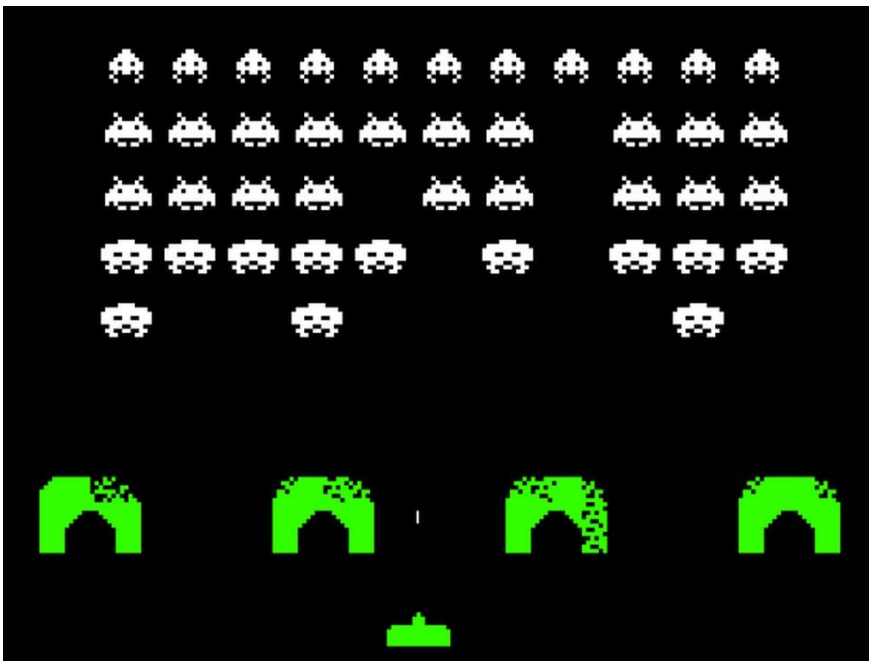
Universidade do Minho
Escola de Engenharia

SPACE INVADERS



Trabalho elaborado por:

João Peixoto nº 91960

João Rocha nº 91936

Ivan Castro nº 87970

Resumo

O Space Invaders é um jogo de videojogo do tipo arcade desenhado por Tomohiro Nishikado e lançado em 1978. Foi originalmente construído pela Taito Corporation, sendo pouco tempo depois licenciado para produção nos Estados Unidos pela Midway Games. Space Invaders foi um dos primeiros jogos de tiro em 2D.

O objetivo é destruir frotas de naves alienígenas com uma nave espacial humana artilhada, para ganhar o maior número de pontos possível. Apesar da sua simplicidade comparativamente aos jogos de hoje em dia, este jogo ajudou a expandir a indústria dos videojogos à escala global.
Quando o jogo foi inicialmente lançado, alcançou um enorme sucesso e tornou-se mundialmente popular. No jogo Space Invaders, o jogador controla os movimentos de uma nave espacial artilhada com um canhão, que se movimenta na parte inferior do ecrã.

Na parte superior, estão posicionados “alliens” organizados em linhas, que se deslocam até às extremidades (esquerda e direita) do ecrã, sendo que, cada vez que o pelotão inimigo contacta com estas extremidades, inverte a marcha e desce em pequenos saltos em direção à parte inferior do ecrã.

O objetivo do jogador é evitar que os aliens o alcancem na sua secção de atuação. Para essa tarefa o canhão possui munição infinita para disparar. Ao destruir um número grande aliens, os restantes começam a marchar mais rapidamente em direção à nave humana.
Quando todos os aliens são eliminados, um novo pelotão é constituído, iniciando uma linha abaixo da formação anterior. Ocasionalmente uma nave surge e, se atingida, permite ao jogador obter pontos extras.

Índice

1. Introdução

Foi nos proposto o desafio de desenvolver um jogo “retro” na linguagem C, interativo, que promova a não só a exploração dos conceitos de C dados na unidade curricular de Programação de Computadores, mas também a criatividade e a capacidade de exploração de hipóteses para a resolução de problemas.

Embora soubéssemos que o caminho a partir daqui iria ser trabalhoso e que iriam ser testadas não só as nossas capacidades enquanto programadores como também o trabalho em equipa e espírito de união, encaramos este desafio da melhor forma.

