

Trabalho 1 - Relatório

Introdução

O seguinte trabalho, desenvolvido pelo aluno Henry Shinji Suzukawa, nº USP 9771504, objetivou o desenvolvimento de um jogo baseado no “O Mundo de Wumpus”, criado por Gregory Yob em 1972/1973.

Neste trabalho o jogador controla um personagem que dentro de uma caverna deve encontrar o tesouro e sair dela ileso, para isso é necessário desviar de abismos e do terrível monstro Wumpus. Munido com apenas uma flecha, o personagem pode atacar esse monstro, sentindo seu cheiro quando se aproxima dele; semelhante ocorre com os abismos, onde o personagem sente brisas ao se aproximar desses. O personagem, porém, tem um limite de passos, que se extrapolado encerra o jogo.

Descrição

O desenvolvimento desse programa foi fundado na linguagem de programação Python 2.x, versão 2.7.11+ na arquitetura x64, utilizando o editor de texto nativo do sistema operacional Ubuntu (Gedit).

De modo geral, o código^[1] é dividido em 3 partes: Interface por Comandos, Interface textual e Controle Geral. As duas primeiras controlam a interação jogador/personagem, sendo que a terceira cuida das interações do personagem com o próprio jogo, o arquivo README^[2] contém breves explicações sobre cada função.

^[1] Disponível em:

1. <https://github.com/HSuzu/ICC-2016/blob/master/Trabalhos/Trabalho-01/main.py>
2. main.zip/main/main.py

^[2] Disponível em:

1. <https://github.com/HSuzu/ICC-2016/tree/master/Trabalhos/Trabalho-0>
2. main.zip/main/README

Tutorial

Para executar o jogo o usuário deve instalar Python 2.x [*Importante*: Python 2.x é nativo em muitas distribuições Linux, algumas porém utilizam Python 3.x - como Fedora 23 -, que é incompatível com o jogo], para isso vá ao endereço <https://www.python.org/downloads/> e baixe a última versão 2.x (no caso de distribuições Linux consulte o guia para instalá-lo via terminal: <https://wiki.python.org/moin/BeginnersGuide/Download>). Caso tenha dúvidas de como proceder após isso consulte o manual feito por terceiros: <http://wiki.icmc.usp.br/images/8/82/InstalacaoPython.pdf>.

Com o interpretador já instalado, execute no terminal [*Importante*: lembre-se de trocar **[caminho_para_*]** pelo devido caminho]:

- python **[caminho_para_o_programa]**/main.py, caso esteja em uma distribuição Linux
- **[caminho_para_o_python]**/python.exe **[caminho_para_o_programa]**/main.py, caso esteja utilizando Windows

Ao executar esse comando será questionado se o usuário quer utilizar a Interface Textual ou não, caso sim, digite “s” e aperte “[ENTER]” [*Importante*: qualquer outro caractere ou frase será entendido como não], após isso será questionado se o usuário quer executar a Inteligência Artificial (a resposta segue a mesma formatação do questionamento anterior).

Caso o usuário deseje não executar a Interface Textual, aparecerá a seguinte tela:

Posição: (0, 0) Score: 0 Vida: 0 Flechas: 1 Mov. rest: 80 Mapa: 8x10

Digite um comando:

Os comandos válidos são aqueles formatados como “[ação] para [direção]” [*Importante*: uma exceção a essa formatação é o comando “sair”, que encerra o jogo].

[ação] pode assumir os seguintes valores

- **mova**: move o personagem a uma posição dependendo da direção [*Importante*: a direção do personagem é a mesma do usuário, por exemplo, se o personagem está na posição (0,0), o comando “mova para direita” moverá o personagem para a posição (1,0)].
- **atire**: atira a flecha de acordo com a direção (essa atinge apenas a posição indicada pela direção), caso a flecha atinja o monstro o usuário verá a informação “Você matou Wumpus!!!”, caso não “Você não matou nada...”.
- **olhe**: retorna a sensação que há na casa indicada pela direção [*Importante*: caso haja um abismo ou o monstro na casa indicada, o personagem morrerá, perceba que o mesmo não ocorre com o tesouro]

[direção], da mesma forma, pode ser (tomando (x, y) como a posição do personagem):

- **direita**: indica a posição (x+1, y)
- **esquerda**: indica a posição (x-1, y)
- **cima**: indica a posição (x, y-1)
- **baixo**: indica a posição (x, y+1)

