```
import csv

def load_data_from_csv(file_path):
    data = []
    try:
        with open(file_path, 'r', newline='') as csvfile:
            csv_reader = csv.reader(csvfile)
            # Skip the header if present
            header = next(csv_reader, None)
            for row in csv_reader:
                data.append(row)
        return header, data
    except FileNotFoundError:
        print(f"Error: File '{file_path}' not found.")
        return None, None
    except Exception as e:
        print(f"An error occurred: {e}")
        return None, None

# Example usage:
file_path = 'device_features.csv'
header, data = load_data_from_csv(file_path)

if header and data:
    print(f"Header: {header}")
    print(f"Data: {data}")
```

```
Header: ['oem_id', 'brand', 'model', 'released_date', 'announced_date', 'hardware_designer', 'manufacturer', 'codename', 'general_extras
Data: [['A135UZKAUSC', 'Samsung', 'SM-A135U Galaxy A13 2022 Standard Edition TD-LTE US 32GB / SM-A135R4', '28-03-22', '04-03-22', 'Samsu
```

```
import csv

def load_data_from_csv(file_path):
    data = []
    try:
        with open(file_path, 'r', newline='') as csvfile:
            csv_reader = csv.DictReader(csvfile)
            for row in csv_reader:
                data.append(row)
        return data
    except FileNotFoundError:
        print(f"Error: File '{file_path}' not found.")
        return None
    except Exception as e:
        print(f"An error occurred: {e}")
        return None

def retrieve_device_info(data, oem_id):
    devices = []
    for device in data:
        if device.get('oem_id') == oem_id:
            device_info = {
                'Model Name': device.get('model_name', 'N/A'),
                'Manufacturer': device.get('manufacturer', 'N/A'),
                'Weight': device.get('weight', 'N/A'),
                'Price': device.get('price', 'N/A'),
                'Price Currency': device.get('price_currency', 'N/A')
            }
            devices.append(device_info)
    return devices

# Example usage:
file_path = 'device_features.csv'
data = load_data_from_csv(file_path)

if data:
    user_oem_id = input("Enter the OEM ID: ")
    devices_info = retrieve_device_info(data, user_oem_id)
    if devices_info:
        print(f"Device(s) Information for OEM ID '{user_oem_id}':")
        for device in devices_info:
            print(device)
    else:
        print(f"No device found for OEM ID '{user_oem_id}'")
```

```
Enter the OEM ID:  A135UZKAUSC
Device(s) Information for OEM ID 'A135UZKAUSC':
{'Model Name': 'N/A', 'Manufacturer': 'Samsung Electronics', 'Weight': 'N/A', 'Price': '199', 'Price Currency': 'USD'}
```

```
import csv

def load_data_from_csv(file_path):
    data = []
    try:
        with open(file_path, 'r', newline='') as csvfile:
            csv_reader = csv.DictReader(csvfile)
            for row in csv_reader:
                data.append(row)
        return data
    except FileNotFoundError:
        print(f"Error: File '{file_path}' not found.")