2. Descrição do Problema

O desafio consistiu em desenvolver o jogo “Space Invaders” em C, que integre na sua base uma interface gráfica simples mas muito bem cuidada (á semelhança do jogo original), gestão de pontuação e “hall of fame”, que controlará não só a pontuação “ao vivo” enquanto se joga como também o acesso ao Top 10 das melhores pontuações, mais do que um nível com diferentes graus de dificuldade, itens de prémio (por exemplo, vidas extras ou armas alternativas), “Boss” final e possibilidade de guardar/restaurar jogo.

O desenvolvimento desta aplicação levou-nos a explorar todos os conceitos dados nas aulas, quer nas teóricas quer nas práticas.
Desde a manipulação de variáveis auxiliares simples, ciclos e estruturas de decisão, funções para a organização e reutilização de código (o que se verificou muito importante), matrizes (por exemplo, para a configuração dos pelotões de “alliens”), estruturas de dados para guardar informação relevante à escrita e leitura de ficheiros para o alojamento em memória permanente do estado do jogo e do carregamento do mesmo.

Para tal, e para além dos conhecimentos adquiridos nas aulas de Programação, tivemos de realizar várias pesquisas, no intuito de desenvolver a melhor aplicação possível.

3. Arquitetura do Sistema

De uma forma resumida, o jogo possui quatro modos de jogo.
O primeiro modo é o convencional de todos os jogos (modo solo).
O segundo e o terceiro são, como nós apelidamos, os modos "Desafio".
Um é o "Multiplayer" tradicional, isto é, é constituído por dois jogadores e outro em que é possível estarem três jogadores a jogarem em simultâneo!
Por fim, temos o modo equipa, em que dois jogadores unem forças e tentam destruir o pelotão de "alliens" inimigo.

É possível ao utilizador escolher o tipo de arma que deseja utilizar, sendo lhes atribuído a possibilidade de escolha entre três tipos de armas.

O utilizador regista-se no jogo (uma espécie de "log-in"), em que colocará as suas credenciais (3 caracteres que o identifiquem).
Este registo irá possuir bastante importância futuramente, nomeadamente na identificação do utilizador no "Hall of Fame" e no processo de guardar/carregar jogo.

Como em qualquer outro jogo, não nos poderíamos esquecer de implementar diferentes graus de dificuldade.
Constam no jogo três tipos de dificuldade, uma fácil, uma intermédia e uma difícil. O utilizador tem total controlo sobre a dificuldade, sendo este mesmo capaz de escolher a que mais se ajusta à sua perícia.

Após a recolha de todas estes dados, irá aparecer uma mensagem de boas vindas para o utilizador, dando-se assim, o início do jogo.

4. Implementação do Jogo

a) Representação do Estado do Jogo:

Tal como no Space Invaders original, o objetivo do jogo é tentar destruir a frota de "alliens" inimiga e evitar que estes nos atinjam, atingindo o maior número de pontos possível.
Partindo do original e inovando um pouco, criámos uma série de regras/caraterísticas do novo jogo que será importante referir.

O modo Solo e os modos "Desafio" irão se desenrolar numa matriz 28 por 31 enquanto no modo Equipa/Multiplayer o mapa irá ser mais abrangente, atingindo as 61colunas.

Ambos os modos partilham as mesmas as mesmas regras de jogo e as mesmas caraterísticas, à exceção do modo Equipa que como o próprio nome indica, duas naves partilham a mesma "tela de jogo" e unem forças para derrotar o inimigo.
Este é também o único modo em que a escolha da arma por parte dos jogadores não pode ser a mesma, tornando-se assim mais fácil a identificação das naves dos dois jogadores.

Uma das caraterísticas do jogo e em que decidimos inovar foi na criação/implementação de diferentes alliens no pelotão inimigo.
Para tal, criámos três tipos de alliens que serão distribuídos pelo mapa e que possuem diferentes tipos de vulnerabilidade.
Os alliens do tipo 1 (correspondente ao "W" no jogo) morrem com um tiro, os do tipo 2 (correspondente ao "@") quando acertados pela primeira vez transformam-se nos alliens do tipo 1 e só quando voltam a ser atingidos é que morrem. Já os do tipo 3 (correspondente ao "A"), só morrem quando são atingidos 3 vezes, transformando-se antes em alliens do tipo 2 e do tipo 1, respetivamente.