Por exemplo, são comandos válidos:

- “olhe para cima”
- “mova para baixo”
- “atire para direita”

[*Importante*: a adição de artigos “o”, “a”, etc. ou qualquer outro caractere fará com que a ação se torne inválida, por exemplo “atire para a direita” é um comando inválido]

Após indicar um comando aparecerá as seguintes informações:

[Retorno da Ação]

Você está sentindo: **[Informação da Posição]**

[Retorno da Ação]: indica o que a ação resultou, os possíveis valores são:

- Você não pode mais se movimentar: o usuário excedeu o limite de movimento
- Comando inválido!: o usuário não formatou corretamente o comando
- Você Morreu...: o personagem morreu
- Você está sem flechas...: o usuário excedeu o limite de flechas (uma flecha)
- Posição Inválida: o usuário tentou acessar uma posição inválida

- [Informação da Posição]:** indica o que o personagem está sentindo na forma “**[tipo]** **[intensidade]**”. **[tipo]** pode ser brisa ou cheiro, enquanto intensidade depende da distância do personagem à origem da sensação: caso a distância seja uma casa, **[intensidade]** será **forte**; caso esteja a duas casas, **médio**; e à três casas, **fraco**. **[Importante]:** caso não exista nenhuma sensação, **[Informação da Posição]** indicará “**nada**”

Redimensione seu terminal de modo que a linha abaixo fique contínua (80 caracteres).
Pressione [ENTER] quando estiver pronto

Após isso, o usuário será recebido com a seguinte tela:

Essa tela é composta do título (primeira linha), informações do jogo (terceira linha), mapa (sexta linha até no máximo a vigésima primeira) [**Importante:** caso o mapa exceda 20x15

será impresso um mapa com essas dimensões, porém com flechas indicativas (como acima)] e informações do comando (penúltima linha).

Seu funcionamento é semelhante à interface por comando, sendo que às únicas diferenças são:

- melhor visualização do mapa e da posição do personagem (indicado por x)
- adição do comando **cursor**

O comando **cursor** permite que o usuário reveja as sensações de alguma posição no mapa (indicado com o caractere “[”), para movê-lo, diferente do personagem, basta utilizar as setas do teclado, e para sair, basta pressionar “s”.

Inteligencia Artificial

A IA não tem nenhum privilégio maior do que o usuário, acessando apenas variáveis globais (exceto o mapa) que informam sobre o jogo (por exemplo a variável `knownMap`, que guarda as sensações já recebidas, ou a variável `wumpus`, que informa se o monstro já foi morto). Para garantir isso, a única interação da IA com o restante do programa é feito por meio da variável `command`, a mesma que o usuário modifica ao executar um comando.

As escolhas de comando são feitas a partir de “notas” para os movimentos. Caso o movimento seja impossível ou resulte em uma morte provável é dado “nota” 0, caso o movimento ocasione no retorno a uma casa já conhecida é dado “nota” 1. As notas 2+ são dadas a partir de preferências por movimentos. No fim, é calculado qual movimento apresenta maior “nota”, sendo esta a executada.

O objetivo máximo da IA não é conseguir o maior número de pontos, mas sim não morrer. Por isso não foi implementado nenhuma função que sobreponha um possível perigo, dessa forma, caso não haja nenhum movimento “seguro”, a IA se moverá em ciclos até que o personagem se canse.

Outras Informações

1. Comandos não são CaSe Sensitive
2. A matriz teste está disponível em
 - a. <https://github.com/HSuzu/ICC-2016/blob/master/Trabalhos/Trabalho-01/matriz.txt>
 - b. `main.zip/main/matriz.txt`
3. Não renomeie o arquivo “main.py”, caso seja necessário fazer isso, deve-se modificar a linha 652 do arquivo “main.py” para que o argumento da função `replace()` contenha o novo nome do arquivo.
4. Caracteres especiais podem não ser visualizados corretamente no Windows, por isso é recomendado utilizar alguma distribuição Linux, de preferencia Ubuntu 14.04+
5. Este relatório está disponível, assim como todos os outros arquivos, em
 - a. <https://github.com/HSuzu/ICC-2016/blob/master/Trabalhos/Trabalho-01/>
 - b. `main.zip/main/`