        return None
    except Exception as e:
        print(f"An error occurred: {e}")
        return None

def retrieve_device_info_by_oem_id(data, oem_id):
    devices = []
    for device in data:
        if device.get('oem_id') == oem_id:
            device_info = {
                'Model Name': device.get('model_name', 'N/A'),
                'Manufacturer': device.get('manufacturer', 'N/A'),
                'Weight': device.get('weight', 'N/A'),
                'Price': device.get('price', 'N/A'),
                'Price Currency': device.get('price_currency', 'N/A')
            }
            devices.append(device_info)
    return devices

def retrieve_device_info_by_code_name(data, code_name):
    devices = []
    for device in data:
        if device.get('code_name') == code_name:
            device_info = {
                'Brand': device.get('brand', 'N/A'),
                'Model Name': device.get('model_name', 'N/A'),
                'RAM Capacity': device.get('ram_capacity', 'N/A'),
                'Market Regions': device.get('market_regions', 'N/A'),
                'Date Added': device.get('date_added', 'N/A')
            }
            devices.append(device_info)
    return devices

def retrieve_device_info_by_ram_capacity(data, ram_capacity):
    devices = []
    for device in data:
        if device.get('ram_capacity') == ram_capacity:
            device_info = {
                'OEM ID': device.get('oem_id', 'N/A'),
                'Release Date': device.get('release_date', 'N/A'),
                'Announcement Date': device.get('announcement_date', 'N/A'),
                'Dimensions': device.get('dimensions', 'N/A'),
                'Device Category': device.get('device_category', 'N/A')
            }
            devices.append(device_info)
    return devices

def retrieve_specific_condition(data, condition_column, condition_value):
    devices = []
    for device in data:
        # Replace 'column1', 'column2', 'column3' with your chosen column names
        if device.get(condition_column) == condition_value:
            device_info = {
                'Column 1': device.get('column1', 'N/A'),
                'Column 2': device.get('column2', 'N/A'),
                'Column 3': device.get('column3', 'N/A')
            }
            devices.append(device_info)
    return devices

# Example usage for each scenario:
file_path = 'device_features.csv'
data = load_data_from_csv(file_path)

# Scenario a1: Retrieve by OEM ID
```

```
if data:
    user_oem_id = input("Enter the OEM ID: ")
    devices_info = retrieve_device_info_by_oem_id(data, user_oem_id)
    if devices_info:
        print(f"Device(s) Information for OEM ID '{user_oem_id}':")
        for device in devices_info:
            print(device)
    else:
        print(f"No device found for OEM ID '{user_oem_id}'")

# Scenario a2: Retrieve by code name
if data:
    user_code_name = input("Enter the Code Name: ")
    devices_info = retrieve_device_info_by_code_name(data, user_code_name)
    if devices_info:
        print(f"Device(s) Information for Code Name '{user_code_name}':")
        for device in devices_info:
            print(device)
    else:
        print(f"No device found for Code Name '{user_code_name}'")

# Scenario a3: Retrieve by RAM capacity
if data:
    user_ram_capacity = input("Enter the RAM Capacity: ")
    devices_info = retrieve_device_info_by_ram_capacity(data, user_ram_capacity)
    if devices_info:
        print(f"Device(s) Information for RAM Capacity '{user_ram_capacity}':")
        for device in devices_info:
            print(device)
    else:
        print(f"No device found for RAM Capacity '{user_ram_capacity}'")

# Scenario a4: Retrieve by specific condition (customize this based on your requirements)
if data:
    user_condition_column = 'column_name'  # Replace with your chosen column name
    user_condition_value = 'desired_value'  # Replace with your desired value
    devices_info = retrieve_specific_condition(data, user_condition_column, user_condition_value)
    if devices_info:
        print("Device(s) Information for Specific Condition:")
        for device in devices_info:
            print(device)
    else:
        print("No device found for the specified condition")
```

```
Enter the OEM ID:  A135UZKAUSC
Device(s) Information for OEM ID 'A135UZKAUSC':
{'Model Name': 'N/A', 'Manufacturer': 'Samsung Electronics', 'Weight': 'N/A', 'Price': '199', 'Price Currency': 'USD'}
Enter the Code Name:  Samsung A135
No device found for Code Name 'Samsung A135'
Enter the RAM Capacity:  3
Device(s) Information for RAM Capacity '3':
{'OEM ID': 'A135UZKAUSC', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.01x6.5x0.35 inches', 'Device Category': 'S
{'OEM ID': 'A135UZKDXAA', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.01x6.5x0.35 inches', 'Device Category': 'S