Para proteger a nave dos jogadores (à semelhança do jogo original), temos uma espécie de escudos que protegem dos tiro
s inimigos.
Também estes escudos baseiam-se no método anterior, sendo inicialmente colocados escudos de nível 3 distribuídos em pontos estratégicos do mapa.
À medida que estes são alvejados, estes vão "perdendo força", caindo de nível (isto é, passando para escudos de nível 2).
Ao chegarem ao nível 1 e quando atingidos, os escudos vão desaparecendo ficando o jogador sem proteção.
Nota: Os tiros do próprio jogador também desintegram os escudos!!

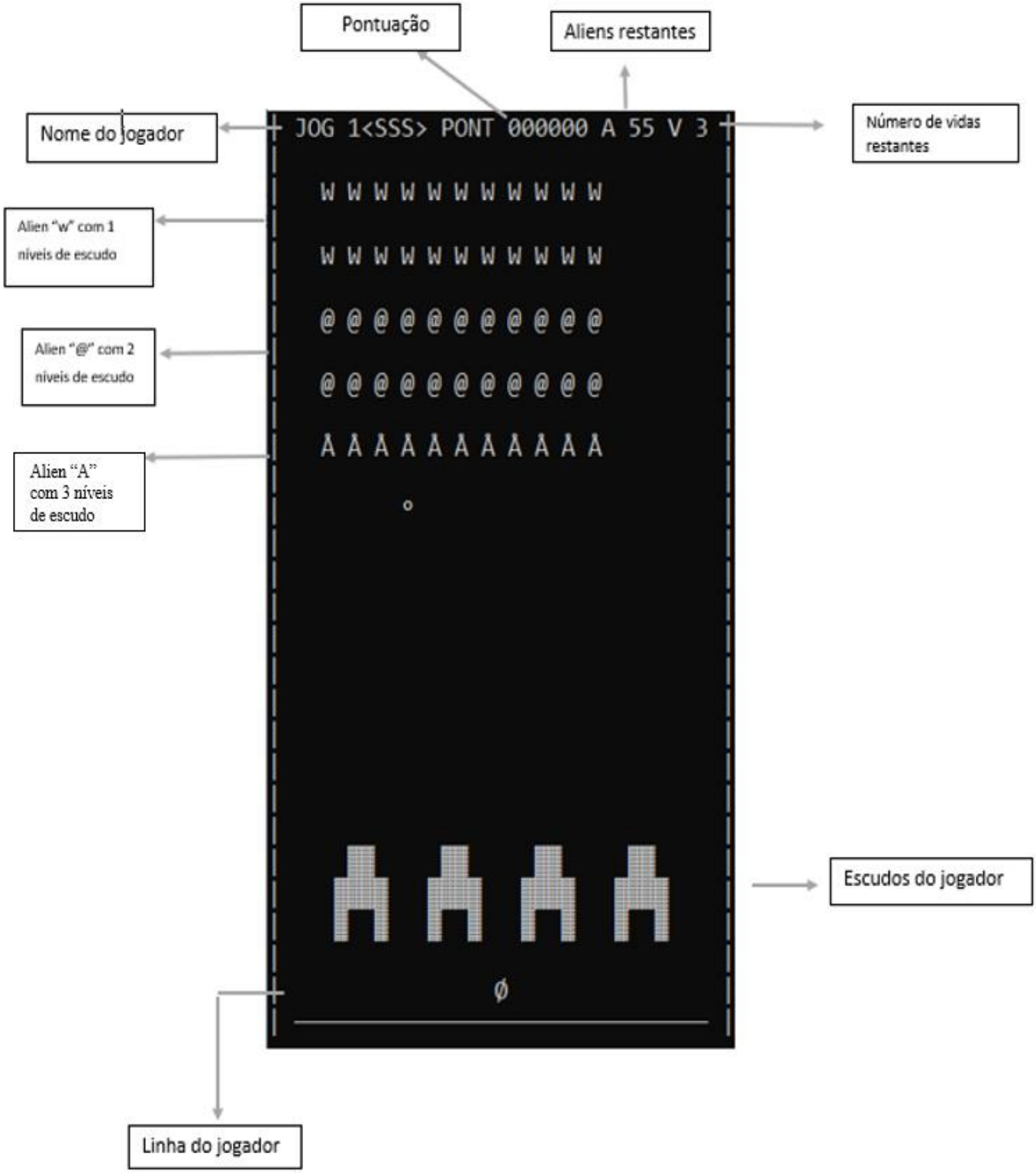
Cada jogador possui 3 vidas por partida, fornecendo assim mais tempo de jogo ao utilizador e tornando-o o mais realista possível (tendo em conta os jogos "retro" de hoje em dia).

