In [ ]:
```
# Carregando Pandas para uso nos calculos das planilhas
import pandas as pd
from datetime import datetime
# Carregando arquivo.csv em memória (para manipulaçaõ)
# Em meu caso, eu estou modificando o "df = pandas.read_csv('../Database/dataset.csv')"
# Eu estou utilizando uma função lambda para usar um parse no campo datas
dateparse = lambda x: datetime.strptime(x, '%H:%M:%S')
df = pd.read_csv('../Database/dataset.csv', parse_dates=True, date_parser=dateparse)
df.head(31)
```

In [ ]:
```
#
# Etapas para tratar os dados e realizar os cálculos - MTTF, MTTR, MTBF e Disponibilidade - Tempo de Recebimento de Mercadoria
#
```

In [ ]:
```
# Primeiramente, iremos Ordenar o valores de recebimento, para criar uma sequência de dados crescente (tempo_inicio_recebimento_mercad
```

In [ ]:
```
df.sort_values(by=['tempo_inicio_recebimento_mercadoria'])
```

In [ ]:
```
tempo_inicio_recebimento_mercadoria = df['tempo_inicio_recebimento_mercadoria']
```

In [ ]:
```
tempo_fim_recebimento_mercadoria = df['tempo_fim_recebimento_mercadoria']
```

In [ ]:
```
tempo_inicio_recebimento_mercadoria = pd.to_datetime(tempo_inicio_recebimento_mercadoria)
```

In [ ]:
```
tempo_fim_recebimento_mercadoria = pd.to_datetime(tempo_fim_recebimento_mercadoria)
```

In [ ]:
```
# Usando uma variável para armazenar o valores de Tempo Total Disponível (1ª tempo inicial - último tempo final)
# Com a subtração acima, é possível ter um valor de tempo disponível
```

In [ ]:
```
tempo_total_disponivel_recebimento_mercadoria = tempo_fim_recebimento_mercadoria[7] - tempo_inicio_recebimento_mercadoria[15]
```

In [ ]:
```
tempo_total_disponivel_recebimento_mercadoria
```

In [ ]:
```
tempo_duracao_recebimento_mercadoria = tempo_fim_recebimento_mercadoria - tempo_inicio_recebimento_mercadoria
```

In [ ]:
```
tempo_duracao_recebimento_mercadoria
```

In [ ]:
```
# Usando uma variável para armazenar Tempo Total Utilizado (somando todos os valores da sequência de tempo de duração)
# Com esta operação, é possível ter o valor final de tempo de atividade (produção)
```

In [ ]:
```
tempo_total_utilizado_recebimento_mercadoria = tempo_duracao_recebimento_mercadoria.sum()
```

In [ ]:
```
tempo_total_utilizado_recebimento_mercadoria
```

In [ ]:
```
# Usando uma variável para armazernar o Tempo Total de Paradas (subtraindo do Tempo Total Disponível o Tempo Total Utilizado)
```

In [ ]:
```
tempo_total_parada_recebimento_mercadoria = tempo_total_disponivel_recebimento_mercadoria - tempo_total_utilizado_recebimento_mercador
```

In [ ]:
```
tempo_total_parada_recebimento_mercadoria
```

In [ ]:
```
# Usando uma variável para armazenar as ocorrências de paradas (quantidade de registro -1, nos mostra todas as paradas da série)
```

In [ ]:
```
qtde_paradas_recebimento_mercadorias = tempo_duracao_recebimento_mercadoria.count() - 1
```

In [ ]:
```
qtde_paradas_recebimento_mercadorias
```

In [ ]:
```
# Calculando o MTTF das Colunas de Recebimento
```

In [ ]:
```
mttf_recebimento_mercadoria = tempo_duracao_recebimento_mercadoria.mean()
```

In [ ]:
```
mttf_recebimento_mercadoria
```

In [ ]:
```
# Calculando o MTTR das Colunas de Recebimento
```

In [ ]:
```
mttr_recebimento_mercadoria = tempo_total_parada_recebimento_mercadoria / qtde_paradas_recebimento_mercadorias
```

In [ ]:
```
mttr_recebimento_mercadoria
```

In [ ]:
```
# Calculando o MTBF das Colunas de Recebimento
```

In [ ]:
```
mtbf_recebimento_mercadoria = (tempo_total_disponivel_recebimento_mercadoria - tempo_total_parada_recebimento_mercadoria) / qtde_parad
```

