior (encontrar uma parede indestrutível) e passa a se mover nesta direção no momento seguinte;
- Se o(a) jogador(a) for atacado por uma toupeira e possuir vidas extras, ele e as toupeiras que não foram neutralizadas voltam para as suas posições iniciais. Seus itens que foram coletados na caverna atual são mantidos. Os itens já coletados não devem voltar a aparecer no mapa, as paredes soterradas que foram mineradas

deverão continuar demolidas e sendo uma passagem válida para o jogador, e as toupeiras que foram mortas não devem aparecer novamente;

- O jogador coleta um minério passando sobre ele (ocupando a mesma posição do minério).
- Caso uma esmeralda tenha sido minerada, ela é guardada no seu inventário até o fim da fase e o jogador ganha 100 pontos. Caso o jogador colete ouro, ele ganha 50 pontos. Quando o jogador neutraliza uma toupeira, ele ganha 200 pontos.
- Os minérios, inimigos e power-ups só devem aparecer na tela do jogador quando eles estiverem na área de alcance do sensor (9X9 blocos centrados na posição do jogador). Quando estiverem fora do alcance, devem ser representados de outra forma para que o jogador não consiga inferir a sua localização.
- A passagem de fase ocorre quando o jogador realiza a coleta de todas as esmeraldas da caverna daquela fase. Quando isso ocorre, uma nova configuração de caverna é carregada com os novos desafios.
- Caso o jogo tenha finalizado, uma tela de fim de jogo deve ser mostrada. Se o jogador cumpriu todos os objetivos do jogo, a tela deve indicar que o jogador venceu. Senão, deve indicar que a missão falhou. Em ambos os casos deve-se exibir a pontuação do jogador.
- O jogo deve mostrar as seguintes informações: a quantidade de vidas restantes do aventureiro, a pontuação atual, o nível atual, a quantidade de esmeraldas coletadas e a quantidade de esmeraldas total da fase;
- O jogador perde 1 vida cada vez que é atacado por uma toupeira;
- O jogo deve funcionar para qualquer número de fases, de modo que baste adicionar um novo arquivo de mapa no diretório do jogo (com numeração adequada) para aumentar o número de fases do jogo. Por exemplo, se incluirmos 100 arquivos de mapas no diretório, o jogo terá 100 fases. O jogo deve ser entregue com no mínimo 3 fases;
- O menu (que pode ser exibido no cabeçalho ou rodapé da tela) deve possuir as seguintes opções:
    - (N) Novo Jogo: Quando o usuário seleciona esta opção, um novo jogo é iniciado, a partir do primeiro nível. Todo progresso e número de vidas do jogador também é resetado.
    - (C) Carregar jogo: Quando o usuário seleciona esta opção, um jogo previamente salvo deve ser carregado. Você pode assumir que existe apenas um jogo salvo e que pode ser carregado.
    - (S) Salvar jogo: Quando o usuário seleciona esta opção, deve-se salvar todas as informações pertinentes do jogo em um arquivo, de modo que ele possa ser carregado e continuado do ponto em que foi salvo. Essas informações incluem posições do jogador e dos monstros, estados das paredes, itens desbloqueados, número de vidas, fase, etc. Tudo que for necessário para o jogo passar a funcionar normalmente a partir daquele ponto, como se o usuário nunca tivesse parado de jogar.

- (Q) Sair do jogo: Quando o usuário seleciona esta opção, o jogo é finalizado, sem salvar.
- (V) Voltar: Quando o usuário seleciona esta opção, deve-se abandonar o menu e o controle volta para o jogo, que continua do ponto em que parou quando o usuário acessou o menu.

O aluno deverá pesquisar a utilização de bibliotecas para criação de uma interface para o programa. A escolha da biblioteca e sua utilização ficarão a cargo do aluno, embora recomenda-se utilizar a biblioteca raylib.

4. Tarefas extras

Para os alunos motivados as seguintes tarefas extras são propostas. Embora opcionais, essas tarefas podem melhorar a avaliação final do seu trabalho, caso não tenham conseguido implementar corretamente algum dos requisitos básicos.

- Implemente o "cheat minerador invencível" onde o jogador nunca morre ao tocar em um monstro. O cheat deve ser ativado ao escrever "cowabunga" (sem as aspas), ou qualquer outra palavra com mais de um caractere durante o jogo (durante o game play, sem precisar parar o jogo);
- Faça a direção do disparo ser independente do último movimento do jogador, por exemplo, usando o mouse como entrada;
- Faça o disparo ser um sinalizador. Assim, ele deverá “iluminar” ao seu redor e mostrar os monstros e minérios;
- Armazenar as 10 maiores pontuações em um arquivo, de forma ordenada, em um ranking. Ao fim do jogo, caso o usuário obtenha uma das 10 maiores, este deve ser alertado, por meio de uma frase como, por exemplo, “Você obteve a segunda maior pontuação! Parabéns!”. Desta forma, sempre que o jogo for finalizado, faz-se necessária a verificação deste arquivo, para que o usuário possa ser alertado. Observa-se que enquanto não existir uma pontuação registrada, não é necessária a existência de um arquivo, que pode ser criado somente no momento em que o primeiro usuário finalizar o jogo. Quando a primeira pontuação for registrada, as demais podem ser indicadas pelo valor zero.
- Implementar diferentes tipos de monstros. O monstro básico do jogo é inspirado em uma toupeira mutante, porém também podem ser criados outros. Por exemplo, inimigos que disparam projéteis ou que explodem quando chegam perto do jogador, destruindo as áreas soterradas próximas.
- Tornar o jogo mais difícil, incluindo um timer com tempo limite, para o jogador passar de cada fase;
- Desenvolver um modo difícil em que o jogador só detecta os tesouros em um raio de dois blocos ao seu redor, ou seja, em um quadrado de 9x9 e/ou com monstros mais rápidos.
- Implementar outras recompensas por passar de fase, como uma armadura que faz com que o jogador possa ser atacado mais de uma vez pelos inimigos, etc.
- Emitir avisos sonoros em alguma(s) da(s) situação(ões) a seguir:

