Especificação do Trabalho Prático

Descrição e Objetivo

Esse trabalho em grupo consiste na implementação de um simples *Jogo da Memória online* utilizando a API de *sockets* estudada na disciplina. Para isso, o grupo deverá partir de uma implementação de referência de um Jogo da Memória local (i.e., jogado em um único computador) e deverá modificá-la/estendê-la, gerando uma nova versão em cada jogador interage de um computador diferente. Mais detalhes da implementação de referência e da implementação pedida nesse trabalho podem ser encontrados nas seções abaixo.

Implementação de Referência

Para auxiliar na tarefa de implementação desse trabalho, **está sendo disponibilizada uma implementação de referência** para uma versão local do Jogo da Memória. Tal implementação pode ser encontrada no arquivo JogoDaMemoria.Py.

A implementação de referência foi escrita em Python 2. Mais especificamente, o código foi testado na versão 2.7. Apenas módulos da biblioteca padrão do Python 2 foram utilizados.

A implementação de referência não recebe nenhum argumento. Para executá-la, basta utilizar o seguinte comando (ou similar, dependendo das configurações do seu sistema):

```
# python JogoDaMemoria.py
```

Embora o programa não receba parâmetros, certas constantes que controlam o comportamento do programa podem ser mudadas nas linhas 223 a 227 do código fonte. São elas:

- *dim*: Dimensão do tabuleiro para a partida. Controla a dificuldade do jogo. Pode ser qualquer número par menor que 10.
- *nJogadores*: número de jogadores na partida.

Ao ser executado, o programa gera um tabuleiro quadrado, contendo *dim* X *dim* casas, identificadas por coordenadas *(i, j)*, com *0 <= i, j < dim*. Cada casa contém uma peça com um valor numérico de 0 a *dim - 1*. Inicialmente, todas as peças encontram-se fechadas, isto é, com seus valores escondidos.

Em uma execução típica, o programa deve exibir inicialmente a seguinte interface texto:

```
     0  1  2  3
-----------------
 0 |  ?  ?  ?  ?
 1 |  ?  ?  ?  ?
 2 |  ?  ?  ?  ?
 3 |  ?  ?  ?  ?

Placar:
---------------------
Jogador 1:  0
Jogador 2:  0



Vez do Jogador 1.

Especifique uma peca:
```

Na parte superior, é exibido o estado atual do tabuleiro. O caractere '?' indica que a posição correspondente possui uma peça que se encontra fechada. O jogo começa com uma oportunidade para o Jogador 1, que deve especificar as coordenadas de duas peças do tabuleiro, uma de cada vez. A especificação é feita no formato "i j", onde i e j representam, respectivamente, o número da linha e da coluna selecionadas. Ao selecionar uma peça, o programa exibe o valor na posição correspondente do tabuleiro. Por exemplo, ao especificar as coordenadas "1 2" (sem aspas):

```
       0  1  2  3
-----------------
 0 |  ?  ?  ?  ?
 1 |  ?  ?  1  ?
 2 |  ?  ?  ?  ?
 3 |  ?  ?  ?  ?

Placar:
---------------------
Jogador 1:  0
Jogador 2:  0


Vez do Jogador 1.

Especifique uma peca:
```

Ao receber coordenadas, o programa realiza uma série de verificações (por exemplo, se as coordenadas são válidas e se a peça correspondente já não está aberta). À medida que os jogadores acertam pares corretos (i.e., de peças de valores iguais), o programa atualiza o tabuleiro, substituindo o valor numérico das peças dos pares já encontrados pelo caractere '-'. Antes de cada nova jogada, o estado atualizado do tabuleiro é exibido aos jogadores:

```
       0  1  2  3
-----------------
 0 |  -  ?  ?  ?
 1 |  ?  ?  -  ?
 2 |  ?  ?  ?  ?
 3 |  ?  ?  ?  ?

Placar:
---------------------
Jogador 1:  1
Jogador 2:  0


Vez do Jogador 1.

Especifique uma peca:
```

Sempre que um jogador erra, a vez passa para o próximo. Ao encontrar um par correto, o jogador ganha a oportunidade de jogar novamente. O jogo termina quando não há mais peças fechadas. O vencedor é o jogador que encontrou o maior número de pares corretos. O programa exibe essa informação na tela.

