Função: JOGO Load (encontra-se no arquivo "6. O Jogo\Source\JOGO.C")

```
/*Assertivas de Entrada da Função */
ptDesenho Load( char * NomeArq , int * Altura, int * Largura )
{
      /*Bloco 1*/
        int Estado = 0, IteradorAltura = 1, IteradorLargura = 1 ;
        FILE * arq = fopen( strcat( NomeArq, ".txt" ), "r" ) ;
      /*Fim Bloco 1*/

      /*Assertiva Intermediária 1*/

      /*Bloco 2*/
        if( arq == NULL )
        {
            /*Bloco 2.1*/
              printf( "Arquivo invalido \n" ) ;
              return NULL ;
            /*Fim Bloco 2.1*/
        }
      /*Fim Bloco 2*/

      /*Assertiva Intermediária 2*/

      /*Bloco 3*/
        fscanf( arq, "%d %d", Altura, Largura ) ;
        DES_GerarMatrizes( &pDesenho, *Altura, *Largura ) ;
      /*Fim Bloco 3*/

      /*Assertiva Intermediária 3*/

      /*Bloco 4*/
        while( IteradorAltura <= *Altura )
        {
            /*Bloco 4.1*/
              for( IteradorLargura =1 ; IteradorLargura <= *Largura ;
                                                      IteradorLargura++ )
              {
                  /*Bloco 4.1.1*/
                    fscanf( arq, "%d ", &Estado ) ;
                  /*Fim Bloco 4.1.1*/

                  /*Assertiva Intermediária 6*/

                  /*Bloco 4.1.2*/
                    if( Estado == 1 )
                    {
                        /*Bloco 4.1.2.1*/
```

```
                    DES_AlterarEstadoCorreto( pDesenho,
                                              IteradorAltura,
                                              IteradorLargura ) ;
                /*Fim Bloco 4.1.2.1*/
              }
            /*Fim Bloco 4.1.2*/
          }
        /*Fim Bloco 4.1*/
        /*Bloco 4.2*/
          IteradorAltura++ ;
        /*Fim Bloco 4.2*/
    }
/*Fim Bloco 4*/
    IteradorAltura = 1 ;

/*Assertiva Intermediária 4*/

/*Bloco 5*/
    while( IteradorAltura <= *Altura )
    {
        /*Bloco 5.1*/
          for( IteradorLargura =1 ; IteradorLargura <= *Largura ;
                                                  IteradorLargura++ )
          {
            /*Bloco 5.1.1*/
              fscanf( arq,"%d ", &Estado ) ;
            /*Fim Bloco 5.1.1*/

            /*Assertiva Intermediária 7*/

            /*Bloco 5.1.2*/
              if( Estado == 1 )
              {
                /*Bloco 5.1.2.1*/
                  DES_AlterarEstadoAtual( pDesenho,
                                          IteradorAltura,
                                          IteradorLargura ) ;
                /*Fim Bloco 5.1.2.1*/
              }
            /*Fim Bloco 5.1.2*/
          }
        /*Fim Bloco 5.1*/
        /*Bloco 5.2*/
          IteradorAltura++ ;
        /*Fim Bloco 5.2*/
    }
/*Fim Bloco 5*/
```

```
      /*Assertiva Intermediária 5*/

      /*Bloco 6*/
        fscanf( arq, "%d", NumHint ) ;
        return pDesenho ;
      /*Fim Bloco 6*/
}
/*Assertivas de Saída da Função*/
```

Assertivas de Entrada da Função:

- String não nula, de até 64 caracteres contendo o nome do arquivo (sem extensão) a ser carregado. O arquivo com o respectivo nome deverá existir na pasta do jogo.
- Endereço de memória de uma variável inteira para guardar o valor da Altura.
- Endereço de memória de uma variável inteira para guardar o valor da Largura.

Assertivas de Saída da Função:

- [Estados correto e atual da matriz alterados de acordo com o que estava salvo no arquivo e retorna ponteiro para o desenho] ou [retorna ponteiro para NULL].

Bloco 1:

---Assertivas de Entrada (1):

- String não nula, de até 64 caracteres contendo o nome do arquivo (sem extensão) a ser carregado. O arquivo com o respectivo nome deverá existir na pasta do jogo.

---Assertivas de Saída (1) == Assertiva Intermediária 1:

- Valores 0, 1 e 1 atribuídos às variáveis "Estado", "IteradorAltura" e "IteradorLargura", respectivamente.
- Ponteiro para o respectivo arquivo passado como parâmetro (string) para a função atribuído à variável ariável "arq". O ponteiro receberá NULL se o arquivo não existir.