    - quando o jogador for devorado por um monstro;
    - quando o jogador realiza um disparo
    - quando o jogador perder uma vida;
    - quando o jogador coleta um power-up
    - quando o jogador coleta uma esmeralda;
    - quando o jogador destrói uma parede com minério de ouro
    - quando o jogador passa de nível;
    - quando o jogo acabar.
    - quando o jogador passa de fase

- Criar uma movimentação inteligente para a perseguição dos monstros, onde cada monstro tenta perseguir o aventureiro (em vez de se mover aleatoriamente), percorrendo o menor caminho possível até o jogador.
- Seja criativo, implemente suas ideias (mas não esqueça dos requisitos mínimos e converse com o professor antes)!

5. Dicas

- Usar uma IDE (como o CodeBlocks) facilita o desenvolvimento.
- Utilizar funções bloqueantes de leitura (como scanf) não é uma alternativa viável para elaborar o jogo, porque o jogo deve continuar sua dinâmica mesmo quando o jogador não fizer nada. Uma alternativa é usar a função kbhit da conio.h para verificar se o usuário digitou algo. Mas existem outras formas.
- Bibliotecas como a Allegro, sdl2 e raylib podem ser usadas para criação de interfaces gráficas. Uma biblioteca altamente recomendada é a raylib (fácil de usar, bem documentada e bastante versátil), que permite a elaboração de interfaces gráficas e jogos.
- Utilizar structs para representar os elementos dinâmicos do jogo.
- Para o jogo não executar muito rápido, pode-se utilizar a função sleep no loop principal, fazendo o jogo aguardar alguns milissegundos entre uma iteração e outra.
- Considere o uso e repositórios remotos de código (Github, Gitlab, Bitbucket) com controle e versão para desenvolvimento em equipe.
- Verifique o material adicional oferecido no moodle como suporte para este trabalho.
- É comum, à medida que formos estudando novos conteúdos, surgirem ideias de como utilizar esses conteúdos para melhorar o jogo. Isso acaba gerando modificações constantes no jogo. Isso é esperado.
- Combine o aumento do número de toupeiras, esmeraldas, e quantidade de paredes indestrutíveis (formando corredores mais estreitos) e áreas soterradas, e diminuição de power-ups para gerar níveis com dificuldade crescente.
- Enquanto ainda não tivermos estudado como manipular arquivos, é possível inicialmente representar o conteúdo do mapa diretamente em código (hardcoded), e utilizar essa informação para elaborar a representação visual e movimentação. Depois de estudar a manipulação de arquivos, essa informação do mapa pode ser carregada dos arquivos.

6. Requisitos Administrativos

- O trabalho deverá ser realizado em duplas. Informar os componentes da dupla até o dia **06 de Julho**, via mensagem direta para o professor pelo teams. Ou seja, no dia anterior ao da aula prática. Caso até esta data alguém não tiver dupla, a dupla será atribuída aleatoriamente. Se tivermos número ímpar de alunos, podemos ter um trio. Duplas podem ser entre quais alunos das turmas E e F.
- Até o dia **31 de Agosto**, a dupla deverá submeter via Moodle um arquivo zip cujo nome deve conter o(s) nome(s) do(s) aluno(s). O arquivo zip deve conter:
    - Um relatório contendo a descrição do trabalho realizado, contendo a especificação completa de como os elementos do jogo foram representados, como foi implementada a interação dos componentes interativos, bem como as estruturas e funções utilizadas e uma explicação de como usar o programa.
    - Os códigos fontes devidamente organizados e documentados (.c).
    - Bibliotecas necessárias para executar os códigos.
    - O jogo devidamente compilado e pronto para ser executado.
- O trabalho será obrigatoriamente apresentado durante a aula prática do dia **01 de Setembro**. Ambos os membros da dupla devem saber responder perguntas sobre qualquer trecho do código e estes serão avaliados separadamente.
- Na aula prática do dia **04 de Agosto**, os alunos deverão apresentar para o professor o que foi desenvolvido do trabalho até então. Espera-se que a esta altura já tenham implementado a representação gráfica do mapa e a movimentação do jogador e das criaturas. Caso o grupo não apresentar este mínimo neste momento, todos os integrantes perdem **5%** da nota final do trabalho.
- Os seguintes itens serão considerados na avaliação do trabalho: estruturação do código em módulos, documentação geral do código (comentários, indentação), “jogabilidade” do jogo e atendimento aos requisitos definidos.
    - (2 pontos) Habilidade em estruturar programas pela decomposição da tarefa em subtarefas, utilizando subprogramação para implementá-las.
    - (1 ponto) Organização e documentação de programas (indentação, utilização de nomes de variáveis significativos, abstração dos procedimentos para obter maior clareza, uso de comentários no código).
    - (2 pontos) Domínio na utilização de tipos de dados simples e estruturados (arranjos, conjuntos, estruturas) e passagem de parâmetros.
    - (5 pontos) Atendimento dos requisitos do enunciado do programa: menus, elementos gráficos esperados, interação, movimentação de personagens, funções dos menus.

Importante: Trabalhos copiados não serão considerados. Existem ferramentas que possibilitam a detecção automática de plágio, as quais serão utilizadas na correção. Se for detectado plágio, todos os trabalhos relacionados serão desconsiderados.
