Summary Report: Input/Output

Objective:

The objective of this Python code is to capture keyboard input from the user, specifically focusing on numeric and alphabetic characters. The program uses the `curses` library to create a simple text-based interface to collect the user's input and display it in different categories: all key presses, numeric characters only, and alphabetic characters only. The code categorizes the input into distinct groups, including all key presses, numeric characters only, and alphabetic characters only. This user-friendly interface can be utilized in various applications, providing a foundation for collecting and processing specific types of user input efficiently.

Key Components:

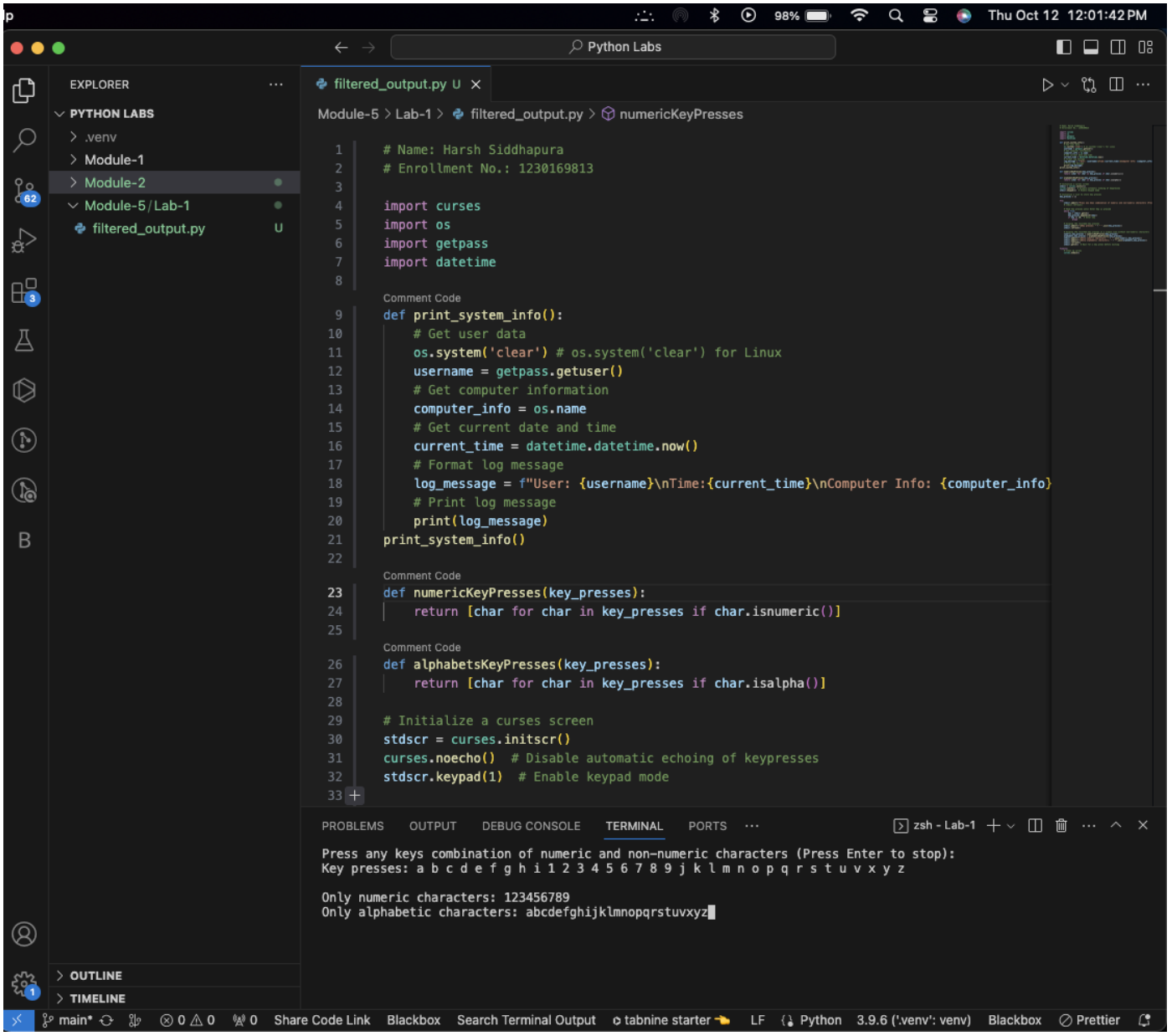
- Function Definitions: The code defines two functions, `numericKeyPresses` and `alphabetsKeyPresses`, which take a list of key presses and filter them to return numeric characters and alphabetic characters, respectively. These functions enhance code modularity and readability.
- Curses Initialization: The code initializes a curses screen using `curses.initscr()`. It also sets certain curses settings to capture user input while disabling automatic echoing of keypresses.
- User Input Capture: The program prompts the user to enter key combinations, with a message displayed at the beginning. It records the key presses until the Enter key is pressed, indicating the user has finished entering keys.
- Filtering and Display: The recorded key presses are then categorized into three groups: all key presses, numeric characters only, and alphabetic characters only. The filtered characters are displayed on the screen along with appropriate labels.
- Curses Cleanup: After the user interaction is complete, the code properly cleans up the curses environment using `curses.endwin()`.

