תרגיל Data Analysis עם פתרונות

1. נתונה המחרוזת הבאה:

```
flights="Madrid, 500\nLondon, 600\nBerlin, 400\n"
```

הפכו את הנתונים לטבלה (רשימת מילונים) והדפיסו אותה.

פתרון:

א. נבצע split על התו שורה חדשה כדי לקבל רשימה של שורות:

```
flights=flights.replace(" ", "").split("\n")
```

ב. נגדיר רשימת עזר חדשה שתהיה רשימת המילונים. נבצע לולאה על flights ונהפוך כל שורה למילון, ונוסיף כל מילון לרשימת המילונים:

```
flights_list=[]
for flight in flights:
    בשורה זו מבצעים ספליט כדי לקבל רשימת ערכים של המילון #
    flight_values=flight.replace(" ", "").split(",")
    אלה העמודות של הטבלה #
    flight_keys=["name", "price"]
    מבצעים זיפ של המפתחות והערכים #
    flight_dict=dict(zip(flight_keys, flight_values))
    מוסיפים לרשימת המילונים #
    flights_list.append(flight_dict)

print(flights_list)
```

ג. אפשר להגדיר פונקציה שתבצע את הפעולה האחרונה, היא תקבל את המחרוזת ורשימת העמודות כקלט ותחזיר את רשימת המילונים:

```
def get_table(data_string, columns_list):
    rows_list=data_string.replace(" ", "").split("\n")
    table=[]
    for row in rows_list:
        row_values=row.replace(" ", "").split(",")
        row_keys= columns_list
        row_dict=dict(zip(row_keys, row_values))
        table.append(row_dict)
    return table
```

2. נתונה המחרוזת הבאה:

```
cars="Toyota, 120\nMazda, 130\nSuzuki, 110\n"
```

הפכו את הנתונים לטבלה (רשימת מילונים) והדפיסו אותה.

פתרון:

נשתמש בפונקציה שהגדרנו בפתרון הקודם:

```
print(get_table(cars,["name", "price"]))
```

3. נתונה המחרוזת הבאה:

```
flights="Madrid, 500, August\nLondon, 600, July\nBerlin, 400, June\n"
```

הפכו את הנתונים לטבלה (רשימת מילונים) והדפיסו אותה.

נשתמש בפונקציה שהגדרנו בפתרון הקודם, שימו לב לרשימת העמודות שנותנים כארגומנט:

```
print(get_table(flights,["name", "price", "month"]))
```

4. בטבלה הנ"ל, הדפיסו את המחיר הגבוה ביותר.

פתרון:

א. נשים את הפלט של הפונקציה get_table במשתנה כדי שיהיה לנו קל לעבוד עם המילון:

```
flights_table=get_table(flights,["name", "price", "month"])
```

ב. נחלץ את המחירים לרשימה נפרדת:

```
price_list=[]
for flight in flights_table:
	price_list.append(flight["price"])

print(max(price_list))
```

ג. נהפוך את הקוד לפונקציה שמקבלת טבלה ושם עמודה ומחזירה את המקסימום של העמודה:

```
def get_max(table, col_name):
	values_list=[]
	for row in flights_table:
		values_list.append(row[col_name])
	return max(values_list)

print(get_max(flights_table, "price"))
```

5. בטבלה הנ"ל, הדפיסו את המחיר הנמוך ביותר.

פתרון: כנ"ל רק לשנות את הפונקציה max ל min

6. בטבלה הנ"ל, הדפיסו את היעד של המחיר הגבוה ביותר.

פתרון:

אפשרות א:
נגדיר פונקציה חדשה שמחזירה את היעד, ומקבלת את המקסימום:

```
def get_dest(max):
	for flight in flights_table:
		if flight["price"]=max:
			return flight["name"]

max=get_max(flights_table, "price")
print(get_dest(max))
```

7. בטבלה הנ"ל, הדפיסו את היעד של המחיר הנמוך ביותר.

פתרון: כנ"ל רק עם min, רצוי להגדיר פונקציה חדשה get_min באותו אופן שעשינו.

8. בטבלה הנ"ל, הדפיסו את החודש בו המחיר הכי זול.

פתרון:

```
def get_month(min):
	for flight in flights_table:
		if flight["price"]=min:
			return flight["month"]

min=get_min(flights_table, "price")
print(get_month(min))
```

9. בטבלה הנ"ל, הדפיסו את היעד של המחיר הכי זול, ואת המחיר עצמו.

פתרון:

```
def get_dest(min):
	for flight in flights_table:
		if flight["price"]=min:
			return flight["name"], min

min=get_min(flights_table, "price")
print(get_dest(min))
```

10. בטבלה הנ"ל, הדפיסו את החודשים בהם המחיר נמוך מ 600.

פתרון:

```
def get_cheap_months(max_price):
	months=[]
	for flight in flights_table:
		if int(flight["price"])<max_price:
			months.append(flight["month"])

print(get_cheap_months(600))
```

11. בטבלה הנ"ל, הדפיסו את היעדים בהם המחיר גבוה מ 450.

פתרון:

כנ"ל רק התנאי הוא גדול מ 450 והערכים הם היעדים ולא החודשים:

```
def get_high_months(min_price):
	dests=[]
	for flight in flights_table:
		if int(flight["price"])>min_price:
			dests.append(flight["name"])

print(get_high_months(450))
```

12. מיקי רוצה לטוס למדריד באוגוסט והתקציב שלו 400. הדפיסו אם יש טיסות מתאימות, אם לא הדפיסו את פרטי הטיסה שהכי מתאימה לתקציב.

```
def find_flight(dest, price):
	flights=[]
	for flight in flights_table:
		if flight["price"]==price and flight["dest"]==dest:
			flights.append(flight)
	return flights


def get_flight(price)
	for flight in flights_table:
		if flight["price"]==price:
			return flight

matching_flights=find_flight("London", 500)
if matching_flights==[]:
	print("sorry, no flights, here is what we found for this price:", get_flight(500))
else:
	print(matching_flights)
```

13. שירה רוצה לטוס לאנגליה ביולי ומבקשת לברר אפשרויות. מה צריך להוסיף לטבלה כדי לבצע את השאילתה המתאימה? רמז: עמודה נוספת

פתרון:

```
flights="Spain, Madrid, 500, August\nEngland, London, 600, July\nGermany, Berlin, 400, June\n"

flights_table=get_table(flights,["country", "city", "price", "month"])

def get_flights_country(country):
	flights=[]
	for flight in flight_table:
		if flight["country"]==country:
			flights.append(flight)
	return flights

print(get_flights_country("England"))
```