day42 Password Checker
Due: Wednesday 12/6/23

Today we are writing a password checking program. The user enters a password and the program tells the user if the password is acceptable. For today's program passwords must meet the following criteria:

- Passwords must be between 8 and 20 characters long inclusive
- Passwords must contain at least one uppercase letter
- Passwords must contain at least one lowercase letter
- Passwords must contain at least one digit (0-9)
- Passwords must contain at least one of the following characters: ! @ # $ % ^ & *
- Passwords may not contain spaces or any other characters not specified

Some tips:

You can access each character in a string like this:

```
for x in password:
```

This puts each character from the variable password into the variable x, one character per loop.

Inside the loop you can use string methods to check if a character is uppercase, lowercase, or a digit. Try these:

- .isupper()
- .islower()
- .isdigit().

These methods return True or False.

For example, this code checks if the single character in x is a number:

```
if x.isdigit():
```

You can tell how long a string is using the len() function. You can tell if a character is one of the weird special characters like this:

```
if x in "!@#$%^&*":
```

You'll need to create variables before the loop to count how many of each thing is in the password. For example, you can create a variable "lower", set it to zero, and then inside the loop, check if a character is a lowercase letter. If it is, add one to lower like this: lower = lower + 1. Then after the loop you can check if lower is still zero. If it is, there weren't any lowercase letters in the password. Do this with lower, upper, digits, special, and other all in a big if/elif/elif/elif/else.

(Continued on the next page)

Program checklist:

- Your program is named day42 password.
- Your program explains the requirements of a password then asks the user to enter a password.
- Your program checks the entered password and reports if the password is acceptable or gives clear errors if the password is not acceptable. If there is more than one problem with the proposed password your code needs to report all of the problems.
- The program asks you if you'd like to run again.
- Include two or more sample runs in triple quotes at the end of your program. The runs should show a password checked with all problems reports, and a password that passes.

Samples:

```
Welcome to the password checker!

Passwords must be 8 to 20 characters long.
At least one digit (0-9) is required.
Both uppercase and lowercase characters are required.
At least one special character !@#$%^&* is required.
No spaces or other characters are allowed.

Please enter a password: go ligers
* * * Sorry, your password is not acceptable.
  Your password must contain at least one digit.
  Your password must contain at least one uppercase letter.
  Your password must contain at least one of these characters: !@#$%^&*
  Your password may not contain spaces or any other characters
  not specified in the instructions.
Run again (y/n)? y

Welcome to the password checker!

Passwords must be 8 to 20 characters long.
At least one digit (0-9) is required.
Both uppercase and lowercase characters are required.
At least one special character !@#$%^&* is required.
No spaces or other characters are allowed.

Please enter a password: GoLigers2022!
Congratulations! Your password: GoLigers2022! is acceptable!
Run again (y/n)? n
I sold your password to hackers, just saying.
```