Integrantes:

Anderson Yavi Fernandez – RA:24025678

Gabriel Gonçalves Pires – RA: 24026518

Isabela Nunes Zeferino – RA:24026460

Kaique Neres de Oliveira – RA:24026134

Felipe Galdino de Carvalho – RA: 2402656

1. Conceito de Busca Gulosa

A busca gulosa é um algoritmo que toma decisões passo a passo, escolhendo em cada etapa a opção que parece ser a melhor no momento, segundo uma heurística definida.

- **Vantagens: execução rápida, simplicidade e facilidade de implementação.**
- **Desvantagens: pode não encontrar a solução global ótima, pois considera apenas decisões locais.**

2. Aplicação da busca gulosa nos Dados da Cannoli

A busca gulosa foi aplicada a diferentes heurísticas utilizando os databases de pedidos, clientes, campanhas e templates fornecidos pela Cannoli. Cada heurística busca otimizar um aspecto do processo de negócio.

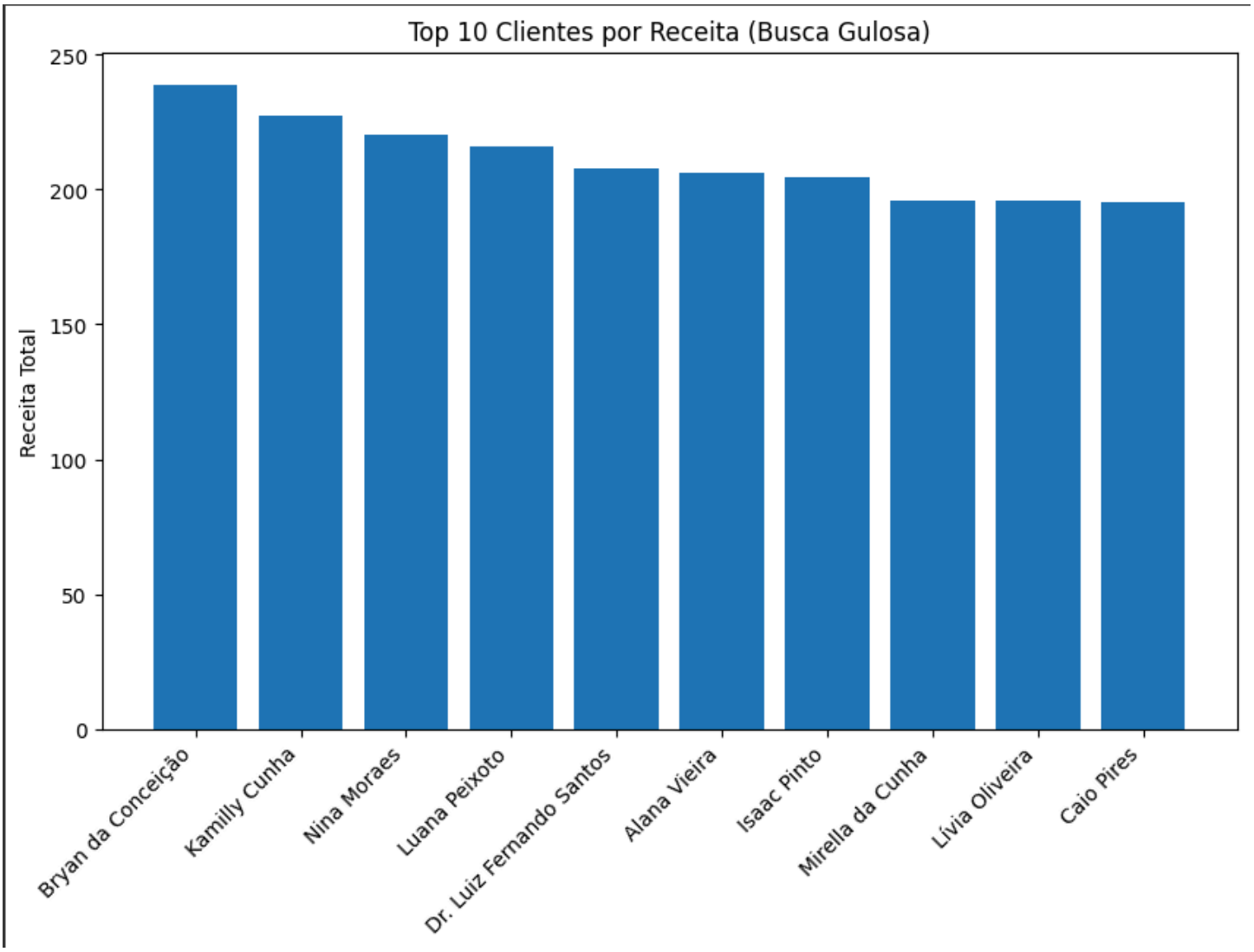
3.1 Maximizar Receita (Pedidos)

