```
# -*- coding: utf-8 -*-
"""Untitled1.ipynb

Automatically generated by Colab.

Original file is located at
    https://colab.research.google.com/drive/1n9PjX7TE6rJJE-
QbrQtV6xdIjVO3Xwce
"""

# ===============================
# Course-End Project
# Analyzing Customer Orders Using Python
# ===============================

# Task 1: Create a list of customer orders & Store the orders
#customer's order details (customer name, product, price, category) as
tuples inside a list
# Dictionary is used, where keys are customer names and values are lists
of ordered products

# Each order = (customer name, product, price, category)
orders = [
    ("John", "Laptop", 900, "Electronics"),
    ("John", "Headphones", 50, "Electronics"),
    ("Jack", "T-Shirt", 25, "Clothing"),
    ("Jack", "Jeans", 40, "Clothing"),
    ("Michael", "Vacuum Cleaner", 120, "Home Essentials"),
    ("Michael", "Laptop", 900, "Electronics"),
    ("Danny", "Notebook", 10, "Home Essentials"),
    ("Danny", "Pen", 5, "Home Essentials"),
    ("Amy", "Smartphone", 700, "Electronics"),
    ("Amy", "Dress", 60, "Clothing")
]

# Dictionary: customer -> list of orders (product, price, category)
customer_orders = {}
for name, product, price, category in orders:
    customer_orders.setdefault(name, []).append((product, price,
category))


# Task 2: Classify products by category ---
# Dictionary: product -> category
product_category = {product: category for _, product, _, category in
orders}

# Unique product categories
categories = set(product_category.values())
print("✅ Available Product Categories:", categories)


# Task 3: Analyze customer spending & classification ---
customer_spending = {}
```

```
customer_classification = {}

for customer, items in customer_orders.items():
    total = sum(price for _, price, _ in items)
    customer_spending[customer] = total

    if total > 100:
        customer_classification[customer] = "High-Value"
    elif 50 <= total <= 100:
        customer_classification[customer] = "Moderate"
    else:
        customer_classification[customer] = "Low-Value"


#Task 4: Generate business insights
# Revenue per category
category_revenue = {}
for _, product, price, category in orders:
    category_revenue[category] = category_revenue.get(category, 0) +
price

# Unique products across all orders
unique_products = {product for _, product, _, _ in orders}

# Customers who purchased electronics
electronics_customers = [name for name, items in customer_orders.items()
                         if any(cat == "Electronics" for _, _, cat in
items)]

# Top 3 highest-spending customers
top_customers = sorted(customer_spending.items(),
                       key=lambda x: x[1], reverse=True)[:3]

# Customers who purchased from multiple categories
multi_category_customers = {name for name, items in
customer_orders.items()
                            if len({cat for _, _, cat in items}) > 1}

# Customers who purchased both electronics and clothing
electronics_buyers = {name for name, items in customer_orders.items()
                      if any(cat == "Electronics" for _, _, cat in
items)}
clothing_buyers = {name for name, items in customer_orders.items()
                   if any(cat == "Clothing" for _, _, cat in items)}
common_customers = electronics_buyers & clothing_buyers


# --- Step 5: Display results ---
print("\n--- Customer Spending & Classification ---")
for customer, total in customer_spending.items():
    print(f"{customer}: ${total} ({customer_classification[customer]})")

print("\n--- Revenue by Category ---")
for cat, revenue in category_revenue.items():
    print(f"{cat}: ${revenue}")

print("\n--- Key Insights ---")
print("Top 3 Customers:", top_customers)
print("Unique Products:", unique_products)
print("Electronics Customers:", electronics_customers)
print("Multi-Category Customers:", multi_category_customers)
print("Common Customers (Electronics & Clothing):", common_customers)
```