In [ ]:
```
mtbf_recebimento_mercadoria
```

In [ ]:
```
# Calculando a Disponibilidade das Colunas de Recebimento
```

In [ ]:
```
disponibilidade_recebimento_mercadoria = mtbf_recebimento_mercadoria / (mttr_recebimento_mercadoria + mtbf_recebimento_mercadoria)
```

In [ ]:
```
disponibilidade_recebimento_mercadoria
```

In [ ]:
```
#
# Etapas para tratar os dados e realizar os cálculos - MTTF, MTTR, MTBF e Disponibilidade - Tempo de Pré Triagem
#
```

In [ ]:
```
# Primeiramente, iremos Ordenar o valores de recebimento, para criar uma sequência de dados crescente (tempo_inicio_pre_triagem)
```

In [ ]:
```
df.sort_values(by=['tempo_inicio_pre_triagem'])
```

In [ ]:
```
tempo_inicio_pre_triagem = df['tempo_inicio_pre_triagem']
```

In [ ]:
```
tempo_fim_pre_triagem = df['tempo_fim_pre_triagem']
```

In [ ]:
```
tempo_inicio_pre_triagem = pd.to_datetime(tempo_inicio_pre_triagem)
```

In [ ]:
```
tempo_fim_pre_triagem = pd.to_datetime(tempo_fim_pre_triagem)
```

In [ ]:
```
# Usando uma variável para armazenar o valores de Tempo Total Disponível (1ª tempo inicial - último tempo final)
# Com a subtração acima, é possível ter um valor de tempo disponível
# O uso da operação "+ pd.Timedelta("1 days")" é para corrigir erro do valor negativo.
```

In [ ]:
```
tempo_total_disponivel_pre_triagem = (tempo_fim_pre_triagem[15] - tempo_inicio_pre_triagem[1]) + pd.Timedelta("1 days")
```

In [ ]:
```
tempo_total_disponivel_pre_triagem
```

In [ ]:
```
tempo_duracao_pre_triagem = tempo_fim_pre_triagem - tempo_inicio_pre_triagem
```

In [ ]:
```
tempo_duracao_pre_triagem
```

In [ ]:
```
# Usando uma variável para armazenar Tempo Total Utilizado (somando todos os valores da sequência de tempo de duração)
# Com esta operação, é possível ter o valor final de tempo de atividade (produção)
```

In [ ]:
```
tempo_total_utilizado_pre_triagem = tempo_duracao_pre_triagem.sum() + pd.Timedelta("1 days")
```

In [ ]:
```
tempo_total_utilizado_pre_triagem
```

In [ ]:
```
# Usando uma variável para armazernar o Tempo Total de Paradas (subtraindo do Tempo Total Disponível o Tempo Total Utilizado)
```

In [ ]:
```
tempo_total_parada_pre_triagem = tempo_total_disponivel_pre_triagem - tempo_total_utilizado_pre_triagem
```

In [ ]:
```
tempo_total_parada_pre_triagem
```

In [ ]:
```
# Usando uma variável para armazenar as ocorrências de paradas (quantidade de registro -1, nos mostra todas as paradas da série)
```

In [ ]:
```
qtde_paradas_pre_triagem = tempo_duracao_pre_triagem.count() - 1
```

In [ ]:
```
qtde_paradas_pre_triagem
```

In [ ]:
```
# Calculando o MTTF das Colunas de Pré Triagem
```

In [ ]:
```
# Correção do item 15 da nossa lista (-1 days +00:05:00).
# A correção é necessária para desprezar o valor de "-1 days" do resultado. Correto é "00:05:00"
# Observação-1: a multiplicação por -1 (*-1) não resolvia o problema, por que estamos trabalhando com tipo data/hora
# Observação-2: esta solução deve ser executada uma única vez. Caso necessário uma segunda ou mais execuções
#               será necessário "limpar todas as saídas" e rexecutar todos os calculos novamente.
```

In [ ]:
```
x_tempo_duracao_pre_triagem = tempo_duracao_pre_triagem
```

In [ ]:
```
x_tempo_duracao_pre_triagem[15] = x_tempo_duracao_pre_triagem[15] + pd.Timedelta("1 days")
```

In [ ]:
```
mttf_pre_triagem = x_tempo_duracao_pre_triagem.mean()
```

In [ ]:
```
mttf_pre_triagem
```

In [ ]:
```
# Calculando o MTTR das Colunas de Pré Triagem
```

In [ ]:
```
mttr_pre_triagem = tempo_total_parada_pre_triagem / qtde_paradas_pre_triagem
```

