Introduction

Opening your comic book store, the Sorcery Society, has been a lifelong dream come true. You quickly diversified your shop offerings to include miniatures, plush toys, collectible card games, and board games. Eventually, the store became more a games store with a selection of this week's newest comic books and a small offering of graphic novel paperbacks. Completing your transformation means offering space for local tabletop gamers. They love to play their favorite RPG, "Abruptly Goblins!" and will happily pay you per chair to secure the space to do it. Unfortunately, planning the game night has fallen to you. If you pick the wrong night, not enough people will come and the game night will be cancelled. You decide it's best that you automate the game night selector to get the most people through the door. First you need to create a list of people who will be attending the game night.

Instructions

Create an empty list called `gamers`. This will be your list of people who are attending game night.

```
In [257]: gamers = []
```

Now we want to create a function that will update this list and add a new gamer to the this `gamers` list. Each `gamer` should be a dictionary with the following keys:

- `"name"`: a string that contains the gamer's full or presumed name. E.g., "Vicky Very"
- `"availability"`: a list of strings containing the names of the days of the week that the gamer is available. E.g., ["Monday", "Thursday", "Sunday"]

Instructions

Create a function called `add_gamer` that takes two parameters: `gamer` and `gamers_list`. The function should check that the argument passed to the `gamer` parameter has both `"name"` and a `"availability"` as keys and if so add `gamer` to `gamers_list`.

```
In [258]: def add_gamer(gamer, gamers_list):
              if gamer['name'] and gamer['availability']:
                  gamers_list.append(gamer)
```

Next we want to add our first gamer! Her name is Kimberly Warner and she's available on Mondays, Tuesdays, and Fridays.

Instructions

1. Create a dictionary called `kimberly` with the name and availability given above.
2. Call `add_gamer` with `kimberly` as the first argument and `gamers` as the second.

```
In [259]: kimberly = {'name':'Kimberly Warner', 'availability':['Monday', 'Tuesday', 'Friday']}
          add_gamer(kimberly, gamers)
```

Great! Let's add a couple more gamers to the list!

```
In [260]: add_gamer({'name':'Thomas Nelson','availability': ["Tuesday", "Thursday", "Saturday"]}, gamers)
          add_gamer({'name':'Joyce Sellers','availability': ["Monday", "Wednesday", "Friday", "Saturday"]}, gamers)
          add_gamer({'name':'Michelle Reyes','availability': ["Wednesday", "Thursday", "Sunday"]}, gamers)
          add_gamer({'name':'Stephen Adams','availability': ["Thursday", "Saturday"]}, gamers)
          add_gamer({'name': 'Joanne Lynn', 'availability': ["Monday", "Thursday"]}, gamers)
          add_gamer({'name':'Latasha Bryan','availability': ["Monday", "Sunday"]}, gamers)
          add_gamer({'name':'Crystal Brewer','availability': ["Thursday", "Friday", "Saturday"]}, gamers)
          add_gamer({'name':'James Barnes Jr.','availability': ["Tuesday", "Wednesday", "Thursday", "Sunday"]}, gamers)
          add_gamer({'name':'Michel Trujillo','availability': ["Monday", "Tuesday", "Wednesday"]}, gamers)
```

Finding the perfect availability

Now that we have a list of all of the people interested in game night, we want to be able to calculate which nights would have the most participation. First we need to create a frequency table which correlates each day of the week with gamer availability.

Instructions

Create a function called `build_daily_frequency_table` that takes no argument returns a dictionary with the days of the week as keys and `0`s for values. We'll be using this to count the availability per night. Call `build_daily_frequency_table` and save the results to a variable called `count_availability`.

```
In [261]: def build_daily_frequency_table():
              return {'Monday':0, 'Tuesday':0, 'Wednesday':0, 'Thursday':0, 'Friday':0, 'Saturday':0, 'Sunday':0}
          count_availability = build_daily_frequency_table()
```

Next we need to count the number of people every night.

Instructions

Write a function called `calculate_availability` that takes a list of gamers as an argument `gamers_list` and a frequency table `available_frequency`. The function should iterate through each gamer in `gamers_list` and iterate through each day in the gamer's availability. For each day in the gamer's availability, add one to that date on the frequency table.

```
In [262]: def calculate_availability(gamers_list, availability_frequency):
              for dictionary in gamers_list:
                  for day in dictionary['availability']:
                      availability_frequency[day] += 1
              return availability_frequency
```

Now let's use these tools to find the best night to run Abruptly Goblins!

Instructions

Call `calculate_availability` with `gamers` and `count_availability`. Print out `count_availability` afterwards.

In [263]:
```
print(calculate_availability(gamers, count_availability))
```

```
{'Monday': 5, 'Tuesday': 4, 'Wednesday': 4, 'Thursday': 6, 'Friday': 3, 'Saturday': 4, 'Sunday': 3}
```

Lastly we need a way to pick the day with the most available people to attend so that we can schedule game night on that night.

Instructions

Write a function `find_best_night` that takes a dictionary `availability_table` and returns the key with the highest number.