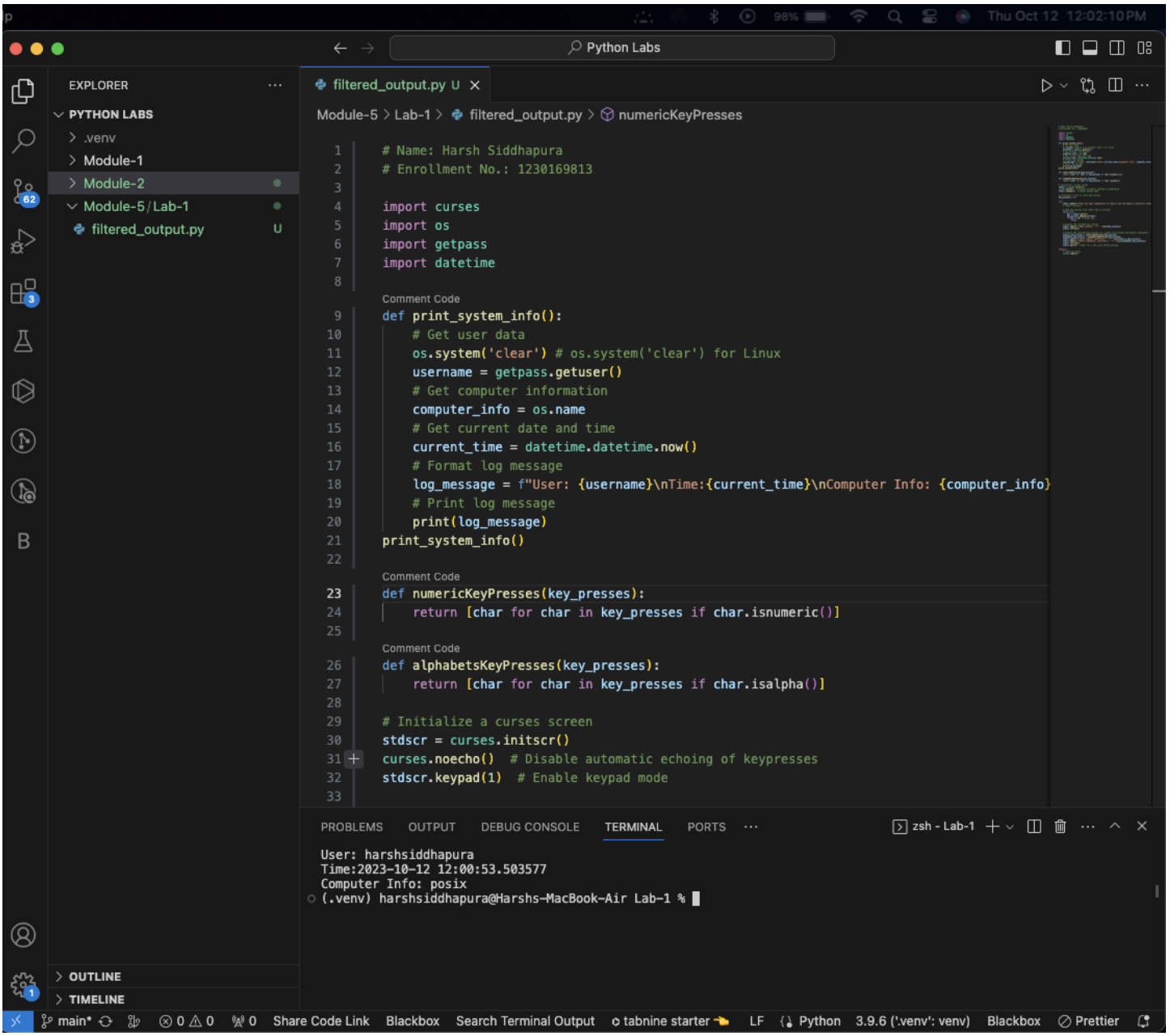
Usage:

This code can be used to capture and categorize keyboard input from the user in a text-based console environment. It is suitable for applications where specific types of user input need to be extracted and processed separately, such as numeric input, alphabetic input, or other specific patterns.

Overall, the code effectively achieves its objective of capturing and categorizing keyboard input using the curses library. It offers a foundation for building text-based applications that require specific user input processing.

Screenshot




Thu Oct 12 12:02:10 PM
Python Labs
EXPLORER
PYTHON LABS
.venv
Module-1
Module-2
Module-5 / Lab-1
filtered_output.py U
filtered_output.py U
Module-5 > Lab-1 > filtered_output.py > numericKeyPresses
# Name: Harsh Siddhapura
# Enrollment No.: 1230169813
import curses
import os
import getpass
import datetime
Comment Code
def print_system_info():
# Get user data
os.system('clear') # os.system('clear') for Linux
username = getpass.getuser()
# Get computer information
computer_info = os.name
# Get current date and time
current_time = datetime.datetime.now()
# Format log message
log_message = f"User: {username}\nTime:{current_time}\nComputer Info: {computer_info}
# Print log message
print(log_message)
print_system_info()
Comment Code
def numericKeyPresses(key_presses):
return [char for char in key_presses if char.isnumeric()]
Comment Code
def alphabetsKeyPresses(key_presses):
return [char for char in key_presses if char.isalpha()]
# Initialize a curses screen
stdscr = curses.initscr()
curses.noecho() # Disable automatic echoing of keypresses
stdscr.keypad(1) # Enable keypad mode
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
zsh - Lab-1
User: harshsiddhapura
Time:2023-10-12 12:00:53.503577
Computer Info: posix
(.venv) harshsiddhapura@Harshs-MacBook-Air Lab-1 %
OUTLINE
TIMELINE
main*
0 0
0
Share Code Link
Blackbox
Search Terminal Output
tabnine starter
LF
Python
3.9.6 ('.venv': venv)
Blackbox
Prettier