Bloco 2:

---Assertivas de Entrada (2) == Assertiva Intermediária 1:

- Assertivas de Saída do Bloco 1.

---Assertivas de Saída (2) == Assertiva Intermediária 2:

- [Variável "arq" não nula] ou [emite a mensagem "Arquivo invalido \n" e retorna um ponteiro para desenho apontando para NULL].

---1. AE2 && (Condição == verdade) + Bloco 2.1 => AS2:

- Pela AE2, o ponteiro para o arquivo pode ser NULL, caso o arquivo não exista. Como (Condição == verdade), a mensagem "Arquivo invalido \n" é emitida e a função retorna um ponteiro para desenho apontando para NULL, satisfazendo assim a AS2.

---2. AE2 && (Condição == falsa) => AS2:

- Pela AE2, o ponteiro para o arquivo pode não ser NULL, caso o arquivo exista. Como (Condição == falsa), nada acontece e o código continua. A variável "arq", portanto, não está apontando para NULL, e pode ser

utilizada, satisfazendo assim a AS2.

Bloco 3:

---Assertivas de Entrada (3) == Assertiva Intermediária 2:

- Assertivas de Saída do Bloco 2.

---Assertivas de Saída (3) == Assertiva Intermediária 3:

- Dois inteiros lidos em sequência no arquivo correspondente ao ponteiro "arq" e atribuídos aos parâmetros da função (passados por referência) Altura e Largura, respectivamente.
- Espaço alocado na memória para o desenho do jogo.

Bloco 4:

---Assertivas de Entrada (4) == Assertiva Intermediária 3:

- Assertivas de Saída do Bloco 3.

---Assertivas de Saída (4) == Assertiva Intermediária 4:

- O arquivo foi interpretado e os estados corretos das células indicados pelo arquivo foram alterados nas posições corretas.

---1. AE4 => AINV:

- Pela AE4, IteradorAltura aponta para o primeiro elemento vertical no arquivo. Todos os elementos estão no conjunto "a pesquisar" e o conjunto "já pesquisou" está vazio, valendo assim a AINV.

---2. AE4 && (Condição == falsa) => AS4:

- Para que (Condição == falsa), nessa repetição, é necessário que IteradorAltura > LL (no caso, a altura da matriz), ou seja, o arquivo será lido até onde necessário, valendo assim a AS4.

---3. AE4 && (Condição == verdade) + Bloco 4.1 + Bloco 4.2 => AINV:

- Pela AE4, IteradorAltura aponta para o primeiro elemento. Como (Condição == verdade), o elemento vai ser lido e seu respectivo lugar na matriz será ou não alterado, dependendo do seu estado. 3 garante que IteradorAltura será reposicionado para um novo elemento, e AINV é válida.

---4. AINV && (Condição == verdade) + Bloco 4.1 + Bloco 4.2 => AINV:

- Para que AINV continue valendo, 3 deve garantir que um elemento passe do conjunto "a pesquisar" para "já pesquisou" e IteradorAltura seja reposicionado.

---5. AINV && (Condição == falsa) => AS4:

- Para que (Condição == falsa), o elemento apontado por IteradorAltura é maior do que LL (no caso, a altura da matriz). Neste último caso, o conjunto "a pesquisar" está vazio, e vale a AS4.

---6. Término:

- Como o conjunto "a pesquisar" é finito, e 3 a cada ciclo retira um elemento deste conjunto, a repetição termina em um número finito de ciclos, que é igual à altura da matriz.

Bloco 5:

---Assertivas de Entrada (5) == Assertiva Intermediária 4:

- Assertivas de Saída do Bloco 4.

---Assertivas de Saída (5) == Assertiva Intermediária 5:

- O arquivo foi interpretado e os estados atuais das células indicados pelo arquivo foram alterados nas posições corretas.

---**1. AE5 => AINV:**
- Pela AE5, IteradorAltura aponta para o primeiro elemento vertical no arquivo. Todos os elementos estão no conjunto "a pesquisar" e o conjunto "já pesquisou" está vazio, valendo assim a AINV.

---**2. AE5 && (Condição == falsa) => AS5:**
- Para que (Condição == falsa), nessa repetição, é necessário que IteradorAltura > LL (no caso, a altura da matriz), ou seja, o arquivo será lido até onde necessário, valendo assim a AS5.

