Enzo Galeazzo Casagrande - 32209606 Marcos Carvalho Junior - 32234120 Rafael Marques Cruz Russo - 32220294

Problema das Escadas Rolantes Duplas

## Código:

Declaração de constantes, estruturas e funções do ambiente;

```
#define MAX_LEN 500

struct user
{
    int momento;
    int direcao;
};

typedef struct user usuario;

void PreparaData(char data[MAX_LEN][MAX_LEN], int line, usuario *usuario);
void fillGroup(usuario *grupo, char data[MAX_LEN][MAX_LEN]);
int escada(usuario *grupo, int n);
```

### Função main():

Aqui é feita a leitura do arquivo de entrada e chamada das funções auxiliares

```
int main()
        int line = 0;
        usuario *grupo;
        char data[MAX_LEN][MAX_LEN];
        FILE *file = fopen("..//escada.txt", "r");
        if (file == NULL)
            printf("ERRO AO ABRIR O ARQUIVO");
           return 1;
        }
        while (!feof(file) && !ferror(file))
            if (fgets(data[line], MAX_LEN, file) != NULL)
                line++;
        fclose(file);
        grupo = (usuario *)malloc(atoi(data[0]) * sizeof(usuario));
        fillGroup(grupo, data);
        int momento = escada(grupo, atoi(data[0]));
        printf("Ultima chegada: %d", momento);
26 }
```

### Funções fillGroup() e PrepareData():

funções auxiliares que mapeiam o arquivo de entrada em uma lista do tipo "usuario";

```
void fillGroup(usuario *grupo, char data[MAX_LEN][MAX_LEN])

{
    int currentLine = 0;
    int n_pessoas = atoi(data[currentLine]);
    currentLine++;
    for (int i = 0; i < n_pessoas; i++)
    {
        PreparaData(data, i + 1, &grupo[i]);
    }

void PreparaData(char data[MAX_LEN][MAX_LEN], int line, usuario *usuario)

{
    char *token = strtok(data[line], " ");
    if (atoi(token) == 0)
    {
        return;
    }
    usuario->momento = atoi(token);
    token = strtok(NULL, " ");
    usuario->direcao = atoi(token);
}
```

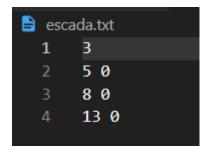
#### Função escada():

Função responsável por tomar as decisões e simular o problema

```
int escada(usuario *grupo, int n)
        usuario current = grupo[0];
        int i = 0, estimativa;
        usuario fila[MAX_LEN];
        int restantes = n, direcao = -1, tempo = 0;
        bool await = false;
        while (restantes > 0)
            if (await && (grupo[i].momento > estimativa || i >= restantes))
                await = false;
               current = fila[0];
                tempo += 10;
               direcao = current.direcao;
                estimativa = tempo + 10;
                restantes -= 1;
                current = grupo[i];
                if (direcao == -1)
                   tempo = current.momento < tempo ? tempo : current.momento;</pre>
                   direcao = current.direcao;
                   estimativa = current.momento + 10;
                    i++, restantes--;
                else if (direcao == current.direcao)
                    tempo = current.momento;
                    estimativa = current.momento + 10;
                    i++, restantes--;
                    if (grupo[i + 1].momento - grupo[i].momento > grupo[i - 1].momento)
                        tempo = estimativa, direcao = -1;
                    else if (grupo[i + 1].momento <= estimativa)</pre>
                        fila[0] = grupo[i];
                        await = true;
                    else if (grupo[i + 1].momento == 0 || grupo[i + 1].momento > estimativa)
                        direcao = -1;
        tempo += 10;
        return tempo;
```

# Teste:

#### Entrada:



#### Saída:

```
Ultima chegada: 23
```