Até a linha 218, a implementação de referência apresenta uma série de funções simples de manipulação do tabuleiro, do placar e de impressão de informações na tela. Tais funções muito provavelmente serão aproveitadas na implementação do trabalho. Há comentários explicando o propósito de cada função, fazendo com que sua compreensão não seja difícil. A partir da linha 232, o código diz respeito ao programa principal que realiza a interação com o usuário. Esse, provavelmente, será o ponto mais profundamente alterado pelos grupos.

O tabuleiro é representado por uma lista bidimensional (i.e., uma lista de listas) em que cada posição *(i, j)* guarda um dos seguintes valores:

- Valor inteiro positivo: valor da peça correspondente, que está aberta (i.e., seu valor está atualmente sendo exibido).
- Valor inteiro negativo: em módulo, também indica o valor da peça correspondente, mas ela está atualmente fechada.
- Caractere '-': indica que a peça foi removida (porque já foi corretamente pareada com outra peça por algum jogador).

O placar é simplesmente uma lista indexada pelo identificador de cada jogador.

Para efeito desse trabalho, os grupos são livres para escolher a linguagem de programação mais adequada ao desenvolvimento das suas implementações. A linguagem escolhida pode, inclusive, ser diferente de Python. Nesse caso, no entanto, **o grupo se responsabilizará por reimplementar as demais funcionalidades da implementação de referência na linguagem escolhida.**

Versão Online Distribuída

A versão *online* distribuída a ser implementada pelo grupo deve possuir as seguintes características:

- Há dois programas: um cliente e um servidor.
- O programa servidor é responsável por esperar pela conexão dos clientes (jogadores) e pelo controle da lógica do jogo.
- O programa cliente é responsável por se conectar ao servidor, receber informações que devem ser exibidas para o jogador e enviar comandos do jogador para o servidor quando adequado.
- Toda a comunicação deve ser realizada através de *sockets* TCP.

Nas seções subsequentes, são dados mais detalhes a respeito do cliente e do servidor.

Programa Cliente

Ao ser iniciado, o programa cliente deverá receber as seguintes informações do jogador:

- Endereço IP ou nome do *host* que executa o programa servidor.
- Número de porta na qual o programa servidor escuta.

A maneira pela qual essas informações são obtidas não é importante (elas podem ser lidas, por exemplo, da linha de comando).

Uma vez de posse dessas informações, o programa cliente deverá abrir uma conexão TCP para o servidor. Uma vez aberta a conexão, o programa cliente deverá receber do servidor um identificador numérico do seu jogador (e.g., 0, 1, 2, ...) dentro da partida que se iniciará. Repare que quando o cliente abre a conexão com o servidor, é possível que ainda não haja jogadores suficientes conectados. Assim, é necessário que o cliente aguarde que o servidor avise sobre o início do jogo. Enquanto isso não acontece, o programa cliente deverá exibir uma mensagem similar a "Aguardando outros jogadores..."

Uma vez iniciada a partida, o programa cliente receberá informações do servidor e imprimirá os dados do jogo na tela. Em particular, o servidor deverá enviar os seguintes dados quando pertinente:

- Aviso do jogador da vez.
- Aviso da jogada do jogador atual.
- Atualização do estado do tabuleiro.
- Aviso de que o jogador da vez acertou um par.
- Atualização do placar.

Por sua vez, quando for a vez do jogador local, o programa cliente deverá solicitar as coordenadas das peças a serem abertas do jogador humano e deverá enviá-las ao servidor.

Ao final da partida, o programa cliente deverá encerrar sua execução.

Programa Servidor

O programa servidor recebe dois parâmetros como entrada: o tamanho do tabuleiro (valor par menor que 10) e o número de jogadores (ao menos dois). A maneira pela qual essas informações são lidas não é importante (sugere-se o uso de argumentos de linha de comando).

Ao ser executado, o programa servidor abre um *socket* TCP em modo de escuta e aguarda a conexão de jogadores. É importante notar que **o servidor precisará manipular um número de conexões simultâneas igual ao número de jogadores**. Ao receber a conexão de um novo jogador, o servidor deverá atribuir um identificador numérico único para aquele jogador durante a partida. Os identificadores devem ser valores inteiros sequenciais, começando de 0 para o primeiro jogador. Quando o número de jogadores conectados alcança o número especificado de jogadores para a partida, o servidor deverá enviar uma mensagem de início de jogo para todos os clientes conectados. Tal mensagem deverá ser acompanhada de informações básicas como o estado inicial do tabuleiro e do placar.