---**3. AE5 && (Condição == verdade) + Bloco 5.1 + Bloco 5.2 => AINV:**
- Pela AE5, IteradorAltura aponta para o primeiro elemento. Como (Condição == verdade), o elemento vai ser lido e seu respectivo lugar na matriz será ou não alterado, dependendo do seu estado. 3 garante que IteradorAltura será reposicionado para um novo elemento, e AINV é válida.

---**4. AINV && (Condição == verdade) + Bloco 5.1 + Bloco 5.2 => AINV:**
- Para que AINV continue valendo, 3 deve garantir que um elemento passe do conjunto "a pesquisar" para "já pesquisou" e IteradorAltura seja reposicionado.

---**5. AINV && (Condição == falsa) => AS5:**
- Para que (Condição == falsa), o elemento apontado por IteradorAltura é maior do que LL (no caso, a altura da matriz). Neste último caso, o conjunto "a pesquisar" está vazio, e vale a AS5.

---**6. Término:**
- Como o conjunto "a pesquisar" é finito, e 3 a cada ciclo retira um elemento deste conjunto, a repetição termina em um número finito de ciclos, que é igual à altura da matriz.

Bloco 6:

---**Assertivas de Entrada (6) == Assertiva Intermediária 5:**
- Assertivas de Saída do Bloco 5.

---**Assertivas de Saída (6):**
- É lido corretamente do arquivo o número de hints restantes e atribuído à variável global NumHint.
- Assertivas de Saída da Função.

Bloco 2.1:

---**Assertivas de Entrada (2.1) == Assertivas de Entrada (2):**
- Assertivas de Entrada do Bloco 2.

---**Assertivas de Saída (2.1):**
- Mensagem "Arquivo invalido \n" é emitida.
- Retorna um ponteiro para desenho apontando para NULL.

Bloco 4.1:

---**Assertivas de Entrada (4.1) == Assertivas de Entrada (4):**
- Assertivas de Entrada do Bloco 4.

---**Assertivas de Saída (4.1):**

- O arquivo foi interpretado e os estados corretos das células indicados pelo arquivo foram alterados nas posições corretas.

---1. AE4.1 => AINV:

- Pela AE4.1, IteradorLargura aponta para o primeiro elemento horizontal no arquivo. Todos os elementos estão no conjunto "a pesquisar" e o conjunto "já pesquisou" está vazio, valendo assim a AINV.

---2. AE4.1 && (Condição == falsa) => AS4.1:

- Para que (Condição == falsa), nessa repetição, é necessário que IteradorLargura > LL (no caso, a largura da matriz), ou seja, o arquivo será lido até onde necessário, valendo assim a AS4.1.

---3. AE4.1 && (Condição == verdade) + Bloco 4.1.1 + Bloco 4.1.2 => AINV:

- Pela AE4.1, IteradorLargura aponta para o primeiro elemento. Como (Condição == verdade), o elemento vai ser lido e seu respectivo lugar na matriz será ou não alterado, dependendo do seu estado. 3 garante que IteradorLargura será reposicionado para um novo elemento, e AINV é válida.

---4. AINV && (Condição == verdade) + Bloco 4.1.1 + Bloco 4.1.2 => AINV:

- Para que AINV continue valendo, 3 deve garantir que um elemento passe do conjunto "a pesquisar" para "já pesquisou" e IteradorLargura seja reposicionado.

---5. AINV && (Condição == falsa) => AS4.1:

- Para que (Condição == falsa), o elemento apontado por IteradorLargura é maior do que LL (no caso, a largura da matriz). Neste último caso, o conjunto "a pesquisar" está vazio, e vale a AS4.1.

---6. Término:

- Como o conjunto "a pesquisar" é finito, e 3 a cada ciclo retira um elemento deste conjunto, a repetição termina em um número finito de ciclos, que é igual à largura da matriz.

Bloco 4.2:

---O Bloco 4.2 não possui nenhuma assertiva, pois é apenas uma linha de código, incrementando o iterador da repetição.

Bloco 5.1:

---Assertivas de Entrada (5.1) == Assertivas de Entrada (5):

- Assertivas de Entrada do Bloco 5.

---Assertivas de Saída (5.1):

- O arquivo foi interpretado e os estados atuais das células indicados pelo arquivo foram alterados nas posições corretas.

---1. AE5.1 => AINV:

- Pela AE5.1, IteradorLargura aponta para o primeiro elemento horizontal no arquivo. Todos os elementos estão no conjunto "a pesquisar" e o conjunto "já pesquisou" está vazio, valendo assim a AINV.

---2. AE5.1 && (Condição == falsa) => AS5.1:

- Para que (Condição == falsa), nessa repetição, é necessário que IteradorLargura > LL (no caso, a largura da matriz), ou seja, o arquivo será lido até onde necessário, valendo assim a AS5.1.

