Trabalho Prático Final – 2018/2

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 conceitos ainda 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.

O jogo que deverá ser implementado é inspirado no jogo de Atari chamado “Mouse Trap”. A Fig. 1 ilustra uma tela do jogo original. A mecânica do jogo é similar ao famoso jogo “Pac-Man". Neste clone de Pac-Man, o personagem principal que é controlado pelo jogador é substituído por um rato que tem como objetivo comer todos os queijos disponíveis no labirinto. Entretanto, no labirinto também estão presentes gatos famintos que desejam devorar o rato, assim como ossos que transformam o personagem temporariamente em um cachorro, o qual tem o poder de enviar os gatos para sua posição inicial no labirinto. Um diferencial do Mouse Trap em relação ao Pac-Man é a possibilidade do jogador mudar a geometria do labirinto ao pressionar uma tecla que muda a orientação de algumas paredes.

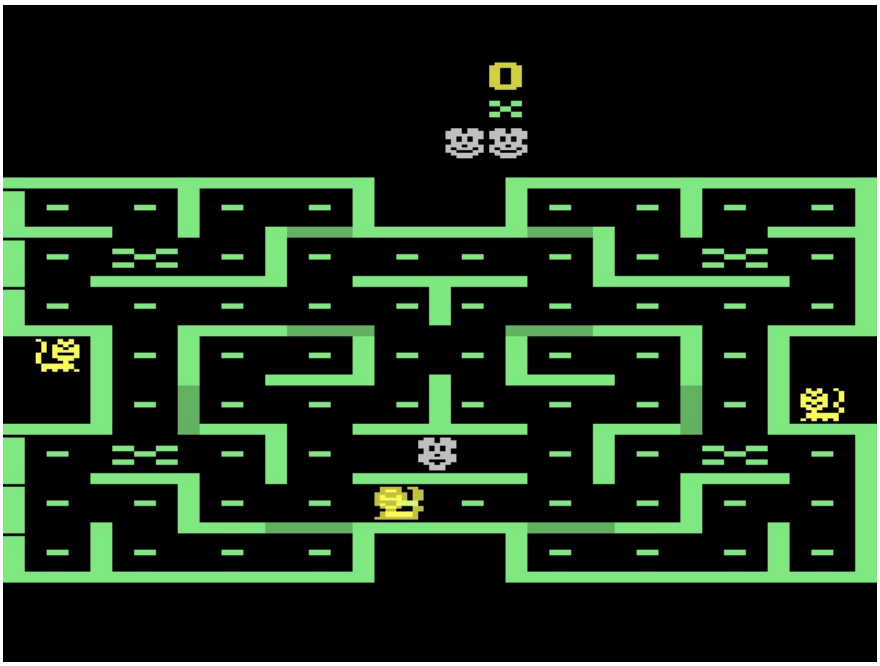


Figura 1 – *Screenshot* do jogo *Mouse Trap* para Atari (1981)

O jogo é estruturado em níveis, e o jogador ganha quando passar por todos os níveis. Para passar de nível, é necessário que o rato coma todos os queijos disponíveis no nível. Durante uma partida o jogador dispõe de três vidas. Cada vez que um gato encontra o rato, o jogador perde uma vida. Caso o rato perca todas as suas vidas, é considerado *game*

over. O vídeo do link a seguir https://www.youtube.com/watch?v=Bvp0xq4L298 demonstra a mecânica do jogo.

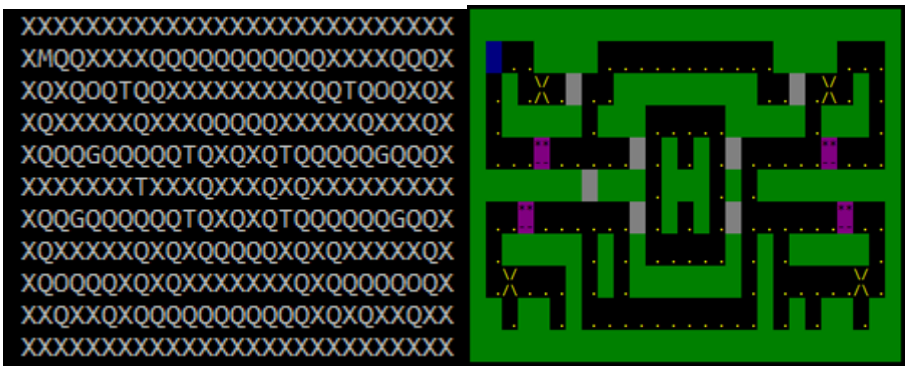
2. Requisitos Mínimos de Implementação

A implementação realizada pelos alunos deverá respeitar os seguintes requisitos mínimos:

- Os elementos visuais do jogo devem ser implementados e exibidos em modo texto (caracteres);
- 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 rato para direita, o rato deve imediatamente ir para a direita;
- O mapa do labirinto deverá ser carregado a partir de um arquivo texto nomeado “nivel01.txt”, onde o valor numérico indica o nível correspondente do mapa;
- O arquivo de texto deve estruturar o mapa no formato de 11 linhas por 27 colunas (11x27) de caracteres, os quais descrevem a configuração do labirinto;
- No arquivo texto, os artefatos do jogo devem ser representados com os seguintes caracteres:

(espaço em branco)	Posição vazia
M	Posição inicial do Rato
T	Porta Movediça
G	Posição inicial de um gato
O	Osso
X	Parede
Q	Queijo

A figura a seguir demonstra um exemplo de arquivo de mapa e sua respectiva apresentação na tela.



