```
import time
import random
import matplotlib.pyplot as plt

def sort_names(data):
    sorted_data = sorted(data, key=lambda x: (x['First
Name'], x['Last Name']))
    return sorted_data


name_list = [
    {'First Name': 'Raj', 'Last Name': 'Nayyar'},
    {'First Name': 'Suraj', 'Last Name': 'Sharma'},
    {'First Name': 'Karan', 'Last Name': 'Kumar'},
    {'First Name': 'Jade', 'Last Name': 'Canary'},
    {'First Name': 'Raj', 'Last Name': 'Thakur'},
    {'First Name': 'Raj', 'Last Name': 'Sharma'},
    {'First Name': 'Kiran', 'Last Name': 'Kamla'},
    {'First Name': 'Armaan', 'Last Name': 'Kumar'},
    {'First Name': 'Jaya', 'Last Name': 'Sharma'},
    {'First Name': 'Ingrid', 'Last Name': 'Galore'},
    {'First Name': 'Jaya', 'Last Name': 'Seth'},
    {'First Name': 'Armaan', 'Last Name': 'Dadra'},
    {'First Name': 'Ingrid', 'Last Name': 'Maverick'},
    {'First Name': 'Aahana', 'Last Name': 'Arora'}
]


sorted_names = sort_names(name_list)

for person in sorted_names:
    print(f"{person['First Name']} {person['Last Name']}")
```

```
def sort_names(data):
    start_time = time.time()
    sorted_data = sorted(data, key=lambda x: (x['First
Name'], x['Last Name']))
    end_time = time.time()
    return sorted_data, end_time - start_time


data_sizes = list(range(10, 1001, 10))
execution_times = []

for size in data_sizes:

    random_data = [{'First Name': f'First{i}', 'Last
Name': f'Last{i}'} for i in range(size)]


    _, time_taken = sort_names(random_data)
    execution_times.append(time_taken)


plt.plot(data_sizes, execution_times, marker='o')
plt.xlabel('Data Size')
plt.ylabel('Execution Time (seconds)')
plt.title('Execution Time of Sort Algorithm vs Data Size')
plt.grid(True)
plt.show()
```

فراخوانی تابع:
برنامه با فراخوانی تابع sort_names با لیست ورودی دیکشنری ها (داده) شروع می شود.
عملیات مرتب سازی:
در داخل تابع sort_names، تابع sorted برای مرتبسازی فهرست (دادهها) بر اساس کلید مرتبسازی سفارشی استفاده میشود.
کلید مرتبسازی سفارشی با استفاده از یک تابع لامبدا تعریف میشود که «نام» و «نام خانوادگی» را از هر فرهنگ لغت استخراج میکند.
معیارهای مرتب سازی:
تابع مرتبشده، فهرست را در درجه اول بر اساس «نام» و سپس، در صورت وجود نامهای مساوی، بر اساس «نام خانوادگی» مرتب میکند.
تابع لامبدا یک تاپل (x['نام']، x['نام خانوادگی']) برای هر فرهنگ لغت x برمیگرداند که معیارهای مرتبسازی را ارائه میکند.
اندازه گیری زمان:
الگوریتم زمان اجرا را با ثبت زمان شروع قبل از عملیات مرتب سازی و زمان پایان پس از عملیات مرتب سازی اندازه گیری می کند.
زمان صرف شده برای مرتب سازی به عنوان زمان پایان - زمان شروع محاسبه می شود.
برگشت:
تابع لیست مرتب شده (sorted_data) و زمان اجرا را برمی گرداند.
تکرار بر روی اندازه داده ها:
سپس اسکریپت اصلی داده های تصادفی با اندازه های مختلف (از 10 تا 1000 در مراحل 10) تولید می کند.
برای هر اندازه داده، تابع sort_names را فراخوانی می کند، زمان اجرا را اندازه گیری می کند و زمان تجزیه و تحلیل را ثبت می کند.
نمایش چارت:
در نهایت، اسکریپت از matplotlib برای رسم نتایج استفاده می کند. محور x نشان دهنده اندازه داده ها و محور y نشان دهنده زمان اجرای مربوطه است.
به طور خلاصه، الگوریتم شامل مرتبسازی فهرستی از لغت نامهها در درجه اول بر اساس «نام» و در درجه دوم بر اساس «نام خانوادگی» است. زمان اجرا برای اندازههای مختلف داده اندازهگیری میشود و نتایج از طریق یک نمودار برای تجزیه و تحلیل عملکرد الگوریتم با افزایش اندازه دادهها نمایش داده میشوند.