O utilizador é capaz de acompanhar em tempo real a quantidade de vidas que possuiu, a sua pontuação bem como a quantidade de alliens que faltam derrotar.

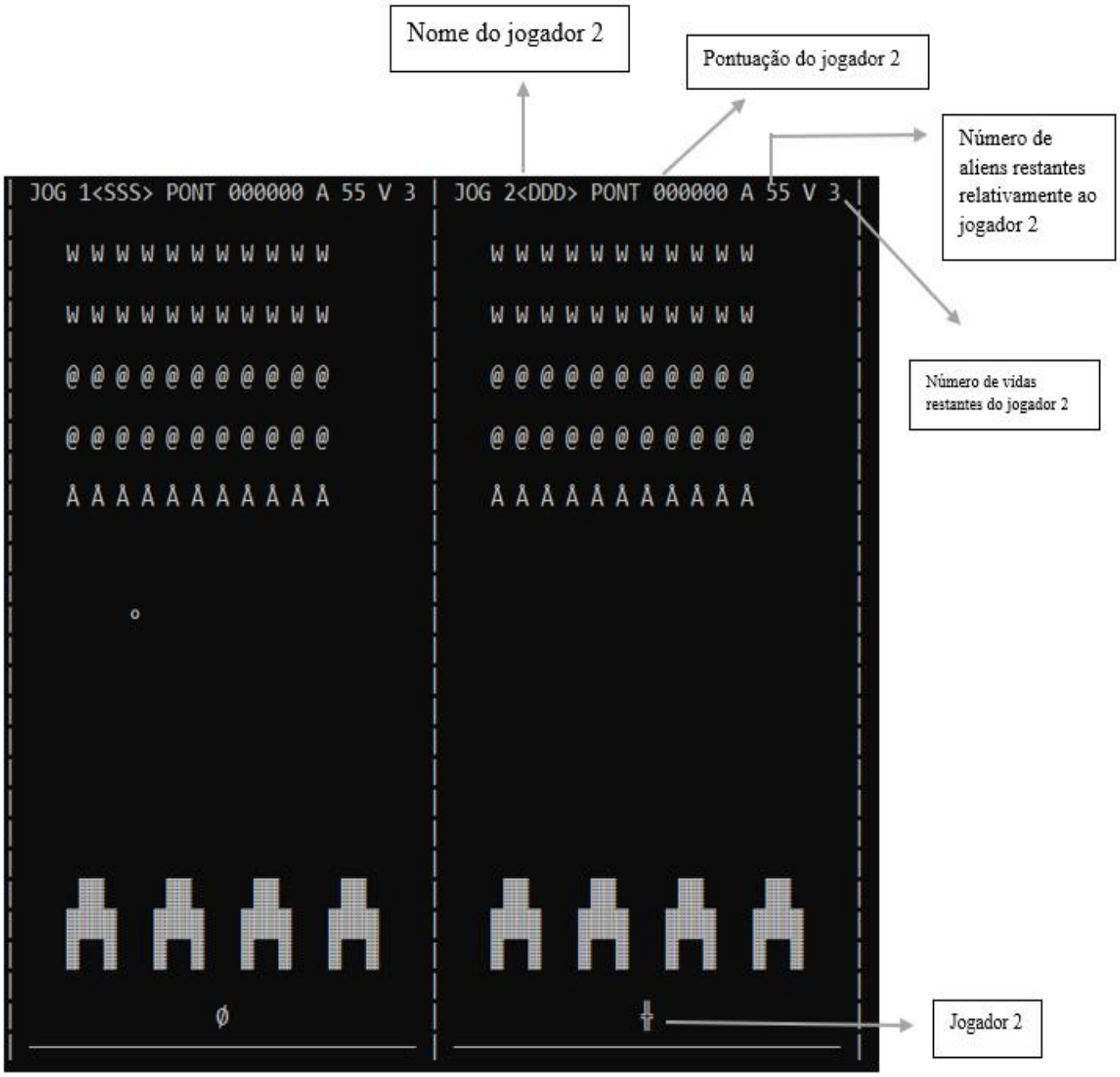
No inicio do código do jogo estão definidas todas as variáveis essenciais ao jogo (variáveis globais), nomeadamente as variáveis que foram referenciadas em cima e as variáveis do movimento dos utilizadores (seja este 1, 2 ou 3 jogadores).

