Aluno: Thomas Santos Pollarini

SIMULADOR AEROPORTO

1.INTRODUÇÃO

O trabalho desenvolvido teve como objetivo, desenvolver uma simulação para um aeroporto, onde deveria ser implementado a entrada de aviões, fazer o controle de combustível, de aterrissagem e decolagem, de forma a minimizar a queda de aviões e o tempo médio de espera. Neste documento será descrito todo o funcionamento do aeroporto, de forma a facilitar a compreensão das estruturas de dados desenvolvidas.

2.1 AVIÕES E FILAS

TAD criado para controle dos aviões:

Struct tipoitem:

int ID: Número de identificação do avião
int comb: Quantidade de combustível do avião
int tempo: Contabiliza tempo de espera do avião na fila

}

Struct tipocelula:

tipoitem avião: Armazena os dados do avião
tipoapontador prox: Aponta para o próximo avião

}

Struct tipoapontador: ponteiro para tipocelula.

Struct tipofila:

int tam: Contabiliza quantidade de aviões presentes na fila
int tipo: Define tipo da fila (1=Aterrissagem/2=Decolagem)
tipoapontador prim: Aponta para célula cabeça da fila
tipoapontador ult: Aponta para ultimo avião da fila

}

OPERAÇÕES BASES:

void cria(tipofila *fila): Inicializa apontadores de uma fila passada por referência, apontando-os para a célula cabeça da fila.

void enfileira(tipofila *fila, int *id): Cria nova célula para armazenar novo avião na fila passada por referência, insere dados (id, combustível), inicializa contador de tempo de espera e organiza os ponteiros.

void desenfileira(tipoapontador aux, tipofila *fila): Faz controles dos ponteiros para remover aviões armazenados na fila e libera célula retirada.

void imprime(tipofila fila): Imprime dados armazenados na fila.

int maior_fila(tipofila *fila): Função responsável por definir qual fila será escolhida para retirada de um avião. A função compara tamanho de todas as filas de um tipo e retorna fila com maior número de aviões. Lembrando que em caso de filas com tamanho igual, retorna fila onde primeiro avião contém menos combustível.

int menor_fila(tipofila *fila): Função responsável por definir qual fila será escolhida para entrada de um avião. A função compara tamanho de todas as filas de um tipo e retorna fila com menor número de aviões. Lembrando que em caso de filas com tamanho igual, retorna fila onde primeiro avião está com maior tempo de espera, dessa forma agilizando saída de aviões com maior tempo de espera.

Int tamanho(tipofila *fila): Soma quantidade de aviões em todas as filas de um tipo e retorna tamanho total de aviões desse tipo.

2.2 CONTROLE DE CICLOS

Antes do primeiro ciclo as filas são iniciadas através de laços de repetição. Para a simulação começar o usuário digita o número de ciclos que serão rodados antes de imprimir resultados. Em cada ciclo são realizadas 3 funções: **entrada de dados, controle das pistas e controle de dados**. A cada ciclo é aumentada uma unidade de tempo, no ultimo ciclo acontece a impressão das filas e estatísticas importantes.

2.3 ENTRADA

void entrada((tipofila *fa, tipofila *fd,int *idA, int *idD): Gera número aleatório de aviões para aterrissar e decolar (0 a 4). Utiliza função **menor_fila** e passa fila retornada para função **enfileira**, dessa forma mantendo as filas com mesmo tamanho.

```
void entrada(tipofila *fa, tipofila *fd,int *idA, int *idD){

    int i,j,nA,nD,pos;

    nA=rand()%4;  //recebe número de aviões que vão aterrizar, de 0 a 3
    nD=rand()%4;  //recebe número de aviões que vão decolar, de 0 a 3

    for(i=0;i<nA;i++){       //laço com número de aviões que serão adicionados
        pos=menor_fila(fa);  //seleciona em qual fila o avião será colocado
        enfileira(&fa[pos],idA); //aloca novo avião na fila encontrada
    }

    for(i=0;i<nD;i++){          //laço com número de aviões que serão adicionados
        pos=menor_fila(fd);     //seleciona em qual fila o avião será colocado
        enfileira(&fd[pos],idD);//aloca novo avião na fila encontrada
    }
```