{'OEM ID': 'A135UZKAVZW', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.01x6.5x0.35 inches', 'Device Category': 'S
{'OEM ID': 'A135UZKAXAU', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.01x6.5x0.35 inches', 'Device Category': 'S
{'OEM ID': 'TA-1560', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.91x6.33x0.34 inches', 'Device Category': 'Smar
{'OEM ID': 'TA-1554', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.91x6.33x0.34 inches', 'Device Category': 'Smar
{'OEM ID': 'TA-1515', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.56x0.34 inches', 'Device Category': 'Smar
{'OEM ID': 'TA-1550', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.56x0.34 inches', 'Device Category': 'Smar
{'OEM ID': 'TA-1500', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.56x0.35 inches', 'Device Category': 'Smar
{'OEM ID': 'TA-1430', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.56x0.35 inches', 'Device Category': 'Smar
{'OEM ID': 'ZAAJ0006US', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '9.89x6.25x0.29 inches', 'Device Category': 'T
{'OEM ID': 'A037WZKAXAC', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.46x0.36 inches', 'Device Category': '
{'OEM ID': 'A042MZKFLTP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.46x0.36 inches', 'Device Category': '
{'OEM ID': 'A042MLBAARO', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.46x0.36 inches', 'Device Category': '
{'OEM ID': 'A042MZCHZTO', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.46x0.36 inches', 'Device Category': '
{'OEM ID': 'A047FZKUEUB', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.48x0.36 inches', 'Device Category': '
{'OEM ID': 'MC3EF', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.94x6.44x0.31 inches', 'Device Category': 'Smartp
{'OEM ID': 'MC3EE', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.94x6.44x0.31 inches', 'Device Category': 'Smartp
{'OEM ID': 'MC3D4', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.95x6.45x0.33 inches', 'Device Category': 'Smartp
{'OEM ID': 'TA-1429', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.49x0.34 inches', 'Device Category': 'Smar
{'OEM ID': 'TA-1401', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.48x0.33 inches', 'Device Category': 'Smar
{'OEM ID': 'J181AP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Table
{'OEM ID': 'J181AP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Table
{'OEM ID': 'J182AP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Table
{'OEM ID': 'J182AP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Table
{'OEM ID': 'J182AP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Table
{'OEM ID': 'J182AP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Table
```

```
{'OEM ID': 'J182AP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Table
{'OEM ID': 'J182AP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Table
{'OEM ID': 'MC36F', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.51x0.37 inches', 'Device Category': 'Smartp
{'OEM ID': 'LMK500MM.AMTPTN', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.57x0.33 inches', 'Device Category
{'OEM ID': 'LMK500UMT3.ATMOTN', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.57x0.33 inches', 'Device Catego
{'OEM ID': 'A125UZKAVZW', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.98x6.46x0.35 inches', 'Device Category': '
{'OEM ID': 'LGL555DL.ATRFTNY', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.57x0.33 inches', 'Device Categor
{'OEM ID': 'M127NZKAKOO', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.46x0.38 inches', 'Device Category': '
{'OEM ID': 'M127FZKJMXO', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.46x0.38 inches', 'Device Category': '
{'OEM ID': 'J171aAP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Tabl
{'OEM ID': 'J172aAP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Tabl
{'OEM ID': 'J172aAP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Tabl
{'OEM ID': 'J172aAP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Tabl
{'OEM ID': 'J171aAP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Tabl
{'OEM ID': 'J172aAP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Tabl
{'OEM ID': 'J172aAP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Tabl
{'OEM ID': 'J172aAP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '6.85x9.87x0.30 inches', 'Device Category': 'Tabl
{'OEM ID': 'LMK500QM7.AUSATN', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.57x0.33 inches', 'Device Categor
No device found for the specified condition
```

```
import csv

def load_data_from_csv(file_path):
    data = []
    try:
        with open(file_path, 'r', newline='') as csvfile:
            csv_reader = csv.DictReader(csvfile)
            for row in csv_reader:
                data.append(row)
        return data
    except FileNotFoundError:
        print(f"Error: File '{file_path}' not found.")