---3. AE5.1 && (Condição == verdade) + Bloco 5.1.1 + Bloco 5.1.2 => AINV:

- Pela AE5.1, IteradorLargura aponta para o primeiro elemento. Como (Condição == verdade), o elemento vai ser lido e seu respectivo lugar na matriz será ou não alterado, dependendo do seu estado. 3 garante que IteradorLargura será reposicionado para um novo elemento, e AINV é válida.

---4. AINV && (Condição == verdade) + Bloco 5.1.1 + Bloco 5.1.2 => AINV:

- Para que AINV continue valendo, 3 deve garantir que um elemento passe do conjunto "a pesquisar" para "já pesquisou" e IteradorLargura seja reposicionado.

---5. AINV && (Condição == falsa) => AS5.1:

- Para que (Condição == falsa), o elemento apontado por IteradorLargura é maior do que LL (no caso, a largura da matriz). Neste último caso, o conjunto "a pesquisar" está vazio, e vale a AS5.1.

---6. Término:

- Como o conjunto "a pesquisar" é finito, e 3 a cada ciclo retira um elemento deste conjunto, a repetição termina em um número finito de ciclos, que é igual à largura da matriz.

Bloco 5.2:

---O Bloco 5.2 não possui nenhuma assertiva, pois é apenas uma linha de código, incrementando o iterador da repetição.

Bloco 4.1.1:

---Assertivas de Entrada (4.1.1) == Assertivas de Entrada (4.1):

- Assertivas de Entrada do Bloco 4.1.

---Assertivas de Saída (4.1.1) == Assertiva Intermediária 6:

- É lido corretamente do arquivo o estado da respectiva célula na matriz, que é atribuído à variável Estado.

Bloco 4.1.2:

---Assertivas de Entrada (4.1.2) == Assertiva Intermediária 6:

- Assertivas de Saída do Bloco 4.1.1.

---Assertivas de Saída (4.1.2):

- [Estado correto da posição baseada no IteradorAltura e IteradorLargura é alterado no desenho] ou [nada é feito].

---1. AE4.1.2 && (Condição == verdade) + Bloco 4.1.2.1 => AS4.1.2:

- Pela AE4.1.2, o estado foi lido do arquivo. Como (Condição == verdade), o estado atual naquelas coordenadas (baseadas no IteradorAltura e no IteradorLargura) é modificado, satisfazendo assim a AS4.1.2.

---2. AE4.1.2 && (Condição == falso) => AS4.1.2:

- Nada é feito, satisfazendo a AS4.1.2.

Bloco 5.1.1:

---Assertivas de Entrada (5.1.1) == Assertivas de Entrada (5.1):

- Assertivas de Entrada do Bloco 5.1.

---Assertivas de Saída (5.1.1) == Assertiva Intermediária 7:

- É lido corretamente do arquivo o estado da respectiva célula na matriz, que é atribuído à variável Estado.

Bloco 5.1.2:

---Assertivas de Entrada (5.1.2) == Assertiva Intermediária 7:

- Assertivas de Saída do Bloco 5.1.1.

---Assertivas de Saída (5.1.2):

- [Estado atual da posição baseada no IteradorAltura e IteradorLargura é alterado no desenho] ou [nada é feito];

---1. AE5.1.2 && (Condição == verdade) + Bloco 5.1.2.1 => AS5.1.2:

- Pela AE5.1.2, o estado foi lido do arquivo. Como (Condição == verdade), o estado atual naquelas coordenadas (baseadas no IteradorAltura e no IteradorLargura) é modificado, satisfazendo assim a AS5.1.2.

---2. AE5.1.2 && (Condição == falso) => AS5.1.2:

- Nada é feito, satisfazendo a AS5.1.2.

Bloco 4.1.2.1:

---Assertivas de Entrada (4.1.2.1) == Assertivas de Entrada (4.1.2):

- Assertivas de Entrada do Bloco 4.1.2.

---Assertivas de Saída (4.1.2.1):

- O estado atual naquelas coordenadas (baseadas no IteradorAltura e no IteradorLargura) é modificado.

Bloco 5.1.2.1:

---Assertivas de Entrada (5.1.2.1) == Assertivas de Entrada (5.1.2):

- Assertivas de Entrada do Bloco 5.1.2.

---Assertivas de Saída (5.1.2.1):

- O estado atual naquelas coordenadas (baseadas no IteradorAltura e no IteradorLargura) é modificado.