2.4 ATERRISSAGEM E DECOLAGEM

void cont_pista(tipofila *fa, tipofila *fd, int *emer, float *tempoA, float *tempoD, int *contA, int *contD): Primeiramente a função realiza a verificação de aviões que não possuem reserva de combustível, dessa forma necessitando de fazer um pouso de emergência, o qual tem prioridade máxima, levando em conta que só existem 3 pistas disponíveis. Também é feito o controle de dados importantes para as estáticas do simulador (média de tempo de espera, e contador de pouso de emergência).

```
for(i=0;i<tA;i++){             //controle de pouso de emergência
    if(fa[i].prim->prox!=NULL){
        if(fa[i].prim->prox->aviao.comb==1){    //verifica se aviões estão sem reserva
            cont++; //contador de ações (pouso/decolagem)
            if(cont<nPista){//pousa no máximo 3 aviões em emergência

                *tempoA=*tempoA+fa[i].prim->prox->aviao.tempo;  //adiciona o tempo de espera desse avião ao tempo total
                *contA=*contA+1;                                //adiciona esse avião ao número de aviões que aterrissaram

                desenfileira(fa[i].prim,&fa[i]);  //remove aviões pousados da lista

                fa[i].tam--; //diminui contador de tamanho
                *emer=*emer+1;  //aumenta contador de pousos de emergência
            }
        }
    }
}
```

Após o controle de pouso de emergência, o simulador decide se vai realizar uma decolagem ou uma aterrissagem com base no número de aviões totais de cada ação, lembrando que caso não seja realizado nenhum pouso de emergência uma pista é destinada apenas para decolagem.

```
for(i=cont;i<nPista;i++){
    if(i==0){ //verifica se pista 3 não foi utilizada para pouso de emergência
        tam2=tamanho(fd);  //verifica se há aviões para decolar
        if(tam2>0){
            pos=maior_fila(fd); //seleciona fila de decolagem apropriada

            *tempoD=*tempoD+fd[pos].prim->prox->aviao.tempo; //adiciona o tempo de espera desse avião ao tempo total
            *contD=*contD+1;                                 //adiciona esse avião ao número de aviões que decolaram

            desenfileira(fd[pos].prim,&fd[pos]); //retira avião que decolou da fila

            fd[pos].tam--;//diminui contador de tamanho
        }
```

controle de pista dedicada a decolagem.

```
}else{
    tam1=tamanho(fa); //armazena número total de aviões que devem aterrissar
    tam2=tamanho(fd); //armazena número total de aviões que devem decolar

                  //verifica tam>0 para evitar problemas
    if(tam1>tam2 && tam1>0){ //seleciona se pista será usada para decolagem ou aterrissagem (definido por quem tem mais aviões)
        pos=maior_fila(fa); //seleciona fila de aterrissagem apropriada

        *tempoA=*tempoA+fa[pos].prim->prox->aviao.tempo;  //adiciona o tempo de espera desse avião ao tempo total
        *contA=*contA+1;                                  //adiciona esse avião ao número de aviões que aterrissaram

        desenfileira(fa[pos].prim,&fa[pos]); //retira avião que aterrissou da fila

        fa[pos].tam--;//diminui contador de tamanho
    }else{
        if(tam2>0){
            pos=maior_fila(fd);//seleciona fila de decolagem apropriada

            *tempoD=*tempoD+fd[pos].prim->prox->aviao.tempo; //adiciona o tempo de espera desse avião ao tempo total
            *contD=*contD+1;                                 //adiciona esse avião ao número de aviões que decolaram

            desenfileira(fd[pos].prim,&fd[pos]);//retira avião que decolou da fila

            fd[pos].tam--; //diminui contador de tamanho
        }
    }
}
```

A retirada de todos os aviões é feita utilizando a função **maior_fila** para localizar fila adequada para retirada e passando-a para a função **desenfileira**.

2.5 CONTROLE DE DADOS

void cont_dados(tipofila *fa, tipofila *fd, int *queda): Essa função realiza o controle de dados dos aviões, diminuindo o combustível dos aviões que estão voando e aumentando o tempo de espera daqueles que ainda não aterrissaram e decolaram no ciclo. Juntamente com o controle de combustível, é realizado o controle de quedas, ou seja, quando o combustível de um avião acaba, é feita a retirada dele da fila através da função **desenfileira** e é aumentado o contador de quedas.

```c
void cont_dados(tipofila *fa, tipofila *fd,int *queda){

    int i,j;
    tipoapontador aux;

    for(i=0;i<tA;i++){  //aviões para  aterrissar
        aux=fa[i].prim;

        while(aux->prox!=NULL){
            aux->prox->aviao.tempo++; //aumenta tempo de espera de cada avião
            aux->prox->aviao.comb--;  //diminui combustivel de aviões voando

            if(aux->prox->aviao.comb<=0){
                desenfileira(aux,&fa[i]);  //retira aviões sem combustivel da lista - queda
                *queda=*queda+1;           //aumenta contador de quedas
                fa[i].tam--;               //diminui contador de tamanho da lista
            }else{
                aux=aux->prox;
            }
        }
    }
    for(i=0;i<tD;i++){ //aviões para decolar
        aux=fd[i].prim->prox;

        while(aux!=NULL){
            aux->aviao.tempo++; //aumenta tempo de espera
            aux=aux->prox;
        }
    }
}
```