Descrição: Seleciona os clientes com maior valor total de pedidos.
Objetivo: priorizar clientes que geram mais receita.

📊 **Gráfico – Maximizar Receita (Pedidos)*)***

Gráfico de Barras Verticais (clientes no eixo X, valor total dos pedidos no eixo Y).

- **Bom para comparar valores de receita entre diferentes clientes.**



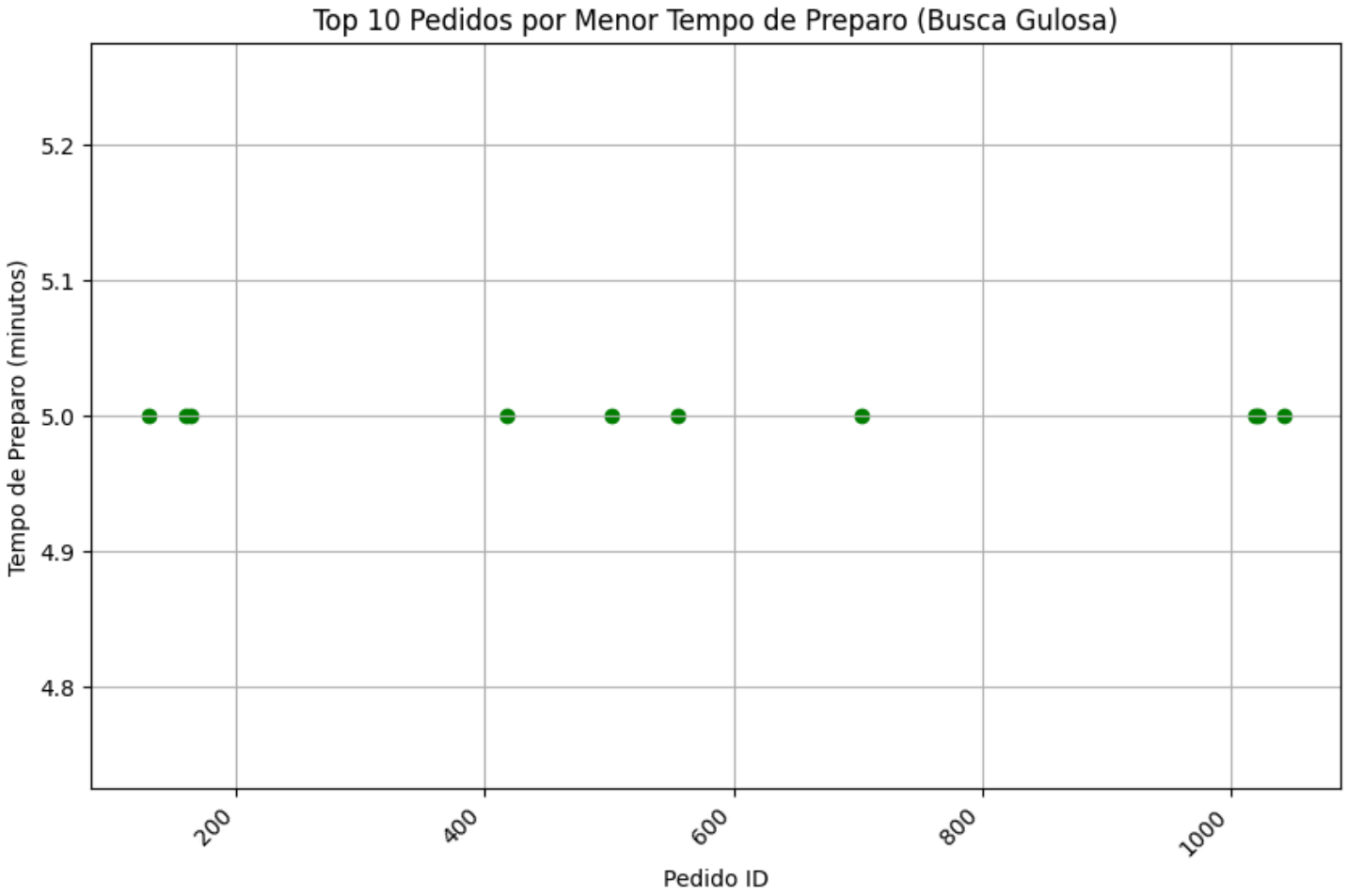
3.2 Otimizar Tempo de Preparo (Pedidos)

