1. What are the three types of errors you will encounter while debugging? **1 / 1 point**

- ○ Exceptions, logic errors, iterative errors
- ◉ Syntax errors, logic errors, and exceptions
- ○ Logic errors, comment errors, and iterative errors
- ○ Syntax errors, exceptions, and comment errors

✓ Correct

2. The purpose of the following code is to print the characters in a device ID. 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**

```
1  device_id = "p35rv47
2  for char in device_id:
3      print(char)
```

Run

Reset

What is the error related to?

- ◉ A missing quotation mark (")
- ○ A missing double equals sign (==)
- ○ A misspelled variable
- ○ A missing colon (:)

✓ 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**

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

Run

Reset

How can you fix the error?

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

✓ Correct

4. You did not define a function before calling it. What type of error is this? **0 / 1 point**

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

✗ Incorrect
Review the video on strategies for debugging code.

5. Why might you use print statements when debugging code? **1 / 1 point**

- ○ To prevent errors from occurring
- ○ To create error messages
- ○ To add missing syntax to the code
- ◉ To identify which sections of the code are working properly

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

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**

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

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**

- debugging
- slicing
- parsing
- dividing

Correct

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

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

- 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`
- Reads the `old_format` variable, which contains a file, and stores it as a string in `new_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**

- [ ] 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

- [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