In [ ]:
```
mttr_pre_triagem
```

In [ ]:
```
# Calculando o MTBF das Colunas de Pré Triagem
```

In [ ]:
```
mtbf_pre_triagem = (tempo_total_disponivel_pre_triagem - tempo_total_parada_pre_triagem) / qtde_paradas_pre_triagem
```

In [ ]:
```
mtbf_pre_triagem
```

In [ ]:
```
# Calculando a Disponibilidade das Colunas de Pré Triagem
```

In [ ]:
```
disponibilidade_pre_triagem = mtbf_pre_triagem / (mttr_pre_triagem + mtbf_pre_triagem)
```

In [ ]:
```
disponibilidade_pre_triagem
```

In [ ]:
```
#
# Etapas para tratar os dados e realizar os cálculos - MTTF, MTTR, MTBF e Disponibilidade - Tempo de Raio X
#
```

In [ ]:
```
# Primeiramente, iremos Ordenar o valores de recebimento, para criar uma sequência de dados crescente (tempo_inicio_raio_x)
```

In [ ]:
```
df.sort_values(by=['tempo_inicio_raio_x'])
```

In [ ]:
```
tempo_inicio_raio_x = df['tempo_inicio_raio_x']
```

In [ ]:
```
tempo_fim_raio_x = df['tempo_fim_raio_x']
```

In [ ]:
```
tempo_inicio_raio_x = pd.to_datetime(tempo_inicio_raio_x)
```

In [ ]:
```
tempo_fim_raio_x = pd.to_datetime(tempo_fim_raio_x)
```

In [ ]:
```
# Usando uma variável para armazenar o valores de Tempo Total Disponível (1ª tempo inicial - último tempo final)
# Com a subtração acima, é possível ter um valor de tempo disponível
```

In [ ]:
```
tempo_total_disponivel_raio_x = (tempo_fim_raio_x[4] - tempo_inicio_raio_x[27]) + pd.Timedelta("1 days")
```

In [ ]:
```
tempo_total_disponivel_raio_x
```

In [ ]:
```
tempo_duracao_raio_x = tempo_fim_raio_x - tempo_inicio_raio_x
```

In [ ]:
```
# Usando uma variável para armazenar Tempo Total Utilizado (somando todos os valores da sequência de tempo de duração)
# Com esta operação, é possível ter o valor final de tempo de atividade (produção)
```

In [ ]:
```
tempo_total_utilizado_raio_x = tempo_duracao_raio_x.sum()
```

In [ ]:
```
tempo_total_utilizado_raio_x
```

In [ ]:
```
# Usando uma variável para armazernar o Tempo Total de Paradas (subtraindo do Tempo Total Disponível o Tempo Total Utilizado)
```

In [ ]:
```
tempo_total_parada_raio_x = tempo_total_disponivel_raio_x - tempo_total_utilizado_raio_x
```

In [ ]:
```
tempo_total_parada_raio_x
```

In [ ]:
```
# Usando uma variável para armazenar as ocorrências de paradas (quantidade de registro -1, nos mostra todas as paradas da série)
```

In [ ]:
```
qtde_paradas_raio_x = tempo_duracao_raio_x.count() - 1
```

In [ ]:
```
qtde_paradas_raio_x
```

In [ ]:
```
# Calculando o MTTF das Colunas de Raio X
```

In [ ]:
```
mttf_raio_x = tempo_duracao_raio_x.mean()
```

In [ ]:
```
mttf_raio_x
```

In [ ]:
```
# Calculando o MTTR das Colunas de Raio X
```

In [ ]:
```
mttr_raio_x = tempo_total_parada_raio_x / qtde_paradas_raio_x
```

In [ ]:
```
mttr_raio_x
```

In [ ]:
```
# Calculando o MTBF das Colunas de Raio X
```

In [ ]:
```
mtbf_raio_x = (tempo_total_disponivel_raio_x - tempo_total_parada_raio_x) / qtde_paradas_raio_x
```

In [ ]:
```
mtbf_raio_x
```

In [ ]:
```
# Calculando a Disponibilidade das Colunas de Raio X
```

In [ ]:
```
disponibilidade_raio_x = mtbf_raio_x / (mttr_raio_x + mtbf_raio_x)
```

In [ ]:
```
disponibilidade_raio_x
```

In [ ]:
```

```