Descrição: Ordena os pedidos(id) pelo menor tempo de preparação.
Objetivo: agilizar processos e reduzir o tempo de entrega.

📊 **Gráfico – Otimizar Tempo de Preparo (Pedidos)**

Gráfico de Dispersão (pedido no eixo X, tempo de preparo no eixo Y).

- **Bom para visualizar a distribuição dos tempos, identificar pedidos mais rápidos, mais lentos e possíveis outliers.**



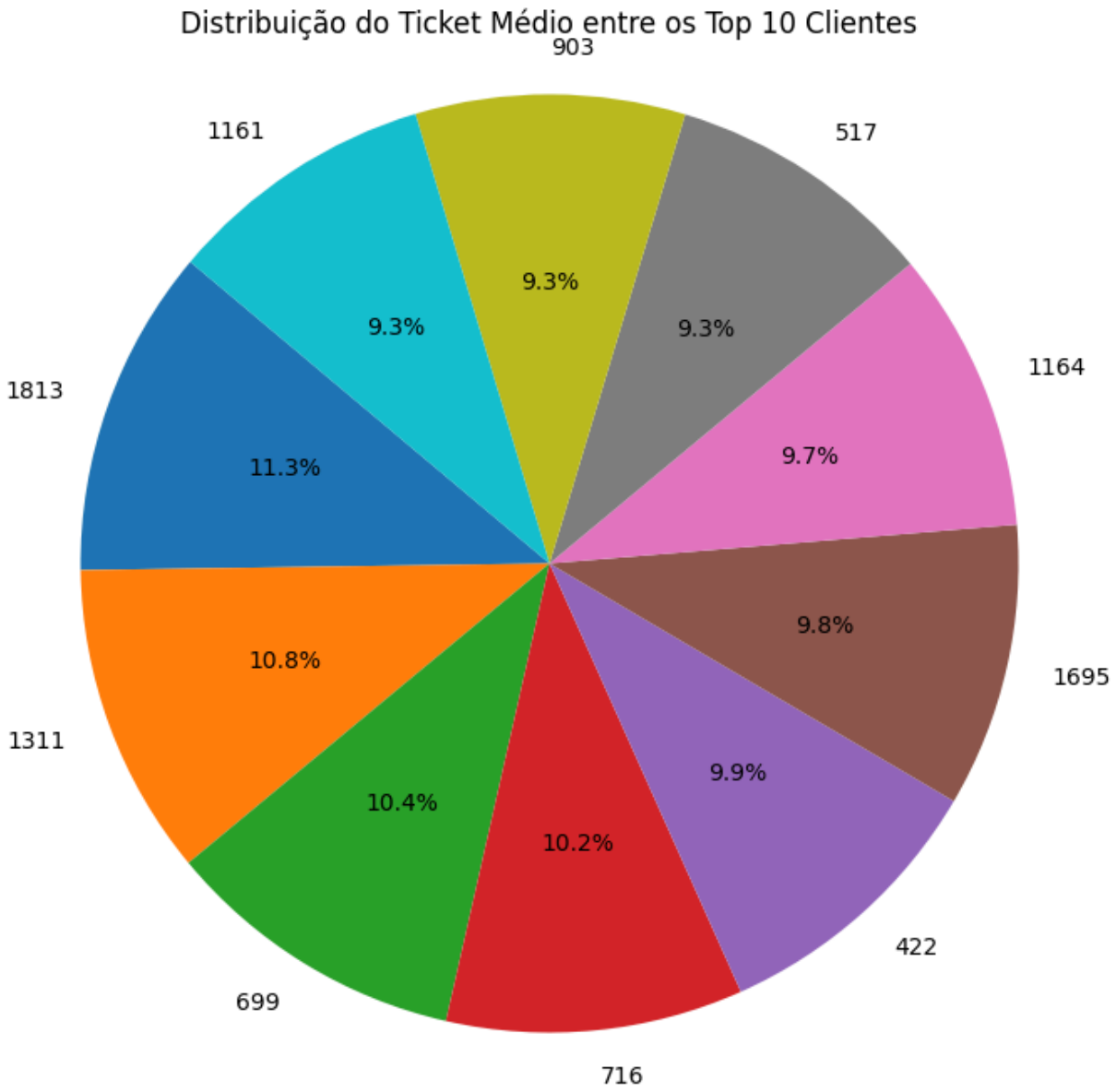
3.3 Priorizar Clientes pelo Ticket Médio