        return None
    except Exception as e:
        print(f"An error occurred: {e}")
        return None

def retrieve_device_info_by_oem_id(data, oem_id):
    devices = []
    for device in data:
        if device.get('oem_id') == oem_id:
            device_info = {
                'Model Name': device.get('model_name', 'N/A'),
                'Manufacturer': device.get('manufacturer', 'N/A'),
                'Weight': device.get('weight', 'N/A'),
                'Price': device.get('price', 'N/A'),
                'Price Currency': device.get('price_currency', 'N/A')
            }
            devices.append(device_info)
    return devices

def retrieve_device_info_by_code_name(data, code_name):
    devices = []
    for device in data:
        if device.get('code_name') == code_name:
            device_info = {
                'Brand': device.get('brand', 'N/A'),
                'Model Name': device.get('model_name', 'N/A'),
                'RAM Capacity': device.get('ram_capacity', 'N/A'),
                'Market Regions': device.get('market_regions', 'N/A'),
                'Date Added': device.get('date_added', 'N/A')
            }
            devices.append(device_info)
    return devices

def retrieve_device_info_by_ram_capacity(data, ram_capacity):
    devices = []
    for device in data:
        if device.get('ram_capacity') == ram_capacity:
            device_info = {
                'OEM ID': device.get('oem_id', 'N/A'),
                'Release Date': device.get('release_date', 'N/A'),
                'Announcement Date': device.get('announcement_date', 'N/A'),
                'Dimensions': device.get('dimensions', 'N/A'),
                'Device Category': device.get('device_category', 'N/A')
            }
            devices.append(device_info)
    return devices

# Example usage:
file_path = 'device_features.csv'
data = load_data_from_csv(file_path)

if data:
    # Example 1: Retrieve device info by OEM ID
    user_oem_id = input("Enter the OEM ID: ")
    devices_info_by_oem_id = retrieve_device_info_by_oem_id(data, user_oem_id)
    if devices_info_by_oem_id:
        print(f"\nDevice(s) Information for OEM ID '{user_oem_id}':")
        for device in devices_info_by_oem_id:
            print(device)
    else:
        print(f"No device found for OEM ID '{user_oem_id}'")

    # Example 2: Retrieve device info by Code Name
    user_code_name = input("Enter the Code Name: ")
    devices_info_by_code_name = retrieve_device_info_by_code_name(data, user_code_name)
```

```
if devices_info_by_code_name:
    print(f"\nDevice(s) Information for Code Name '{user_code_name}':")
    for device in devices_info_by_code_name:
        print(device)
else:
    print(f"No device found for Code Name '{user_code_name}'")

# Example 3: Retrieve device info by RAM Capacity
user_ram_capacity = input("Enter the RAM Capacity: ")
devices_info_by_ram_capacity = retrieve_device_info_by_ram_capacity(data, user_ram_capacity)
if devices_info_by_ram_capacity:
    print(f"\nDevice(s) Information for RAM Capacity '{user_ram_capacity}':")
    for device in devices_info_by_ram_capacity:
        print(device)
else:
    print(f"No device found for RAM Capacity '{user_ram_capacity}'")
```

```
Enter the OEM ID:  A135UZKAUSC

Device(s) Information for OEM ID 'A135UZKAUSC':
{'Model Name': 'N/A', 'Manufacturer': 'Samsung Electronics', 'Weight': 'N/A', 'Price': '199', 'Price Currency': 'USD'}
Enter the Code Name:  Samsung A135
No device found for Code Name 'Samsung A135'
Enter the RAM Capacity:  4

Device(s) Information for RAM Capacity '4':
{'OEM ID': 'MC400', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.94x6.41x0.32 inches', 'Device Category': 'Sma
{'OEM ID': 'MC401', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.94x6.41x0.32 inches', 'Device Category': 'Sma
{'OEM ID': 'TA-1430', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.56x0.35 inches', 'Device Category': 'S
{'OEM ID': 'TA-1476', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.55x0.35 inches', 'Device Category': 'S
{'OEM ID': 'TA-1448', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.55x0.35 inches', 'Device Category': 'S
{'OEM ID': 'ZAAM0094JP', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '9.89x6.25x0.29 inches', 'Device Category':
{'OEM ID': 'ZAAJ0010US', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '9.89x6.25x0.29 inches', 'Device Category':
{'OEM ID': 'ZAAN0007TH', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '9.89x6.25x0.29 inches', 'Device Category':
{'OEM ID': 'ZAAJ0061US', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '9.89x6.25x0.29 inches', 'Device Category':
{'OEM ID': 'ZAAN0002SG', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '9.89x6.25x0.29 inches', 'Device Category':
{'OEM ID': 'ZABN0007IN', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '9.89x6.25x0.29 inches', 'Device Category':
{'OEM ID': 'ZAAN0113SE', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '9.89x6.25x0.29 inches', 'Device Category':
{'OEM ID': 'ZABR0000IN', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '9.89x6.25x0.29 inches', 'Device Category':