Além da mensagem de início de jogo, o servidor deverá enviar também uma mensagem informando a vez do primeiro jogador (sempre o jogador de identificador 0). Note que essa mensagem deve ser enviada a todos os jogadores (não só o de identificador 0) para que todos saibam de quem é a vez agora. Nesse ponto, o servidor deverá aguardar uma resposta do cliente do Jogador 0, contendo sua jogada (i.e., as peças que o jogador deseja abrir). De posse dessa informação, o servidor deverá:

- Enviar para os demais jogadores a jogada do Jogador da vez.
- Verificar se o par selecionado é correto (i.e., se ambas as peças possuem mesmo valor).
    - Se sim, atualizar o placar, enviar o novo placar e o novo estado do tabuleiro para todos os jogadores.
- Atualizar, se necessário, o jogador da vez e enviar essa informação para todos os jogadores.

Quando não houver mais peças fechadas, o servidor deverá verificar qual foi o vencedor e informá-lo em uma mensagem de final de partida enviada a todos os jogadores conectados.

Note que, ao final da partida, o programa servidor deverá esperar por novas conexões de novos jogadores para iniciar uma nova partida.

Outros Aspectos de Implementação

Essa especificação é propositalmente omissa em certos detalhes da implementação pedida. Por exemplo, não estão definidos aqui os **formatos das mensagens utilizados pelo protocolo da aplicação** proposta. Cabe a cada grupo definir formatos de mensagem que sejam adequados à aplicação proposta. Além disso, nas seções anteriores citou-se várias vezes o *estado atual do tabuleiro* enviado em várias ocasiões do servidor para os clientes, mas não foram especificadas exatamente as informações contidas nesse estado. Cada grupo tem liberdade para enviar as informações que julgue necessárias ao correto funcionamento da aplicação. O uso da criatividade para incorporar características extras contará positivamente na nota. Qualquer parte do enunciado que dê margem a ambiguidade deve ser interpretada pelo grupo de forma independente (usando o bom senso, buscando melhorar características de desempenho ou facilidade de interação com o usuário, etc.), mas as escolhas e interpretações feitas devem ser justificadas/descritas no relatório.

Relatório

Além da implementação, cada grupo deverá preparar um relatório documentando o desenvolvimento do trabalho realizado. O relatório deverá conter:

- uma breve documentação do código desenvolvido (por exemplo, uma lista das funções/classes/módulos usados e seu propósito);
- uma breve descrição do modo de uso dos programas cliente e servidor (por exemplo, quais argumentos de linhas de comando devem ser usados);
- (caso o grupo tenha optado por uma linguagem de programação diferente de Python) a linguagem utilizada e instruções de compilação/execução;
- observações quanto a decisões e melhorias feitas a partir da interpretação do enunciado,
- política de divisão de tarefas ou de colaboração entre os membros do grupo, e

- quaisquer outras informações que o grupo julgue pertinentes para o processo de avaliação.

Requisitos e Restrições

Embora a parte de implementação do trabalho possa ser feita em qualquer linguagem, algumas restrições devem ser observadas:

- O código deverá ser de autoria dos membros do grupo, sem que se recorra a trechos de códigos de terceiros ou bibliotecas que implementem total ou parcialmente as funcionalidades aqui listadas (funções/métodos de manipulações básicas da linguagem escolhida, bem como de sua API de *sockets* podem ser utilizados).
- O código final deve ser compilável/executável sem a necessidade de bibliotecas, *frameworks* ou ferramentas pagas.

O relatório é de formato livre, sem limites inferiores ou superiores de páginas. Apenas como um guia geral, 3 páginas devem ser suficientes (embora não necessárias) para contemplar todos os itens listados nessa especificação.

Critério de Avaliação

Cada trabalho receberá uma nota base variando de 0 a 10. Esta pontuação será dividida nas seguintes proporções:

1. Até 3,5 pontos para a implementação do programa cliente
2. Até 4,5 pontos para a implementação do programa servidor
3. Até 2,0 pontos para o relatório.

Entrega

A data limite para a entrega do trabalho é dia 9 de Maio (via Classroom). Os grupos devem ficar atentos a potenciais mudanças nessa data ao longo do período. A entrega deve incluir:

- Relatório, com uma folha de rosto contendo a Lista dos integrantes do grupo.
- Código fonte da implementação.
- Instruções de compilação/execução/uso da implementação (se pertinente).

Uma vez entregue o trabalho, não **serão aceitas alterações** (nem inclusões, nem remoções) na lista de integrantes do grupo em nenhuma hipótese. Por isso, sugere-se atenção no momento do envio da mensagem para que a lista contenha todos os integrantes do grupo.