b) Inicialização do Estado de Jogo

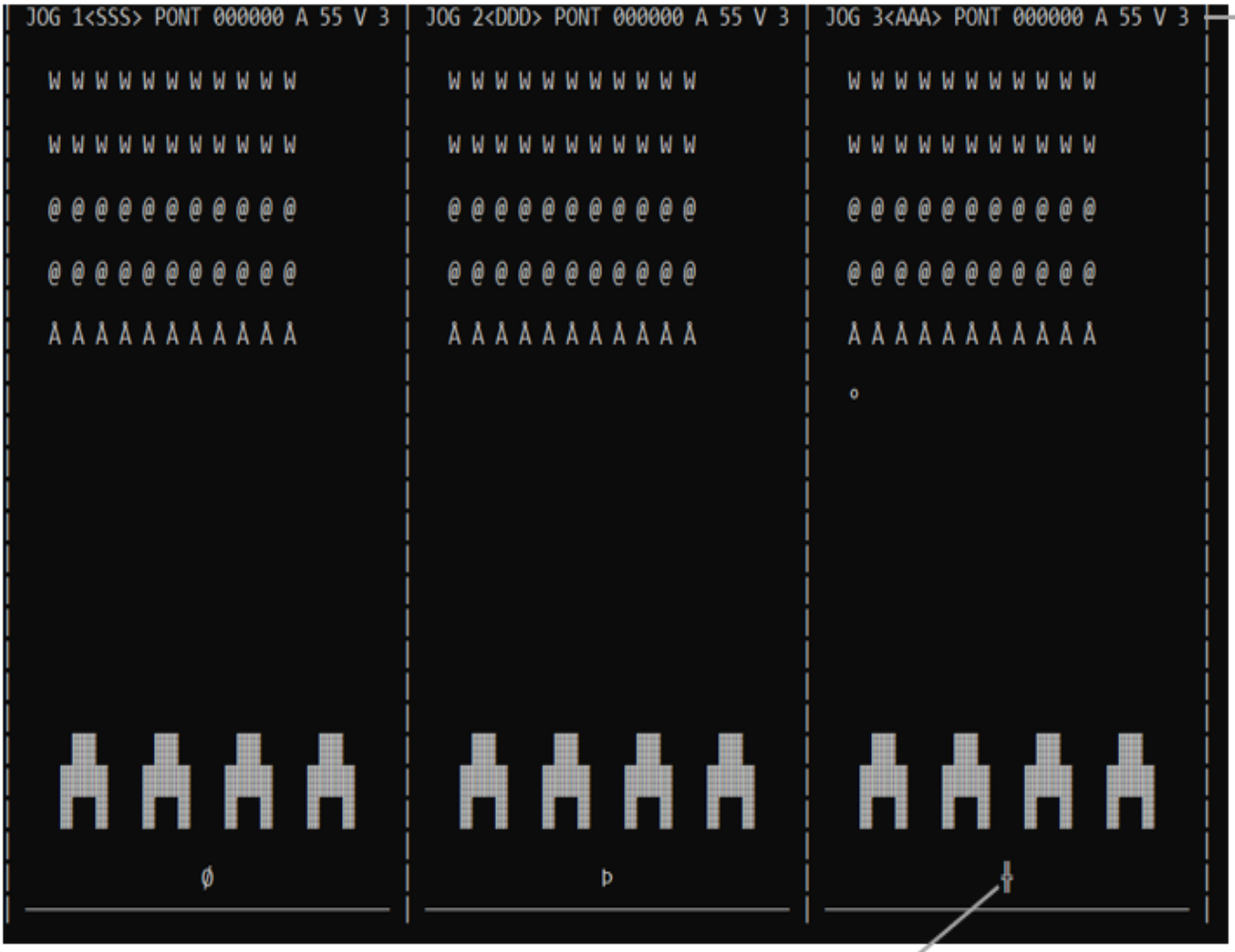
Modo Solo



Modo Desafio (2 jogadores)



Modo Desafio (3 jogadores)



