**Lab Exercise 1 - Controlling the Mouse Using Python for RPA Lab**

**Objective**

Learn how to control the mouse using Python for automation tasks, including moving the cursor, clicking, scrolling, and performing drag-and-drop actions.

**Prerequisites**

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

pip install pyautogui

1. A basic understanding of Python syntax.

**Exercise Steps**

**Step 1: Moving the Mouse Cursor**

1. **Objective**: Move the mouse to a specific screen location.
2. **Code**:

import pyautogui

# Get the current screen resolution

screen\_width, screen\_height = pyautogui.size()

print(f"Screen resolution: {screen\_width}x{screen\_height}")

# Move the mouse to the center of the screen

pyautogui.moveTo(screen\_width // 2, screen\_height // 2, duration=1) # Moves in 1 second

print("Mouse moved to the center of the screen.")

1. **Expected Output**:  
   The mouse cursor should move to the center of your screen.

**Step 2: Simulating Mouse Clicks**

1. **Objective**: Perform left, right, and double-click actions.
2. **Code**:

import pyautogui

# Move the mouse to a specific location

pyautogui.moveTo(500, 500, duration=1)

# Perform a left click

pyautogui.click()

print("Left click performed.")

# Perform a right click

pyautogui.rightClick()

print("Right click performed.")

# Perform a double click

pyautogui.doubleClick()

print("Double click performed.")

1. **Expected Output**:  
   The mouse should perform left, right, and double-click actions at the specified location.

**Step 3: Drag and Drop Action**

1. **Objective**: Drag an item from one location to another.
2. **Code**:

import pyautogui

# Drag the mouse from one point to another

start\_x, start\_y = 300, 300

end\_x, end\_y = 600, 600

# Move to the start position

pyautogui.moveTo(start\_x, start\_y, duration=1)

# Perform drag and drop

pyautogui.dragTo(end\_x, end\_y, duration=2, button='left')

print("Drag and drop completed.")

1. **Expected Output**:  
   The mouse cursor should simulate dragging an item from (300, 300) to (600, 600).

**Step 4: Scrolling with the Mouse**

1. **Objective**: Simulate vertical and horizontal scrolling.
2. **Code**:

import pyautogui

# Scroll up

pyautogui.scroll(200) # Scrolls up by 200 units

print("Scrolled up.")

# Scroll down

pyautogui.scroll(-200) # Scrolls down by 200 units

print("Scrolled down.")

1. **Expected Output**:  
   The screen content should scroll up and down.

**Step 5: Detecting Mouse Position**

1. **Objective**: Find the current position of the mouse cursor.
2. **Code**:

import pyautogui

# Get the current mouse position

current\_position = pyautogui.position()

print(f"Current mouse position: {current\_position}")

1. **Expected Output**:  
   The program will print the current (x, y) coordinates of the mouse.

**Step 6: Automating a Simple Task**

1. **Objective**: Automate opening a text editor and typing a message.
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 is an automated message written by Python.\n", interval=0.1)

print("Message written in Notepad.")

1. **Expected Output**:  
   The script will open Notepad, type a message, and display the result.