**Tarefa 1**

Exemplo de um labirinto aonde a **largura é par** e um outro aonde a **largura é ímpar** e isso faz com que **a casa dos fantasmas mude de tamanho**.

Falar:

Que o labirinto está rodeado por muros menos no túnel que fica no meio de labirinto.

1° Adicionar as paredes que rodeiam o labirinto

2° Adicionar os tuneis fazendo cálculos que permitem encontrar o centro da altura do labirinto

3° Mantemos os corredores que ficam exatamente em cima e em baixo da casa dos fantasmas e posemo-la no meio, depois tivemos que acrescentar as peças que falta à esquerda e direita da casa dos fantasmas (essas peças são as originais do labirinto e não peças novamente aleatórias).

**Tarefa 2:**

q (gera um state com labirinto)

* Encontro com Fantasma Alive: play (Move 0 U)
* Comida Pequena (+1): play (Move 0 U)
* Comida Grande (+5): play (Move 0 U)
* Encontro com Fantasma Dead: play (Move 0 U)
* Mudança de Orientação: play (Move 0 R)
* Túnel/Posição Vazia: (Move 0 R)
* Encontro com Wall: (Move 0 R)

Temos uma função que **determina que peça** esta numa determinada posição

Na função principal play fizemos de um modo que podemos **controlar qualquer jogador através do seu ID**, a seguir temos as **funções move** que se ocupa dos movimentos possíveis de serem feitos assim como o que acontece em cada caso, por exemplo encontro com comida pequena, grande, etc.

A funções **auxiliares da move esta separadas em duas partes**, **uma que se ocupa do movimento do jogador, da contagem dos pontos, etc.**

E **outra**, que se **ocupa de atualizar o labirinto sempre que necessário**, ou seja, quando o Pacman come uma comida pequena ou grande, desta forma quando movimentamos os **fantasmas eles não atualizaram o labirinto** pois eles não consomem as comidas.

Na função move também implantamos para que o Pacman abra e fecha a boca a cada movimento que ele faça.

Nesta tarefa, também foi possível fazer com que o player consiga **passar pelo túnel** e sair do outro lado.

**Tarefa 3:**

A tarefa 3 baseia-se na **compressão máximo de um labirinto** do jogo, ou seja, baseia-se na simplificação máxima dos labirintos formados, de modo a que **ocupassem o mínimo de espaço possível, sem perder qualquer tipo de informação nele presente.**

O objetivo principal desta tarefa é atingir a compressão máxima, através da conversão

de um labirinto para um conjunto de tuplos cujo primeiro valor é um inteiro e o segundo uma peça.

\* \_\_**Compactação do labirinto horizontalmente\_\_: primeira fase**

[Wall,Wall,Wall,Empty,Food Big,Food Big,Wall] comprimido seria convertido a uma

instrução do tipo Instruct [(3,Wall),(1,Empty),(2,Food Big),(1,Wall)].

**\* \_\_Compactação das Instruções Verticalmente\_\_: segunda fase**, as instruções obtidas

anteriormente são compactadas verticalmente, ou seja, são identificados os corredores com peças e posições iguais, de forma a que corredores iguais apenas constituam uma instrução, com o índice de um elemento da lista de instruções. Nos labirintos com apenas dois corredores iguais (parede de cima e de baixo), o último corredor será representado por Repeat 0 (repetição do corredor 0)

Tarefa4/5: ghci Main.hs – carregar tecla DOWN e ir para o TUNEL!

* Abrir e Fechar Boca
* Modo Mega
* Movimentação Fantasmas
* Morte

A tarefa 4 baseia-se na **implementação da reação à passagem do tempo** e para além disso, na implementação de movimentação dos jogadores numa interação. Desta forma, esta tarefa é divida em duas partes:

**\* \_\_Parte 1\_\_:** integra os tipos e as funções que advém da 1ª fase numa biblioteca

denominada NCurses

**\* \_\_Parte 2\_\_:** utiliza o estado e informação vinda da primeira parte para avançar o jogo numa iteração, fazendo, desta forma, a movimentação de todos os jogadores

= Objetivos

Os objetivos desta tarefa passam pelo cálculo do efeito da passagem de um instante de tempo num determinado estado de jogo. Através de um /data/ Manager, no final desta tarefa, pretende-se ser possível controlar a passagem do tempo definido para efetuar uma nova jogada, que por si só, deve movimentar os jogadores, tendo em conta a sua velocidade

e o respetivo estado. Para este efeito, devem ser introduzidos na tarefa 2 comportamentos afetos ao estado Mega, ao próprio Pacman (abrir e fechar a boca) e à velocidade dos jogadores.

= Estratégia de Implementação

Numa primeira instância, foram introduzidas no ficheiro Main.hs algumas funções que

permitissem a passagem do tempo a cada jogada, tal como, a "nextFrame", a "updateTime" e a "resetTimer". Para além disso, foi concretizada uma função "updateControlledPlayer" que através das setas do teclado, permite mudar a direção do jogador Pacman.