Nome do Jogador 3

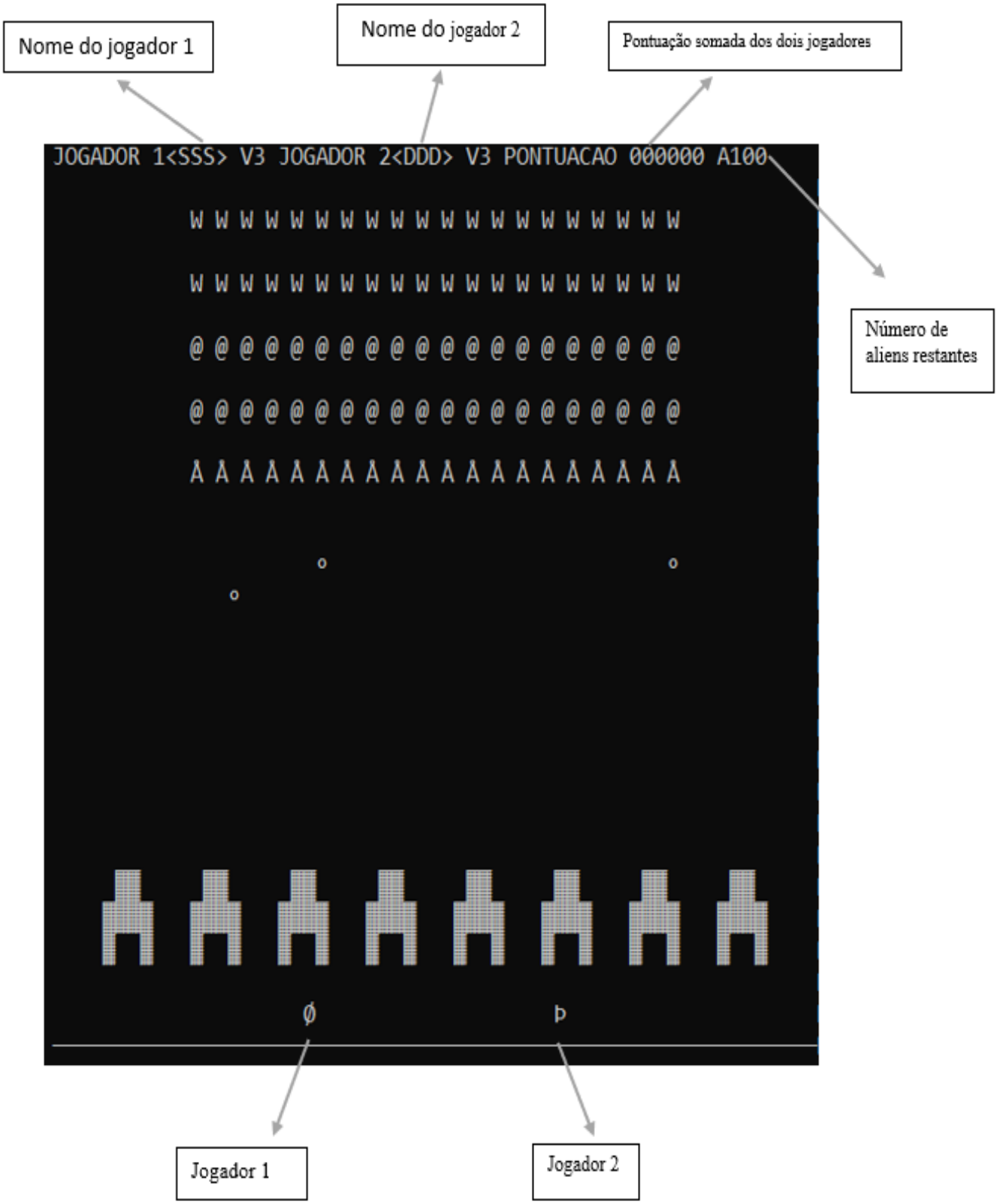
Pontuação jogador 3

Número de aliens restantes relativamente ao jogador 3

Jogador 3

Número de vidas restantes relativamente ao jogador 3

Modo Equipa/Multiplayer



Tal como podemos observar pelas imagens acima, o jogo é constituído por 4 modos de jogo.
Isto é, consoante a escolha do utilizador, o jogo irá inicializar de formas distintas.
Ambos nos modos o(s) nome(s) do(s) usuário(s), a quantidade de alliens restante, o número de vidas que cada jogador possuiu e a sua pontuação (ou a soma das pontuações no caso do Modo Equipa), são exibidos na parte superior da tela.

c) Mudança de Estado/Movimento do Jogador

No que diz respeito à dinâmica do jogo, começamos por definir todas as teclas que serão necessários para o movimento do utilizador, quer seja solo ou com mais jogadores.

Primeiramente começamos por construir uma função "procura posição jogador" em que nos consiga dizer a coluna exata onde a nave do jogador está localizada.
Sendo assim, sabemos a posição exata do jogador, visto que a linha em que o jogador está inserido é a antepenúltima (tal como podemos observar nos prints das imagens mais acima).