{'OEM ID': 'A145MLGGMXD', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A145MLGGZTO', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A145PLGDMEA', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A145PLGGMEA', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A145RZSVEUE', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A145FLGGTUR', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A145FLGGTUR', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A145FZSVXID', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A145RLGUEUB', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A146MLGGZTO', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A146BLGDINS', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A047FZKHXSA', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.48x0.36 inches', 'Device Category'
{'OEM ID': 'A146PZKEATS', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A146PZKGEUE', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A146PLGDEUB', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A146WZKAXAC', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A146UZKDXAA', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A146UZKAXAU', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A146UZKAATT', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'A146UZKAVZW', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.07x6.6x0.36 inches', 'Device Category':
{'OEM ID': 'M045FDBGINS', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.46x0.36 inches', 'Device Category'
{'OEM ID': 'M045FDBKINS', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.46x0.36 inches', 'Device Category'
{'OEM ID': 'M136BLGDINS', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.01x6.48x0.35 inches', 'Device Category'
{'OEM ID': 'E135FIDGINS', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.03x6.51x0.37 inches', 'Device Category'
{'OEM ID': 'E135FIDDINS', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.03x6.51x0.37 inches', 'Device Category'
{'OEM ID': 'M135MLBDMXD', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.03x6.51x0.33 inches', 'Device Category'
{'OEM ID': 'M135MZGGZTO', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.03x6.51x0.33 inches', 'Device Category'
{'OEM ID': 'M135FDBPINS', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.03x6.51x0.37 inches', 'Device Category'
{'OEM ID': 'M135FZGVEUE', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.03x6.51x0.33 inches', 'Device Category'
{'OEM ID': 'M135FLBUEUB', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.03x6.51x0.33 inches', 'Device Category'
{'OEM ID': 'A135NZKEKOO', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.01x6.5x0.35 inches', 'Device Category':
{'OEM ID': 'A235NZKOKOO', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.03x6.51x0.33 inches', 'Device Category'
{'OEM ID': 'A042FLBKINS', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '2.99x6.46x0.36 inches', 'Device Category'
{'OEM ID': 'A047MZWELTM', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.48x0.36 inches', 'Device Category'
{'OEM ID': 'A047FZCHINS', 'Release Date': 'N/A', 'Announcement Date': 'N/A', 'Dimensions': '3.02x6.48x0.36 inches', 'Device Category'
```

```
import pandas as pd
# Load data from the CSV file using pandas
file_path = 'device_features.csv'  # Replace 'your_file_path.csv' with your actual file path
data = pd.read_csv(file_path)

# b1. Identify the top 5 regions where a specific brand of devices was sold.
def top_5_regions_for_brand(brand_name):
    brand_data = data[data['brand'] == brand_name]
    top_regions = brand_data['market_regions'].value_counts().head(5)
    return top_regions

specific_brand = 'Samsung'  # Replace 'Your Brand Name' with the desired brand
top_regions = top_5_regions_for_brand(specific_brand)
print(f"Top 5 regions for '{specific_brand}':")
print(top_regions)
```

```
Top 5 regions for 'Samsung':
market_regions
North America
Asia
Asia,Southeast Asia
Africa,Asia,Australia,Central America,Eastern Europe,Europe,Middle East,North America,Oceania,South America,Southeast Asia,Western Europ
North America,South America
Name: count, dtype: int64
```

```
# b1. Analyse the average price of devices within a specific brand, all in the same currency.
def average_price_for_brand(brand_name):
    brand_data = data[data['brand'] == brand_name]
    # Assuming the currency column is named 'Currency', replace it with your actual column name
    unique_currencies = brand_data['price_currency'].unique()

    # Check if there's only one unique currency for the brand
    if len(unique_currencies) == 1:
        average_price = brand_data['Price'].mean()
        print(f"Average price for '{brand_name}' in '{unique_currencies[0]}': {average_price}")
    else:
        print(f"The prices for '{brand_name}' are in different currencies: {', '.join(unique_currencies)}")

specific_brand = 'Samsung'  # Replace 'Your Brand Name' with the desired brand
average_price_for_brand(specific_brand)
```

```
The prices for 'Samsung' are in different currencies: USD, JPY, MXN, BRL, AED, EUR, TRY, MYR, IDR, GBP, HKD, INR, AUD, CAD, KRW, TWD, CN
```

