GameHouse

Objetivo:

A máquina GameHouse tem como objetivo gerenciar a operação de uma casa de jogos digitais, onde clientes usam dispositivos (como computadores ou consoles) para jogar jogos previamente instalados. O sistema organiza os recursos físicos e lógicos disponíveis, administra sessões de uso e de fila, controle de crédito, e garante o funcionamento correto das regras operacionais da GameHouse.

Descrição Geral:

Os principais elementos necessários para o funcionamento da máquina abstrata são:

- Tipos de dispositivos (PS4, XBOX, Switch… etc);
- Dispositivos físicos do estabelecimento;
- Jogos digitais compatíveis com determinados tipos de dispositivos;
- Clientes do estabelecimento GameHouse;
- Sessões de jogos entre clientes e dispositivos;
- Créditos que limitam o uso de sessões.

Regras de Negócio:

1. Um cliente precisa ter créditos disponíveis para iniciar uma sessão.
2. Um dispositivo só pode estar em uso por um cliente por vez.
3. Um jogo só pode ser iniciado em um dispositivo no qual esteja instalado e seja compatível.
4. Cada cliente pode ter, no máximo, 3 créditos.
5. O número de dispositivos cadastrados não pode ultrapassar um limite máximo.
6. Um tipo de dispositivo ou jogo só pode ser removido se não estiver em uso.
7. O status do cliente e do dispositivo é atualizado automaticamente ao iniciar ou encerrar uma sessão, bem como ao entrar ou sair de uma fila de espera.
8. Não é permitido iniciar sessão com cliente ou dispositivo já envolvidos em outra.
9. Dispositivos em manutenção ou em uso não podem ser removidos.
10. Sessões só podem ser encerradas por clientes ativos.
11. Um cliente não pode entrar na fila se um dispositivo não possui jogos instalados ou se o seu status é “manutencao”

Operações:

Gerenciamento de Tipos:

- add_tipo_dispositivo(td): Adiciona um novo tipo de dispositivo.
- remove_tipo_dispositivo(td): Remove um tipo de dispositivo não referenciado por dispositivos ou jogos.

- **Gerenciamento de Dispositivos:**

    - add_dispositivo(dd, td): Adiciona um novo dispositivo de tipo especificado.
    - remove_dispositivo(dd): Remove um dispositivo que não esteja em uso.
    - set_status_dispositivo(dd, ss): Altera o status de um dispositivo.

- **Gerenciamento de Jogos:**

    - add_jogo(jj, td): Cadastra um novo jogo compatível com um tipo de dispositivo.
    - remove_jogo(jj): Cadastra um novo jogo compatível com um tipo de dispositivo.
    - instalar_jogo(dd, jj): Instala um jogo em um dispositivo, desde que compatíveis.
    - desinstalar_jogo(dd,jj): Desinstala um jogo em um dispositivo, desde que ele não esteja sendo jogado ou solicitado por alguém

- **Gerenciamento de Clientes:**

    - add_cliente(cc): Registra um novo cliente, com status inicial "inativo" e 0 créditos.
    - banir_cliente(cc): Remove um cliente que não esteja em sessão ativa.
    - comprar_creditos(cc, qtd): Acrescenta créditos a um cliente (até o máximo de 3).

- **Gerenciamento de Sessões:**

    - entrar_fila_dispositivo(cc, dd, jj): Entra na fila de espera para acessar um dispositivo específico com um determinado jogo.
    - sair_fila_dispositivo(cc, dd): Retira um cliente da lista de espera para um determinado dispositivo e o jogo solicitado.
    - iniciar_sessao_fila(cc, dd, jj): Inicia uma sessão de jogo entre um cliente e um dispositivo, retirando um crédito, de acordo com a fila de espera.
    - encerrar_sessao(cc): Encerra a sessão de um cliente, liberando o dispositivo e atualizando os estados.

- **Consultas:**

    - pp ← posicao_fila(cc,dd): Retorna a posição de um cliente na fila de determinado dispositivo.

- **Iterações:**

    - init_disp_disponiveis: Inicia ou reseta o processo de iteração de dispositivos disponíveis.
    - more <-- has_more_disp_disponiveis: Verifica se há mais dispositivos disponíveis na coleção para serem lidos.
    - dd <-- get_next_disp_disponivel: Obtém o próximo dispositivos disponível.
    - init_jogos_por_tipo: Inicia ou reseta o processo de iteração de jogos por tipo.
    - more <-- has_more_jogos_por_tipo(td): Verifica se há mais jogos por tipo na coleção para serem lidos.
    - jj <-- get_next_jogo_por_tipo(td): Obtém o próximo jogos por tipo.
    - init_jogos_dispositivo: Inicia ou reseta o processo de iteração de jogos por dispositivo.
    - more <-- has_more_jogos_dispositivo(dd): Verifica se há mais jogos por dispositivo na coleção para serem lidos.
    - jj <-- get_next_jogo_dispositivo(dd): Obtém o próximo jogos por dispositivo.
    - init_clientes_info: Inicia ou reseta o processo de iteração de informações de clientes.
    - more <-- has_more_clientes_info: Verifica se há mais informações de clientes na coleção para serem lidos.
    - cc, ss, cr <-- get_next_cliente_info: Obtém o próximo informações de clientes.
    - init_tipos_disponiveis: Inicia ou reseta o processo de iteração de tipos de dispositivo disponíveis.
    - more <-- has_more_tipos_disponiveis: Verifica se há mais tipos de dispositivos disponíveis na coleção para serem lidos.
    - td <-- get_next_tipo_dispositivo: Obtém o próximo tipo de dispositivo.
    - init_devices_info: Inicia ou reseta o processo de iteração de conjunto de informações de dispositivos.
    - more <-- has_more_devices_info: Verifica se há mais conjunto de informações de dispositivos na coleção para serem lidos.
    - dd, td, ss <-- get_next_device_info: Obtém o próximo conjunto de informações de dispositivos.