- 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,D,S,W)	Move o rato uma posição na direção indicada
b/B	Muda a orientação das portas

- Um labirinto sempre terá exatamente 7 portas, 4 gatos e 4 ossos;
- Os movimentos dos gatos e do rato devem respeitar a configuração do labirinto, impedindo que qualquer um deles possa se mover através de paredes ou portas;
- Os gatos devem se movimentar da seguinte maneira: no início do nível, cada gato deve seguir em uma direção aleatória. A cada segundo o gato deve tentar se mover uma unidade nesta direção. Sempre que o gato não puder se movimentar para a esta direção (por exemplo, se houver uma parede), deve ser selecionada uma direção aleatória que será utilizada na próxima tentativa de se mover;
- Cada porta tem dois estados: aberta ou fechada. As portas começam fechadas, quando a tecla B é pressionada elas “abrem” e mudam uma posição à direita e uma posição abaixo. Note que você deve planejar os labirintos para que esse movimento das portas faça sentido;
- Se o rato for devorado, e o jogador possuir vidas extras, a posição do rato é restaurada para a mesma do início do jogo. Os gatos também devem retornar para suas respectivas posições iniciais neste caso.
- O jogo deve apresentar um placar com a pontuação do jogador. Para cada queijo comido pelo rato o jogador ganha 10 pontos. Para cada gato resetado quando o rato estiver em modo cachorro o jogador ganha 50 pontos.
- O jogo deve mostrar também a quantidade de vidas restantes do rato;
- O modo cachorro, ativado quando o rato pega um osso, deve durar 5 segundos;
- O programa deve ser capaz de funcionar com até 99 níveis. Mas deve ser capaz de rodar com qualquer quantidade de níveis entre 1 e 99. Vamos assumir inicialmente 3 níveis.
- 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 e toda a pontuação 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. Assuma que existe apenas no máximo 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 rato, portas, gatos, queijos, ossos; número de vidas, nível jogado, etc.
    - (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. Algumas bibliotecas sugeridas são a conio2 e a ncurses, usadas para desenvolver interfaces em modo texto, para Windows e para Linux respectivamente. O aluno pode desenvolver interfaces gráficas se quiser. Neste caso, uma biblioteca que poderia ser utilizada seria a Allegro.

3. Tarefas extras

Para os alunos motivados as seguintes tarefas extras são propostas. Embora opcionais, essas tarefas podem melhorar a avaliação final do seu trabalho.

- Implemente o "*cheat* rato invencível" onde o rato nunca morre ao tocar em um gato. O cheat deve ser ativado ao escrever "sou covarde" (sem as aspas);
- Armazenar as 10 pontuações mais altas em um arquivo. Ao fim do jogo, caso o usuário obtenha uma das 10 maiores pontuações, 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.
- Emitir avisos sonoros em alguma(s) da(s) situação(ões) a seguir:
    - quando o rato for devorado por um inimigo;
    - quando o rato virar cachorro;
    - quando o rato perder uma vida;
    - quando o jogo acabar.
- Criar uma movimentação inteligente para a perseguição dos gatos, onde cada gato percorre o menor caminho possível até o rato.
- Seja criativo, implemente suas ideias (mas não esqueça dos requisitos mínimos e converse com o professor antes)!

4. Requisitos Administrativos

- O trabalho deverá ser realizado em duplas. Informar os componentes da dupla até o dia **03 de outubro**, por e-mail ao professor da disciplina. Ou seja, no dia anterior ao da aula prática.
- Até o dia **21 de novembro**, 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).
- O trabalho será obrigatoriamente apresentado durante a aula prática do dia **22 de Novembro**. Ambos membros da dupla devem saber responder perguntas sobre qualquer trecho do código e estes serão avaliados separadamente.
- Na aula prática do dia **25 de Outubro**, os alunos deverão fazer um breve relato a respeito do andamento 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) Documentação de programas (identação, utilização de nomes de variáveis, abstração dos procedimentos para obter maior clareza, uso de comentários no código).
    - (2 ponto) Domínio na utilização de tipos de dados simples e estruturados (arranjos, conjuntos, estruturas) e passagem de parâmetros.
    - (1 ponto) Formatação e controle de entrada e saída, com construção de interfaces que orientem corretamente o usuário sem instruções ou intervenção adicional do programador.
    - (1 ponto) Utilização de arquivos binários e de texto.
    - (1 ponto) Jogabilidade
    - (2 pontos) Atendimento dos requisitos do enunciado do programa: menus, elementos gráficos esperados, interação, movimentação de personagens, funções dos menus.

5. Dicas

- Use o CodeBlocks para desenvolvimento;
- Alternativas à função scanf que não são bloqueantes (i.e. esperam o usuário digitar a entrada e pressionar ENTER) podem ser encontradas na biblioteca conio.h. Veja os exemplos fornecidos na apresentação do enunciado, disponíveis no moodle;
- A biblioteca conio.h contém várias funções úteis, como kbhit e getch;
- A biblioteca windows.h contém a função SetConsoleCursorPosition que pode ser usada para definir a posição do cursor do terminal;
- Além da conio.h, existem outras bibliotecas avançadas que podem ser usadas, como a Ncurses, para Linux, e a como a Allegro, para criação de interfaces gráficas.
- Para o jogo não executar muito rápido, pode-se utilizar a função sleep;
- Para representar todos os símbolos da tabela ASCII estendida é necessário que a variável correspondente seja declarada como unsigned char;
- Verifique o material adicional oferecido no moodle como suporte para este trabalho.

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.