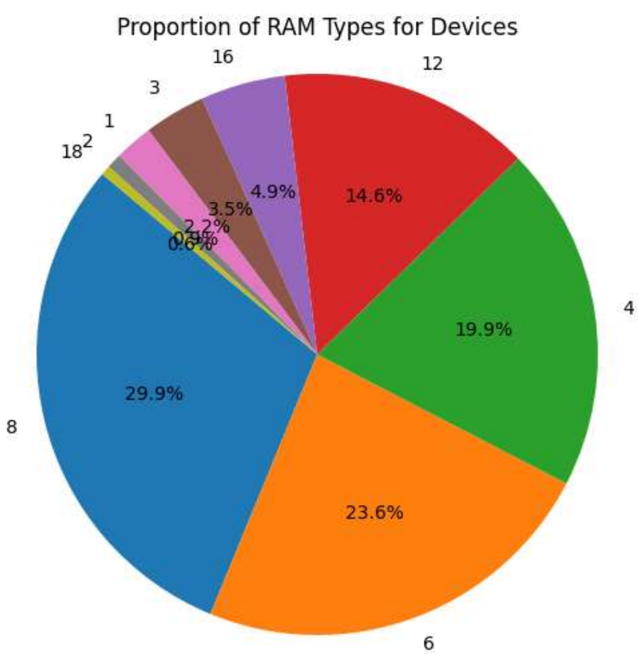
```
# b3. Analyse the average mass for each manufacturer and display the list of average masses for all manufacturers.
def average_mass_per_manufacturer():
    avg_mass = data.groupby('manufacturer')['weight_gram'].mean().sort_values()
    return avg_mass

avg_mass_manufacturer = average_mass_per_manufacturer()
print("Average mass per manufacturer:")
print(avg_mass_manufacturer)
```

```
Average mass per manufacturer:
manufacturer
Luxshare iTech                      31.605000
Hon Hai Precision                  144.000000
Sony                               189.363636
Xiaomi                             190.545455
ZTE                                195.500000
Rising Stars Mobile India          196.500000
Shenzhen TINNO Mobile Technology   198.840000
BBK Electronics                    201.000000
FIH Precision Electronics          203.750000
LG Electronics                     204.075000
Wingtech Mobile                    204.500000
Samsung Electronics                205.644040
Sharp                              209.500000
Lenovo                             216.975904
ASUSTeK Computer                   221.367347
Microsoft                          284.000000
Foxconn                            315.147975
Name: weight_gram, dtype: float64
```

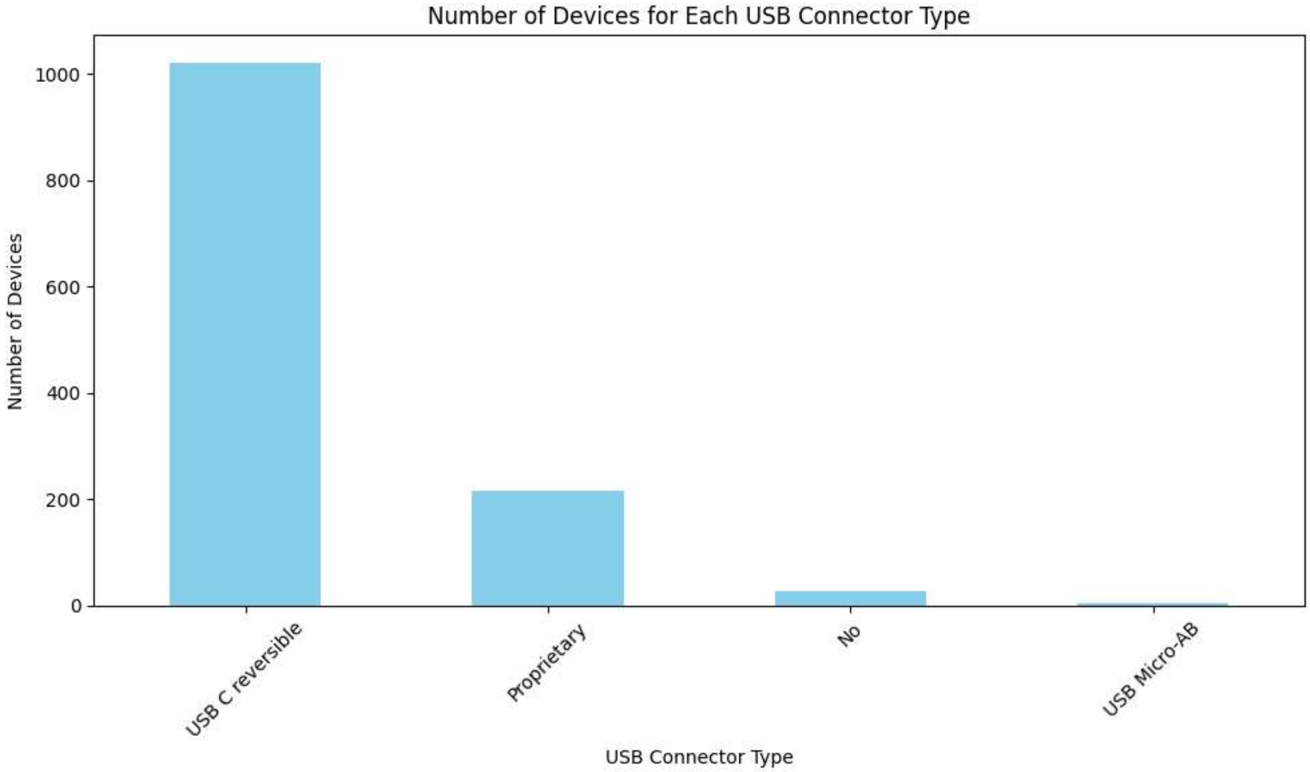
```
#b4. Analyze the data to derive unique insights (Example: Maximum and minimum prices among devices with a specific RAM size)
def insights_based_on_feature(feature_name, feature_value):
    filtered_data = data[data[feature_name] == feature_value]
    if len(filtered_data) > 0:
        max_price = filtered_data['price_currency'].max()
        min_price = filtered_data['price_currency'].min()
        print(f"Maximum price among devices with {feature_name} '{feature_value}': {max_price}")
        print(f"Minimum price among devices with {feature_name} '{feature_value}': {min_price}")
    else:
        print(f"No devices found with {feature_name} '{feature_value}'")

# Example usage:
specific_feature = 'ram_capacity'  # Replace with the feature you want to analyze
specific_value = 8  # Replace with the specific value of the feature you're interested in
insights_based_on_feature(specific_feature, specific_value)
```

```
Maximum price among devices with ram_capacity '8': USD
Minimum price among devices with ram_capacity '8': AED
```

```
