Daily Algorand ASA Holder Scan, Percentage Allocation, and Token Drop
Step-by-step Operations Guide

1. Overview
This document explains how to run a scheduled daily pipeline that:

1. scans holders of an Algorand Standard Asset (ASA) using an Indexer (scan.py)
2. calculates an allocation percentage per holder with exclusions (percentage.py)
3. copies the allocation CSV into a daily working folder (daily/)
4. executes the token drop from within the daily folder (drop.py)

Note: This guide uses placeholders such as <ASA_ID> and <EXCLUDED_ADDRESS_1>. Replace placeholders with your values.

2. Folder Layout
Recommended directory structure (Windows example):

```
C:\path\to\airdrop\
  ├── venv\
  ├── .env
  ├── scan.py
  ├── percentage.py
  ├── run_daily_airdrop.bat
  └── daily\
    ├── drop.py
    └── (daily CSV outputs)
```

3. Prerequisites
You need:

- Windows 10/11
- Python installed (recommended: Python 3.11)
- Internet access to an Algod endpoint and an Indexer endpoint
- A distribution account holding enough of the ASA plus sufficient ALGO for fees

4. Clean Install (Reset pip/SDKs) with Virtual Environment
From your project folder, create a fresh virtual environment and install only required packages:

```
cd C:\path\to\airdrop
rmdir /s /q venv
python -m venv venv
venv\Scripts\activate
python -m pip install --upgrade pip
pip install py-algorand-sdk python-dotenv
```

Note: Once stable, pin versions in requirements.txt to prevent unexpected upgrades.

5. Configure Secrets in .env
Create a file named .env in the project root (same folder as scan.py). Example:

```
WALLET_MNEMONIC="word word word ... (25 words) ..."
```

Warning: Do not commit .env to Git. Add .env to .gitignore. Never paste real mnemonics into tickets, chats, or screenshots.

6. Configure the ASA and Endpoints
Update these constants inside your scripts:

File	Setting	What to set
scan.py	ASA_ID	<ASA_ID> (integer)

scan.py	INDEXER_URL	Indexer base URL (e.g., https://...)
drop.py	ASA_ID	<ASA_ID> (must match scan.py)
drop.py	ALGOD_ADDRESS	Algod base URL (e.g., https://...)
percentage.py	PERCENT	e.g., 0.01 for 1%

Keep <ASA_ID> consistent across all scripts.

7. Step 1 - Run scan.py (Export holder balances)
Purpose: query the Indexer for all opted-in accounts and export balances into a CSV.

Output file pattern:

asa_<ASA_ID>_holders.csv

Command (from project root):

venv\Scripts\python.exe scan.py

8. Step 2 - Run percentage.py (Compute allocations)
Purpose: read the latest holders CSV, apply exclusions, compute the configured percent, and export allocations.

Input file pattern (holders):

asa_<ASA_ID>_holders.csv

Output file pattern (allocations):

asa_<ASA_ID>_holders_one_percent.csv

Command:

venv\Scripts\python.exe percentage.py

9. Managing Exclusions (Removing Addresses)
Use exclusions to prevent payouts to treasury, burn addresses, contracts, escrow addresses, exchanges, or internal wallets.

In percentage.py, edit EXCLUDED_ADDRESSES and add one address per line:

```
EXCLUDED_ADDRESSES = {
  "<EXCLUDED_ADDRESS_1>",
  "<EXCLUDED_ADDRESS_2>",
  "<EXCLUDED_ADDRESS_3>",
}
```

Tips:

- Keep exclusions in uppercase (the script can normalize case).
- Avoid duplicates (duplicates are harmless but messy).
- Maintain a separate exclude list for change control and audits.

10. Step 3 - Copy the Allocation CSV into daily/
Purpose: freeze the daily payout file and ensure drop.py runs only against the daily folder.

Ensure the daily folder exists:

```
mkdir daily
```

Copy the allocation CSV to daily/ (example in batch):

```
copy asa_<ASA_ID>_holders_one_percent.csv daily\\
```

11. Step 4 - Run drop.py (Token Drop)
Purpose: read the latest *_one_percent.csv in daily/ and execute ASA transfers.

Recommended safety controls in drop.py:

- Dry-run mode (no transactions sent)

- Total required ASA calculation
- Sender ASA balance check (fails before sending if insufficient)
- Per-transaction confirmation and error reporting

Run from inside daily/:

cd daily

..\venv\Scripts\python.exe drop.py

To execute real transfers, set:

DRY_RUN = False

12. Batch Script for Scheduled Runs (run_daily_airdrop.bat)
Use a batch script to run the entire pipeline in order. Step 4 intentionally runs from inside daily/.

```
@echo off
setlocal enabledelayedexpansion

set PROJECT_DIR=C:\path\to\airdrop
set DAILY_DIR=%PROJECT_DIR%\daily
set VENV_DIR=%PROJECT_DIR%\venv
set PYTHON=%VENV_DIR%\Scripts\python.exe

if not exist "%PYTHON%" exit /b 1

"%PYTHON%" "%PROJECT_DIR%\scan.py" || exit /b 1
"%PYTHON%" "%PROJECT_DIR%\percentage.py" || exit /b 1

for %%f in ("%PROJECT_DIR%\*_one_percent.csv") do (
  copy "%%f" "%DAILY_DIR%\" >nul
) || exit /b 1

pushd "%DAILY_DIR%"
```

```
"%PYTHON%" "drop.py"
set EXITCODE=%ERRORLEVEL%
popd

exit /b %EXITCODE%
```

13. Troubleshooting

Common issues and fixes:

Symptom	Fix
percentage.py reads the wrong CSV	Ensure it only selects asa_*_holders.csv (not *_one_percent.csv).
drop.py cannot find CSV	If step 4 runs from daily/, drop.py should search '*_one_percent.csv' (no 'daily/' prefix).
Insufficient ASA balance	Top up the distributor account or implement a daily cap / scaling logic.
Opt-in errors	Holders from the Indexer should be opted-in; if you use a custom list, ensure recipients opt-in first.
SDK import errors	Reset the venv and reinstall only py-algorand-sdk and python-dotenv.

14. Security Practices

- Use a dedicated distribution wallet with limited funds.
- Keep WALLET_MNEMONIC only in .env; never hard-code it.

- Rotate wallets periodically; withdraw leftover funds after runs if desired.
- Back up the mnemonic securely offline (password manager or hardware device).
- Consider multisig or offline signing for large distributions.

Appendix A - Placeholders

Replace these placeholders throughout your scripts and operations:

- <ASA_ID> - The Algorand Standard Asset ID to scan and distribute
- <EXCLUDED_ADDRESS_1>, <EXCLUDED_ADDRESS_2>, ... - Addresses to exclude from payouts
- Algod endpoint - Node URL used to submit transactions
- Indexer endpoint - URL used to list ASA holders