Descrição: Calcula a média de gasto por cliente(id) e prioriza os que possuem ticket mais alto.
Objetivo: identificar clientes estratégicos de maior valor para campanhas e atendimento.

📊 **Gráfico – Ticket médio de clientes**

Gráfico de Pizza (fatias representam a participação relativa do ticket médio por cliente/grupo).

- **Bom para mostrar a proporção de participação e identificar clientes estratégicos de maior valor.**



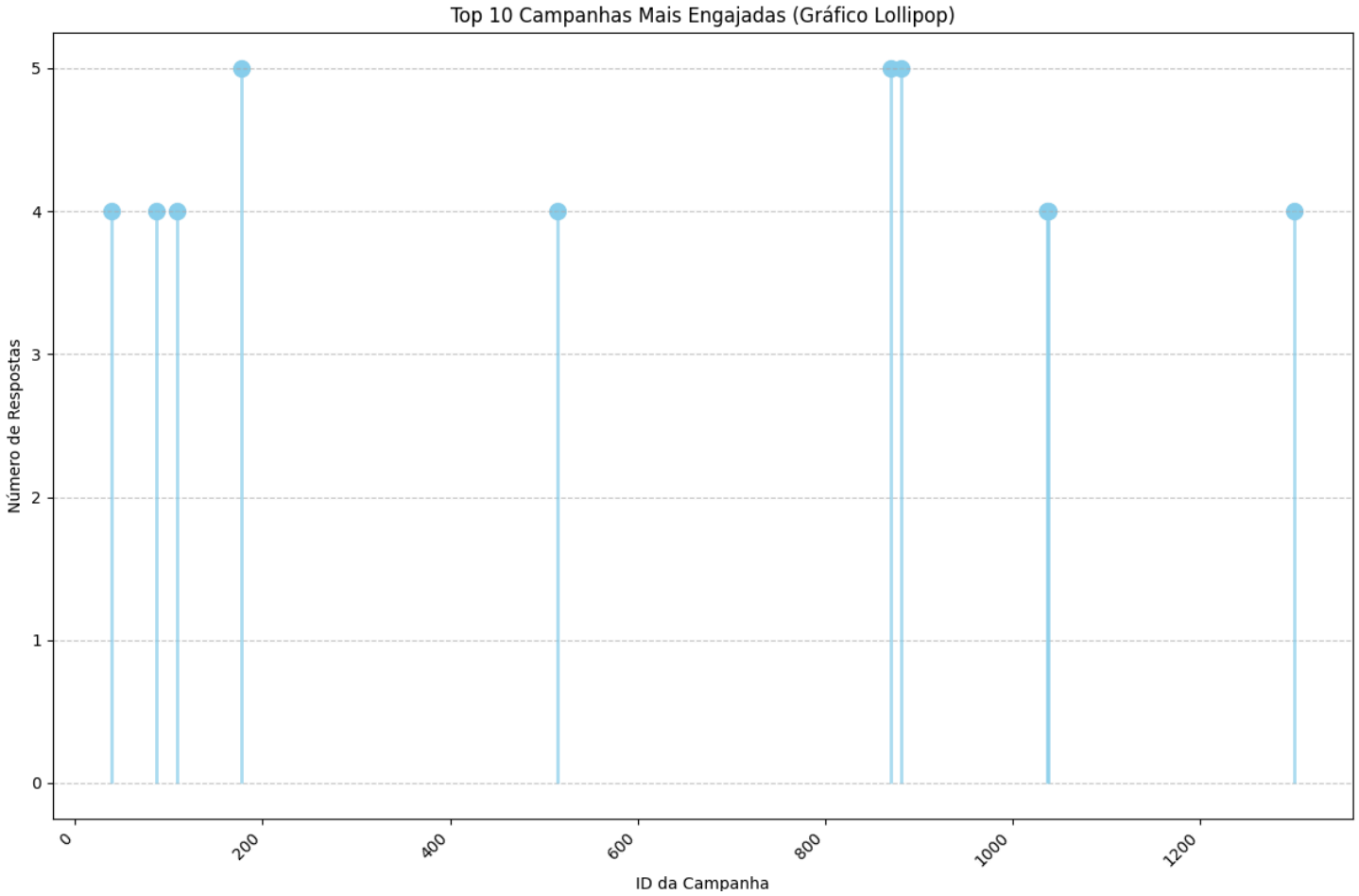
3.4 Selecionar Campanhas Mais Engajadas

Descrição: Conta a quantidade de respostas por campanha e prioriza as mais interativas.
Objetivo: maximizar o impacto das campanhas de marketing.

📊 Gráfico – Selecionar Campanhas Mais Engajadas

Gráfico de Barras Pirulito (Lollipop chart, campanhas no eixo X e respostas no eixo Y).

- **Bom para destacar visualmente quais campanhas tiveram maior engajamento, facilitando a comparação.**



4. Resultados Esperados

A aplicação da busca gulosa permite:

- **Ordenar clientes e pedidos de acordo com prioridades estratégicas.**
- **Otimizar tempo de produção e entrega.**
- **Identificar campanhas de maior engajamento.**
- **Gerar gráficos e listas que auxiliam na tomada de decisão de forma prática e visual.**

5. Conclusão

A utilização da busca gulosa demonstra a aplicabilidade de algoritmos de IA em problemas reais de negócios, permitindo decisões mais eficientes e baseadas em dados concretos da empresa Cannoli.

Essa aplicação evidencia como heurísticas simples podem gerar resultados significativos em processos de priorização e otimização.