Relativamente ao movimento do jogador propriamente dito, construímos uma função que lê a tecla pressionada e que, consoante a direção desejada (direita ou esquerda), substitui a posição inicial da nave por um espaço e adiciona um ou retira um à coluna da matriz (move-se para a direita ou esquerda respetivamente), dependendo do rumo para o qual a nave se tenciona deslocar.

No que toca ao movimento dos alliens, numa fase inicial os alliens movem-se para direita, isto é, é criado um ciclo que incrementa uma unidade à coluna enquanto o allien mais afastado (mais perto da linha final) não chega à última coluna navegável (previamente definida).
Por conseguinte, e uma vez que já atingiram o máximo navegável para a direita, o pelotão de inimigos irá ter de descer. À semelhança do que aconteceu no movimento destes para direita, agora irá ser também incrementada uma unidade, mas nas linhas.
E assim sucessivamente (nota: Quando estes se movem para esquerda o parâmetro deixa de ser a última coluna navegável e passa a ser a primeira).

d) Processamento do Tiro

No que diz respeito ao tiro, podemos dividir em dois casos: o tiro do próprio jogador e o tiro dos inimigos.
Em relação ao tiro do jogador e sabendo já a posição atual da nave (daí a grande importância da função destinada a encontrar a posição do jogador), conseguimos construir uma função que nos imprime no ecrã o tiro do jogador. De uma forma sucinta, é criado um ciclo "for" que após cada execução, decrementa uma unidade à linha e preenche a posição anterior com um espaço, e assim sucessivamente.
Quando o tiro detetar contacto, isto é, detetar um obstáculo (que pode ser um allien, o escudo que protege o jogador ou até mesmo o final da matriz) comporta-se de maneira distinta.

Quando o tiro encontra um allien, este verifica qual é o nível deste (condição/"if"). Se for do nível 1, provoca uma explosão e substitui o espaço ocupado pelo allien por um espaço. Se for do tipo 2, provoca uma explosão e substitui este pelo allien de nível imediatamente inferior (neste caso substitui por um de nível 1). E assim sucessivamente.

Se o tiro encontrar o escudo protetor do jogador, este irá ter um comportamento semelhante ao dos alliens. Dá-se a explosão, é substituído o espaço ocupado por um escudo de nível imediatamente inferior e assim sucessivamente, até não restar escudo (previamente, os escudos são de nível 3, ou seja, são necessários 3 tiros para os desintegrarem totalmente).

Se nenhuma destas hipóteses se concretizar, é sinal que o tiro chegou ao final da matriz. Dá-se a explosão e o tiro desaparece.

Em relação ao tiro inimigo, o processo é quase o mesmo do tiro do jogador. Primeiro, definimos quem são os alliens que devem atacar. Para tal, utilizamos uma função (função "geradorAtaqueAlien") que decide quem são os alliens que vão provocar os ataques. Para tal, utilizamos a posição atual dos alliens e comparamos com a linha seguinte. Se na linha seguinte ao allien não estiver nada (estar preenchido com um espaço), esses são os alliens que vão poder atirar.

Por conseguinte, estes alliens que reúnem as condições para disparar, vão fazê-lo exatamente no processo inverso do tiro do jogador explicado a cima. Vai ser implementado um ciclo "for" que após cada execução, incrementa uma unidade à linha e preenche a posição anterior com um espaço, e assim sucessivamente.

Quando o tiro inimigo encontrar resistência (algo com que colidiu), é sinónimo de uma destas três hipóteses: Ou colidiu com o escudo que protege o jogador, ou colidiu com a nave do jogador ou chegou ao final da matriz.

Em relação à primeira situação, o processo vai ser o mesmo explicado na página anterior (o escudo irá ser substituído por um de nível imediatamente inferior até se desintegrar completamente):

No que toca à colisão com a nave do jogador, se este possuir ainda vidas extra, irá ser decrementada uma unidade a essas vidas. Caso contrário, o jogo termina e o jogador perde.

Por último, se o tiro inimigo chegou ao final da matriz, irá ser efetuada uma explosão e o tiro desaparece.

e) Gestão de Pontuação e “Hall of Fame”:

Relativamente à gestão de pontuação e à sua ordenação de forma a visualizarmos as melhores performances (“Hall of fame”), passaremos a explicar.

