 Python Programming

Unit 2 Lab: Control Flow

Overview

Welcome to the second unit lab! Remember, our goal is that at the end of the fifth lab, you're going to have an app that searches for movies or prints out the Rotten Tomatoes rating for any movie a user enters.

Right now, you have a variable to hold a movie title, a variable to hold a movie's rating, and a print statement to show the user.

Next, let's set up the functions and control flow to print out the values of our variables.

Deliverables

You're going to continue building this locally from the last lab. You'll write all of your code in the same `movie_app.py` file.

Run the file from the command line to check your work.

Reminder: On your laptop, you can run the file from your command line with the following command:

```
python movie_app.py
```

> **Hint**: Make sure you are printing something out with the print statement! Otherwise, you won't see any output from running your program!

Requirements:

By the end of this, you will have edited your existing `movie_app.py`. At the top, you will have a variable called `search_or_ratings`.

Your app will always print : The movie Back to the Future has a great rating! The movie Blade has a great rating! The movie Spirited Away has a great rating!

Additionally:

- If `search_or_ratings` is equal to `1`, your app will then print `Back to the Future Blade Spirited Away`
- If `search_or_ratings` is equal to `2`, your app will then print `8`.
- If `search_or_ratings` is equal to `3`, your app will then print `The rating for Back to the Future is 8.`

Directions

You'll augment the code you wrote for the Unit 1 lab, so leave your two variable declarations at the top of your program and don't delete the `print` statement!

1. Our program's going to get pretty complex. Let's have a definite starting point. At the bottom of your program, create one `main` function. From here, we'll call everything else.
2. In programming, if you have a `main` function, you can set it to automatically run when you start the program. In Python, there's a section of code that does this for us. At the very bottom of your file, put this code:

```
if __name__ == "__main__":
    main()
```

1. Now, let's get started! When a user searches for a movie, your program is going to print a whole list of movie titles. But what if we only need to print one title? Create a function called `print_movie_title` that prints out `movie_title`.
2. In `main`, call `print_movie_title`.
    1. Try running your program. Do both your `print` statements show up? We won't keep reminding you, but throughout this lab, run your program every few steps to be sure it's doing what it's supposed to.
3. Let's do the same for the movie rating. Create a function called `print_movie_rating` that prints out `movie_rating`. In `main`, below your call for `print_movie_title`, call `print_movie_rating`.
4. What if a user wants to print the whole sentence? Create a function called `print_single_movie_rating` and move your `print` sentence from Lab #1 into it. Then, call `print_single_movie_rating` from `main`.
5. Right now, your `main` function has three `print` statements in a row. What if a user doesn't want to print all three? Let's give the user a choice. At the very top of your file, by `movie_title` and `movie_rating`, create a variable called `search_or_ratings`. For now, set it to `1`.
6. In `main`, let's create an `if` statement and move our function calls into it. You can then test this out by setting `search_or_ratings` to different values.
    1. If `search_or_ratings` is `1`, call `print_movie_title`.
    2. Otherwise, if `search_or_ratings` is `2`, call `print_movie_rating`.
    3. Otherwise, call `print_single_movie_rating`.
7. Later, we'll have many movies, so for now, let's temporarily hard code a list to use in our program. At the top of your `main` function, create a list called `default_movie_list` and set it to `["Back to the Future", "Blade", "Spirited Away"]` (or some other movies you like!).
8. Let's set a way to print these out. Create a function called `print_all_ratings` that takes in a parameter `movie_list`. In it, loop through each movie in `movie_list` and print `"The movie", movie, "has a great rating!"` Then, in your `main` function, call `print_all_ratings` and pass it `default_movie_list`. Put this above your `if` block; it should happen no matter what.

9. Later, a user can search for a movie and see a whole list of matching titles. For now, let's just make the function with our default list. Create a function called `list_search_results` that takes a parameter `movie_titles`. In the function, loop through the `movie_titles` list and print each title out with four spaces () in front of it (just so it looks a little nicer).
10. Let's think about our new `list_search_results` function. Right now, in `main`, if `search_or_ratings` is set to `1`, we call `print_movie_title`. However, that's when a user is going to want a list of movie titles, not just one movie, right? Change that `if` statement: if `search_or_ratings` is set to `1`, call `list_search_results` with the argument `default_movie_list` instead.
    1. Even though we deleted the call to `print_movie_title`, don't delete the function! We'll use it later, when we only need one movie title.

You're done! Test it out to be sure you match the requirements above. Great job.