In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import heapq

# Carregar os datasets
order_df = pd.read_csv("Order_semicolon.csv", sep=";")
campaign_queue_df = pd.read_csv("CampaignQueue_semicolon.csv", sep=";")
campaign_df = pd.read_csv("Campaign_semicolon.csv", sep=";")
customer_df = pd.read_csv("Customer_semicolon.csv", sep=";")

# 1. Busca Gulosa - Maximizar Receita (Pedidos)

def busca_gulosa_receita(order_df, top_n=10):
    fila = []
    for _, row in order_df.iterrows():
        valor = row["totalAmount"] if not pd.isna(row["totalAmount"]) else 0
        heapq.heappush(fila, (-valor, row["customerId"]))  # negativo p/ maior valo

    resultado = []
    for _ in range(min(top_n, len(fila))):
        valor, cliente = heapq.heappop(fila)
        resultado.append((cliente, -valor))
    return resultado

top_receita = busca_gulosa_receita(order_df, 10)
plt.figure(figsize=(8,4))
clientes, valores = zip(*top_receita)
plt.bar(clientes, valores, color='dodgerblue')
plt.title("Top 10 Clientes por Receita [Busca Gulosa]")
plt.xlabel("Clientes")
plt.ylabel("Receita Total")
plt.xticks(rotation=45)
plt.show()
```

In [ ]:
```
# 2. Busca Gulosa - Otimizar Tempo de Preparo (Pedidos)

def busca_gulosa_preparo(order_df, top_n=10):
    fila = []
    for _, row in order_df.iterrows():
        tempo = row["preparationTime"] if not pd.isna(row["preparationTime"]) else
        heapq.heappush(fila, (tempo, row["id"]))

    resultado = []
    for _ in range(min(top_n, len(fila))):
        tempo, pedido = heapq.heappop(fila)
        resultado.append((pedido, tempo))
    return resultado

top_preparo = busca_gulosa_preparo(order_df, 10)
plt.figure(figsize=(8,4))
pedidos, tempos = zip(*top_preparo)
plt.scatter(pedidos, tempos, color='green')
plt.title("Top 10 Pedidos por Menor Tempo de Preparo (Busca Gulosa)")
plt.xlabel("Pedido ID")
plt.ylabel("Tempo de Preparo (minutos)")
plt.show()
```

In [ ]:
```
# 3. Busca Gulosa - Priorizar Clientes pelo Ticket Médio

def busca_gulosa_ticket(order_df, top_n=10):
    ticket_medio = order_df.groupby("customerId")["totalAmount"].mean().fillna(0)
```

```
    fila = []
    for cliente, valor in ticket_medio.items():
        heapq.heappush(fila, (-valor, cliente))

    resultado = []
    for _ in range(min(top_n, len(fila))):
        valor, cliente = heapq.heappop(fila)
        resultado.append((cliente, -valor))
    return resultado

top_ticket = busca_gulosa_ticket(order_df, 10)
clientes, tickets = zip(*top_ticket)
plt.figure(figsize=(6,6))
plt.pie(tickets, labels=clientes, autopct="%.1f%%", startangle=90)
plt.title("Distribuição do Ticket Médio entre os Top 10 Clientes")
plt.show()
```

In [ ]:

```
# 4. Busca Gulosa - Campanhas mais engajadas

def busca_gulosa_campanhas(campaign_queue_df, top_n=10):
    engajamento = campaign_queue_df.groupby("campaignId")["response"].count()
    fila = []
    for campanha, resp in engajamento.items():
        heapq.heappush(fila, (-resp, campanha))

    resultado = []
    for _ in range(min(top_n, len(fila))):
        resp, campanha = heapq.heappop(fila)
        resultado.append((campanha, -resp))
    return resultado

top_campanhas = busca_gulosa_campanhas(campaign_queue_df, 10)
campanhas, respostas = zip(*top_campanhas)
plt.figure(figsize=(8,4))
plt.stem(campanhas, respostas, linefmt='skyblue', markerfmt='o', basefmt=" ")
plt.title("Top 10 Campanhas Mais Engajadas (Gráfico Lollipop)")
plt.xlabel("ID da Campanha")
plt.ylabel("Número de Respostas")
plt.show()
```