**Lab Exercise 2 - Controlling the Keyboard Using Python for RPA**

**Objective**

Learn how to automate keyboard actions using Python for RPA tasks, such as typing text, pressing keys, using key combinations, and interacting with applications programmatically.

**Prerequisites**

1. Python 3.x installed on your system.
2. Install the **pyautogui** library:

pip install pyautogui

1. Install the **keyboard** library for advanced key interactions:

pip install keyboard

1. A basic understanding of Python syntax.

**Exercise Steps**

**Step 1: Typing Text**

1. **Objective**: Simulate typing a string of text.
2. **Code**:

import pyautogui

# Type a string with a delay between keystrokes

pyautogui.typewrite("Hello, this is a keyboard automation demo.", interval=0.1)

print("Text typed successfully.")

1. **Expected Output**:  
   The text will be typed wherever the cursor is focused (e.g., Notepad or a browser input field).

**Step 2: Pressing Individual Keys**

1. **Objective**: Simulate pressing single keys, like Enter, Backspace, and Tab.
2. **Code**:

import pyautogui

# Press Enter

pyautogui.press("enter")

print("Enter key pressed.")

# Press Backspace

pyautogui.press("backspace")

print("Backspace key pressed.")

# Press Tab

pyautogui.press("tab")

print("Tab key pressed.")

1. **Expected Output**:  
   The specified keys will be pressed programmatically.

**Step 3: Key Combinations**

1. **Objective**: Simulate pressing key combinations, like Ctrl+C or Alt+Tab.
2. **Code**:

import pyautogui

# Simulate Ctrl+C (Copy)

pyautogui.hotkey("ctrl", "c")

print("Ctrl+C pressed.")

# Simulate Alt+Tab (Switch window)

pyautogui.hotkey("alt", "tab")

print("Alt+Tab pressed.")

1. **Expected Output**:  
   The script will simulate key combinations for copy and window switching.

**Step 4: Writing a Paragraph**

1. **Objective**: Simulate typing multiple lines of text.
2. **Code**:

import pyautogui

# Write a paragraph

text = """Python is a versatile programming language.

It is widely used for automation, web development, and data science.

This paragraph is typed using Python automation."""

pyautogui.typewrite(text, interval=0.1)

print("Paragraph typed successfully.")

1. **Expected Output**:  
   A paragraph will be typed wherever the cursor is focused.

**Step 5: Pressing Function Keys**

1. **Objective**: Simulate pressing function keys like F5 or Esc.
2. **Code**:

import pyautogui

# Press F5 (e.g., Refresh)

pyautogui.press("f5")

print("F5 key pressed.")

# Press Esc (Escape)

pyautogui.press("esc")

print("Escape key pressed.")

1. **Expected Output**:  
   The script will simulate pressing F5 and Esc.

**Step 6: Automating an Application Interaction**

1. **Objective**: Open Notepad, type a message, and save it using a keyboard shortcut.
2. **Code**:

import pyautogui

import time

# Open the start menu (Windows key)

pyautogui.press("win")

time.sleep(1)

# Type "notepad" and press Enter

pyautogui.typewrite("notepad")

pyautogui.press("enter")

time.sleep(1)

# Type a message in Notepad

pyautogui.typewrite("Hello! This file was created using Python keyboard automation.\n", interval=0.1)

# Save the file using Ctrl+S

pyautogui.hotkey("ctrl", "s")

time.sleep(1)

# Type the file name and press Enter

pyautogui.typewrite("automated\_file.txt")

pyautogui.press("enter")

print("File saved successfully.")

1. **Expected Output**:  
   The script will open Notepad, type a message, and save it as automated\_file.txt.

**Step 7: Detecting Key Presses (Using keyboard Library)**

1. **Objective**: Detect a specific key press and print a message.
2. **Code**:

import keyboard

print("Press 'q' to quit.")

while True:

if keyboard.is\_pressed("q"):

print("You pressed 'q'. Exiting...")

break

1. **Expected Output**:  
   The program will exit when the q key is pressed.