Congratulations! You passed!

Grade received 100%
Latest Submission Grade 100%
To pass 80% or higher
Go to next item

Weekly challenge 4

Quiz • 50 min

Review Learning Objectives

1. What is debugging? **1 / 1 point**

- The practice of improving code efficiency.
- The practice of calling a function from multiple places in a larger program
- The practice of identifying and fixing errors in code.
- The practice of improving code readability.

Correct

Submit your assignment
Due Jul 2, 11:59 PM +08 Attempts 3 every 24 hours
Try again
Receive grade
To Pass 80% or higher
Your grade 100%
View Feedback
We keep your highest score

Like Dislike Report an issue

2. The purpose of the following code is to search a list. Run this code, analyze its output, and then debug it. (If you want to undo your changes to the code, you can click the Reset button.) **1 / 1 point**

```
def search_list(username)
    for item in username:
        print(item)
search_list(["elarson", "bmoreno", "tshah"])
```

Run
Reset

What is the error related to?

- A missing comma (`,`)
- A missing colon (`:`)
- A missing quotation mark (`"`)
- A misspelled variable

Correct

3. The purpose of the following code is to iterate through a list and print a warning message if it finds `"user3"` in the list. Run this code, analyze its output, and debug it. (If you want to undo your changes to the code, you can click the Reset button.) **1 / 1 point**

```
list = ["user1", "user2", "user3", "user4"]
for user in list:
    if user != "user3":
        print("Warning: user3 should not access the system.")
```

Run
Reset

How can you fix the error?

- Change `"user3"` to `"user1"` in the conditional.
- Change `"user3"` to `"user2"` in the conditional.
- Change the indentation so that the line that prints the warning is not indented.
- Change the `!=` operator to the `==` operator in the conditional.

Correct

4. You did not assign a value to a variable before using it in a conditional. What type of error is this? **1 / 1 point**

- Index out of bounds
- Syntax error
- Exception
- Logic error

Correct

5. When debugging code, what are effective ways to determine which sections of code are working properly? Select all that apply. **1 / 1 point**

- [x] Use a debugger

Correct

- [ ] Add comments in the code
- [ ] Delete blank lines from the code
- [x] Add print statements

Correct

6. If you want to read a file called `"logs.txt"`, which line of code allows you to open this file for purposes of reading it and store it in a variable called `file`? **1 / 1 point**

- `with open(file, "r") as logs.txt:`
- `with open("logs.txt", "r") as file:`
- `with open("logs.txt", file, "r"):`
- `with file.open("logs.txt", "r"):`

Correct

7. You've read a log file into the variable `file_text`. The `file_text` variable contains a string of 50 usernames of employees at your company. In order to pass it into a function that checks the login count of each user, the string should be divided into a list of separate usernames. How do you convert this string into a list and store it in a variable `usernames`? **1 / 1 point**

- `file_text.split() as usernames`
- `usernames = split(usernames, file_text)`
- `usernames = usernames.split(file_text)`
- `usernames = file_text.split()`

Correct

8. Fill in the blank: If you use the `.split()` method to convert a string into a list so that it can be read more easily, this would be an example of _____. **1 / 1 point**

- slicing
- debugging
- dividing
- parsing

Correct

9. What does the following code do? **1 / 1 point**

```
new_format = old_format.read()
```

- Reads the `old_format` variable, which contains a file, and stores it as a string in `new_format`
- Detects certain text patterns in `old_format`
- Inserts the string stored in the `new_format` variable into the file stored in the `old_format` variable
- Prints the contents of `old_format`

Correct

10. You want to check for unusual login activity. Specifically, you want to read a log file that contains information on each login attempt, including whether it failed or was successful. You should then parse the data into a `logins` list, and then you should separate all failed log entries into a separate `failed_logins` list. If you want to automate this through Python, what would be part of your code? Select three answers. **1 / 1 point**

- [x] A `for` loop to iterate through all items in the `logins` list

Correct

- [x] An `if` statement to check if a login attempt failed

Correct

- [ ] A counter variable to keep track of the number of failed logins
- [x] A `split()` function to split the login information into a list

Correct