SE LAB 5

Name: Ullas M
SRN: PES2UG23CS6661
Section: K

1. Issues table:

Issue	Type	Line(s)	Description	Fix Approach
Mutable default argument	Bug	8	logs=[] shared across multiple function calls	Changed default to None and initialized logs = [] inside the function
Bare except:	Error handling	19	removeItem() used a bare except: which hides real exceptions	Replaced with except KeyError: and added a warning log for missing items
Use of eval()	Security vulnerability	59	eval("print('eval used')") can execute arbitrary code and is unsafe	Removed the eval() call and replaced it with a normal print() statement

File opened without with and encoding	Code quality / Resource handling	26–32	open(file, "r") and open(file, "w") used without with or encoding specification	Used with open(file, "r", encoding="utf-8") and with open(file, "w", encoding="utf-8") for safe file operations
String formatting	convention	12	Using old-style % formatting instead of f-string	Convert to fstring f’{str(datetime.now())} : added {qty} of {item}”
Bare except clause	Warning	19	Expect : catches all exceptions without specifying type	Change to expect KeyError : to catch specific exception

1. Which issues were the easiest to fix, and which were the hardest? Why?

The easiest issues to fix were the simple style problems such as adding docstrings, renaming functions to snake_case, and using “with open()” along with encoding. These changes were straightforward because the tools clearly showed what needed to be corrected.

The hardest issue to fix was removing “eval()”. I had to understand why it was unsafe, since it can execute arbitrary code, and then replace it with a safe alternative that did not affect the output.

2. **Did the static analysis tools report any false positives? If so, describe one example.**

No major false positives were found. The only extra warnings were line-length (E501) from Flake8, which were not real errors but just style or formatting suggestions. The tools did not flag any incorrect or misleading issues.

3. **How would you integrate static analysis tools into your actual software development workflow?**

I would use these tools regularly in my workflow. Before committing code, I would run Pylint, Bandit, and Flake8 locally to check for quality and security issues. I would also add them to a GitHub Actions or CI pipeline so that they automatically check every push and pull request. This helps maintain clean, safe, and consistent code throughout development.

4. What tangible improvements did you observe in the code quality, readability, or potential robustness after applying the fixes?

After fixing all the issues, the code became much more secure, readable, and reliable. The program now handles errors properly, uses safe file operations, and follows consistent naming conventions. It no longer contains the dangerous “eval()” function, and all tools show clean reports. The code is now easier to maintain and less likely to cause runtime or security problems.