# c1. Create a chart to visually represent the proportion of RAM types for devices in the current market.
def visualize_ram_proportion():
    ram_counts = data['ram_capacity'].value_counts()
    # Plotting a pie chart
    plt.figure(figsize=(8, 6))
    plt.pie(ram_counts, labels=ram_counts.index, autopct='%1.1f%%', startangle=140)
    plt.title('Proportion of RAM Types for Devices')
    plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
    plt.show()

visualize_ram_proportion()
```




```
# Create a chart to visually compare the number of devices for each USB connector type.
def visualize_usb_connector_count():
    usb_counts = data['usb_connector'].value_counts()

    # Plotting a bar chart
    plt.figure(figsize=(10, 6))
    usb_counts.plot(kind='bar', color='skyblue')
    plt.title('Number of Devices for Each USB Connector Type')
    plt.xlabel('USB Connector Type')
    plt.ylabel('Number of Devices')
    plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
    plt.tight_layout()
    plt.show()

visualize_usb_connector_count()
```




```
# Convert the 'Release_Date' column to datetime format, specifying the date format
data['Release_Date'] = pd.to_datetime(data['released_date'], format='%d-%m-%y')  # Adjust format if needed

# Filter data for years 2020 to 2023
filtered_data = data[data['Release_Date'].dt.year.between(2020, 2023)]

# Clean 'price_currency' column if it contains mixed currency or non-numeric values
# Example: Extract GBP prices only, assuming they're in numeric format
filtered_data['price_currency'] = pd.to_numeric(filtered_data['price_currency'], errors='coerce')

# Create separate charts for each year showing monthly average price trends in GBP
for year in range(2020, 2024):
    year_data = filtered_data[filtered_data['Release_Date'].dt.year == year]

    if not year_data.empty:
        monthly_avg_prices = year_data.groupby(year_data['Release_Date'].dt.month)['price_currency'].mean()

        plt.figure(figsize=(8, 6))
        plt.plot(monthly_avg_prices.index, monthly_avg_prices.values, marker='o')
        plt.title(f'Monthly Average Price Trends in {year} (GBP)')
        plt.xlabel('Month')
        plt.ylabel('Average Price (GBP)')
        plt.xticks(range(1, 13))  # Set x-axis ticks for each month (1 to 12)
        plt.grid(True)
        plt.show()
    else:
        print(f"No data available for the year {year}")
```