Na segunda parte desta tarefa, foi implementada a função **"passTime".** Esta função permitiu não só a **implementação da função "ghostPlay" tarefa 5, como também o bot tarefa 6**. A estratégia utilizada para esta função passou pela utilização de uma função auxiliar que cobrisse todos os casos obrigatórios nesta tarefa: a **constante movimentação do Pacman na direção em que se encontra** (até ser pressionada uma Key numa direção diferente), **as diferentes velocidades dos jogadores** (nomeadamente os fantasmas) e a implementação da tarefa 5, com a atribuição de cada Play da lista originada na função "ghostPlay" ao respetivo fantasma. A função em formato de comentário permite o funcionamento do bot num jogo Pacman.

**Tarefa 5:**

= Objetivos

Para a concretização desta tarefa, pretende-se que o comportamento normal, no qual o **fantasma se encontra em modo /Alive/, seja perseguir o Pacman**. Caso o fantasma esteja em **modo /Dead/, este deve fugir ao Pacman em modo Mega** até que voltar a estar em modo /Alive/. **Adicionalmente, sempre que os fantasmas passem para o modo /Dead/, estes devem obter uma orientação oposta à que tinha no momento anterior**. Para isto, é necessária a implementação da função "ghostPlay" que recebe um estado de jogo e devolve um conjunto de jogadas, uma para cada fantasma, com a melhor alternativa de reação ao Pacman.

= Estratégia de Implementação

As estratégias utilizadas nesta tarefa, são as seguintes:

\* \_\_**Fantasmas (/Alive/) em perseguição do Pacman**\_\_: quando um fantasma se **deparar com um muro**, este irá **determinar a posição do Pacman**, sendo que o fantasma **irá na direção mais próxima do Pacman**. Por exemplo: o fantasma encontra um muro na posição imediatamente acima da sua se o Pacman se encontrar abaixo, o fantasma seguirá essa direção, utilizando esta estratégia para as outras direções. Para além desta estratégia, colocou-se a **hipótese do fantasma se encontrar entre duas paredes**, tendo sido contemplada esta opção no código abaixo. Neste caso, o fantasma **muda de direção antes de ficar encurralado**.

\* \_\_**Fantasmas (/Dead/) fogem do Pacman (Mega)**\_\_: quando o **fantasma se depara com uma parede, este move-se para a sua direita**, circulando a área onde se encontra no **sentido dos ponteiros de relógico**, tal como mencionado acima. Esta estratégia foi conseguida através da função "scatterMode” que sempre que muda a direção dos fantasmas como descrito.

= Conclusão

Em geral, conclui-se que esta tarefa foi implementada com sucesso e que os objetivos foram todos cumpridos com excelência. Mais uma vez a qualidade do código poderia melhorar, mas o foco desta tarefa foi implementar duas estratégias que **complicassem o jogo e, ao mesmo tempo, que permitissem ao jogador Pacman criar uma estratégia para tentar prever os movimentos dos fantasmas e, desta forma, ganhar o jogo.**

**Tarefa 6:**

Dividimos basimente em 3 partes principais:

- Pacman em modo mega persegue o fantasma mais próximo.

- Pacman no modo normal vai na direção da comida mais próxima

- Mas, se em algum momento o Pacman tiver em perigo, ou seja, a uma distância curta do Pacman que nos posemos de 5, ele ira fugir do fantasma.

\* \_\_**Pacman em modo Mega persegue os fantasmas**\_\_: nesta ação, o Pacman **determina as coordenadas do fantasma mais próximo** e **vai em direção a ele**. Se a **distância horizontal entre os dois for menor do que a distância vertical, ele move-se para a esquerda ou para a direita**, consoante a direção do fantasma. No caso contrário, o Pacman move-se para cima ou para baixo, consoante a direção em que o fantasma se encontre. Para além disso, se o **Pacman se encontrar perto de um túnel (travessa o túnel)** e um dos fantasmas estiver perto do outro túnel, o primeiro irá se deslocar em direção ao segundo. Importa ainda referir que se o Pacman encontrar uma **parede**, ele irá se **desviar** do mesmo.

\* \_\_**Pacman em modo Normal foge dos fantasmas**\_\_: nesta ação, quando o Pacman se encontra em perigo, ou seja, perto de um dos fantasmas (5 peças vertical ou horizontalmente), o **Pacman irá mudar para a direção oposta do fantasma**.

Tambem tomamos cuidado com os **casos de ter uma parede na direção que o Pacman foge, de ele desviar a parede**

\* \_\_**Pacman em modo Normal em direção à comida**\_\_: nesta ação, o Pacman vai determinar as coordenadas da comida mais próxima, e irá na sua direção, usando a **mesma estratégia utilizada na ação "Pacman em modo Mega persegue o fantasma"** descrita acima.

= Conclusão

Em conclusão, apesar de existirem muitas outras estratégias na implementação de um

bot para este jogo, foi possível introduzir algumas que se apresentam como essenciais

para uma maior fluidez do jogo. Desta forma, os objetivos desta tarefa foram atingidos

com sucesso.