In [264]:
```
def find_best_night(availability_table):
    best_day_val = availability_table['Monday']
    best_day = 'Monday'
    for key, value in availability_table.items():
        if value > best_day_val:
            best_day_val = value
            best_day = key
    return best_day
```

Now let's find the best day to host game night.

Instructions

Call `find_best_night` with `count_availability`, store the result in a variable called `game_night`. Print out `game_night` to find out which day it is.

In [265]:
```
game_night = find_best_night(count_availability)
print(game_night)
```

```
Thursday
```

And let's make a list of all of the people who are available that night.

Instructions

- Create a function `available_on_night` that takes two parameters: `gamers_list` and `day` and returns a list of people who are available on that particular day.
- Call `available_on_night` with `gamers` and `game_night` and save the result into the variable `attending_game_night`.
- Print `attending_game_night`.

In [266]:
```
def available_on_night(gamers_list, day):
    people = []
    for dictionary in gamers_list:
        for day_av in dictionary['availability']:
            if day == day_av:
                people.append(dictionary['name'])
    return people

attending_game_night = available_on_night(gamers, game_night)
print(attending_game_night)
```

```
['Thomas Nelson', 'Michelle Reyes', 'Stephen Adams', 'Joanne Lynn', 'Crystal Brewer', 'James Barnes Jr.']
```

Generating an E-mail for the Participants

With the best day for Abruptly Goblins! determined with computer precision, we need to let the attendees know that the game night is on a night they can attend. Let's start by creating a form email to send to each of the participants that we'll fill out with data later.

Instructions

Define a string, called `form_email` with interpolation variables `{name}`, `{day_of_week}`, and `{game}` (in case we decide we want to use this featureset to host a different game night). Use it to tell your gaming attendees the night their Abruptly Goblins! game can be played.

In [267]:
```
form_email = """
Dear {name},

The Sorcery Society is happy to host "{game}" night and wishes you will attend. Come by on {day_of_week} and have a blast!

Magically Yours,
the Sorcery Society
"""
```

Instructions

Create a function `send_email` with three parameters: `gamers_who_can_attend`, `day`, and `game`. Print `form_email` for each gamer in `gamers_who_can_attend` with the appropriate `day` and `game`. Call `send_email` with `attending_game_night`, `game_night`, and `"Abruptly Goblins!"`.

In [268]:
```
def send_email(gamers_who_can_attend, day, game):
    for person in gamers_who_can_attend:
        print(form_email.format(name=person, game=game, day_of_week=day))

send_email(attending_game_night, game_night, 'Abruptly Goblins!')
```

```
Dear Thomas Nelson,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Thursday and have a blast!

Magically Yours,
the Sorcery Society


Dear Michelle Reyes,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Thursday and have a blast!

Magically Yours,
the Sorcery Society


Dear Stephen Adams,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Thursday and have a blast!

Magically Yours,
the Sorcery Society


Dear Joanne Lynn,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Thursday and have a blast!

Magically Yours,
the Sorcery Society


Dear Crystal Brewer,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Thursday and have a blast!

Magically Yours,
the Sorcery Society


Dear James Barnes Jr.,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Thursday and have a blast!

Magically Yours,
the Sorcery Society
```

Afterward

You feel bad for the folks who weren't able to attend on the decided upon game night, and try to use your currently written methods to have a second game night of the week.

Instructions

- Create a list `unable_to_attend_best_night` of everyone in `gamers` that wasn't able to attend game night on `game_night`.
- Create `second_night_availability` frequency table by calling `build_daily_frequency_table`.
- Call `calculate_availability` with `unable_to_attend_best_night` and `second_night_availability`.
- Call `find_best_night` with the now filled-in `second_night_availability`, save the results in `second_night`.

In [269]:
```
unable_to_attend_best_night = []
for dictionary in gamers:
    if dictionary['name'] not in attending_game_night:
        unable_to_attend_best_night.append(dictionary)

second_night_availability = build_daily_frequency_table()
calculate_availability(unable_to_attend_best_night, second_night_availability)
second_night = find_best_night(second_night_availability)

print(second_night)
```

```
Monday
```

Let's send out an email to everyone (whether they can attend the first night or not) whose marked themselves as available on our second game night.

Instructions

- Create the list `available_second_game_night` by calling `available_on_night` with `gamers` and `second_night`
- Let the gamers know by calling `send_email` with `available_second_game_night`, `second_night`, and "Abruptly Goblins!"

In [270]:
```
available_second_game_night = available_on_night(gamers, second_night)
send_email(available_second_game_night, second_night, 'Abruptly Goblins!')
```

```
Dear Kimberly Warner,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Monday and have a blast!

Magically Yours,
the Sorcery Society


Dear Joyce Sellers,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Monday and have a blast!

Magically Yours,
the Sorcery Society


Dear Joanne Lynn,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Monday and have a blast!

Magically Yours,
the Sorcery Society


Dear Latasha Bryan,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Monday and have a blast!

Magically Yours,
the Sorcery Society


Dear Michel Trujillo,

The Sorcery Society is happy to host "Abruptly Goblins!" night and wishes you will attend. Come by on Monday and have a blast!

Magically Yours,
the Sorcery Society
```