library:

```
pip install schedule
```

Create a new Python script (e.g., usb_control.py)

```
import os
import time
import schedule

def turn_usb_off():
    os.system("echo '1-1' | sudo tee /sys/bus/usb/drivers/usb/unbind")

def turn_usb_on():
    os.system("echo '1-1' | sudo tee /sys/bus/usb/drivers/usb/bind")

schedule.every().day.at("02:00").do(turn_usb_off)
schedule.every().day.at("05:30").do(turn_usb_on)

schedule.every().day.at("09:30").do(turn_usb_off)
schedule.every().day.at("11:00").do(turn_usb_on)

schedule.every().day.at("15:00").do(turn_usb_off)
schedule.every().day.at("16:30").do(turn_usb_on)

schedule.every().day.at("20:30").do(turn_usb_off)
schedule.every().day.at("22:00").do(turn_usb_on)

while True:
    schedule.run_pending()
    time.sleep(60)
```

This script uses the schedule library to turn the USB ports off and on at the specified times by unbinding and binding the USB root hub. It runs the scheduled tasks every day.

Save the script and make it executable:

```
chmod +x usb_control.py
```

To run the script, open a terminal and enter:

```
sudo python3 usb_control.py
```

Please note that the script needs to be run with superuser privileges since it's accessing the system files for USB control. This script will now keep running in the terminal, controlling the USB ports as specified. To keep the script running in the background even after closing the terminal, use:

```
sudo nohup python3 usb_control.py &
```

This will start the script as a background process, and it will continue running even if you close the terminal.