Primeiramente, olhemos como se processa o registo de pontuação.
Com alguma pesquisa e de modo a dinamizar o jogo, criámos uma fórmula para a pontuação, em que à medida que os alliens se movimentam em direção ao jogador (descem), esta vai progressivamente aumentando.
Toda esta pontuação pode ser acompanhada em direto pelo utilizador, no seu canto superior direito.

No que concerne ao processo de guardar a pontuação, recorremos aos ficheiros para tal.
Implementamos uma função em que importamos para o ficheiro todas as pontuações registadas no final de cada partida.

No que diz respeito a carregar o “Top 10” das classificações, tivemos de utilizar um método no qual só apareça no ecrã as 10 melhores classificações, por ordem decrescente.
Neste caso, utilizamos o método “Bubble Sort” lecionado na aula.
Organizamos as pontuações num “array”/vetor e utilizámos um ciclo “for” de modo a comparar as consecutivas posições do vetor.
Se a posição numero dois do vetor for maior que a primeira, dá se a troca e assim sucessivamente.
Nisto consiste a ordenação por “Bubble Sort”.

Neste processo, separámos a classificação dos diferentes modos, isto é, há uma tabela classificativa para os modos Solo e Desafio e outra para o modo Equipa/Multiplayer.

f) Graus de Dificuldade:

Como foi referido anteriormente, o jogo possui 3 graus de dificuldade em que o utilizador tem total controlo sob a escolha deste.
Um nível fácil, um de dificuldade intermédia e um difícil, em que neste serão testadas ao limite todas as habilidades do usuário.

Em termos estruturais, organizámos a dificuldade tendo em conta dois aspetos.
O primeiro é na velocidade dos alliens, em que quanto maior for o nível, mais rápido estes se movimentam.
O segundo relaciona-se com a probabilidade de estes disparem.
Mais uma vez, quanto mais difícil, maior a probabilidade de tiros inimigos.

g) Itens de Prémio:

Como itens de prémio, disponibilizamos a cada usuário que inicialize um novo jogo, 3 vidas adicionais.

h) Salvar e Restaurar jogo:

Como em qualquer outro jogo, não nos poderíamos esquecer de implementar uma função que guardasse o progresso do jogo e que, numa outra ocasião, pudéssemos continuar o jogo exatamente no instante onde tínhamos ficado.

Para tal, criámos um ficheiro.txt com o nome "GuardaProgresso" que fosse capaz de guardar a matriz exatamente no instante em que saímos bem como todas a variáveis globais que fossem necessárias para o carregamento deste.

Portanto, no menu inicial do jogo existe a opção "gravar jogo" em que quando executado, faça o carregamento do último jogo para o ficheiro.
Por outras palavras, quando quisermos sair do jogo e continuar este noutra ocasião (seja qualquer um dos 4 modos), basta sair na tecla "ESC" e premir a opção 3 do menu, que irá efetuar o carregamento para o ficheiro.

Para carregar o jogo do ficheiro para a opção 2 do menu ("Retomar Jogo"), o processo é semelhante ao de cima mas no sentido inverso.
Para tal, tivemos de utilizar todos os conhecimentos adquiridos nas aulas, nomeadamente a leitura de ficheiros e as funções "atoi" e "strtok", que serviram para converter "strings" em números inteiros e para dividir uma "string" em vários "tokens", de acordo com o delimitador selecionado.
À medida que percorremos os "tokens", vamos recuperando do ficheiro todos os dados/variáveis que são necessários para a execução deste.
Por fim, recuperamos a matriz (mais uma vez com o auxilio destas duas funções), conseguindo assim carregar o jogo com sucesso.

i) Final da Execução:

Como em qualquer jogo “retro”, os desfechos possíveis não são muitos e neste caso não é diferente.

Se o jogador (ou jogadores, dependendo dos modos) conseguir(em) aniquilar todos os inimigos sem morrerem, dá-se a Vitória!

Caso contrário, o jogador perde.

Nota: Nos modos Desafio e Equipa/Multiplayer, o jogo só acaba quando o último jogador morre.

5) Conclusões e Perspetivas de Desenvolvimento

Do que foi dito pode concluir-se que após tantas horas a trabalhar num projeto, não poderíamos estar mais felizes com o resultado obtido.
Trabalho, espírito de equipa e união estiveram presentes desde o primeiro dia do projeto e da qual nos orgulhamos.
Apesar disso, temos plena consciência que muitos aspetos podem ser melhorados e mais bem trabalhados, mas estamos cá para isso.
No que depender de nós, trabalho, atitude positiva e dedicação não irá faltar.