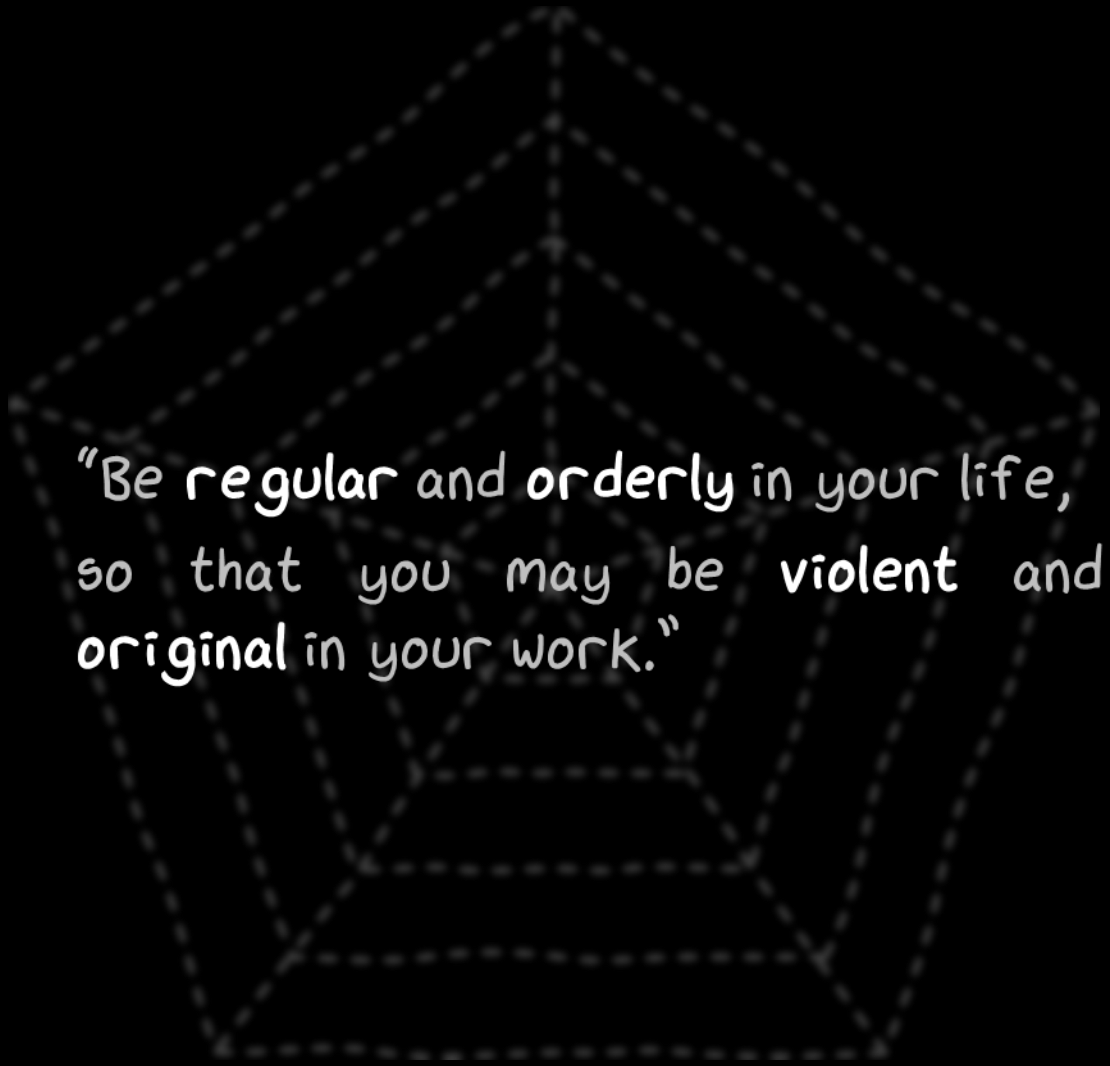


"Be **regular** and **orderly** in your life, so that you may be **violent** and **original** in your work."

1. GENERATE FOLDERS FROM LIST

2. COPY EXCEL SHEET

3. EXTRACTING DATA FROM LAS FILES

4. CONTROLLING MOUSE & KEYBOARD

5. DIGITIZE OLD LOGS

Alvin Alexander
Geotechnician
JX Nippon Oil & Gas Exploration

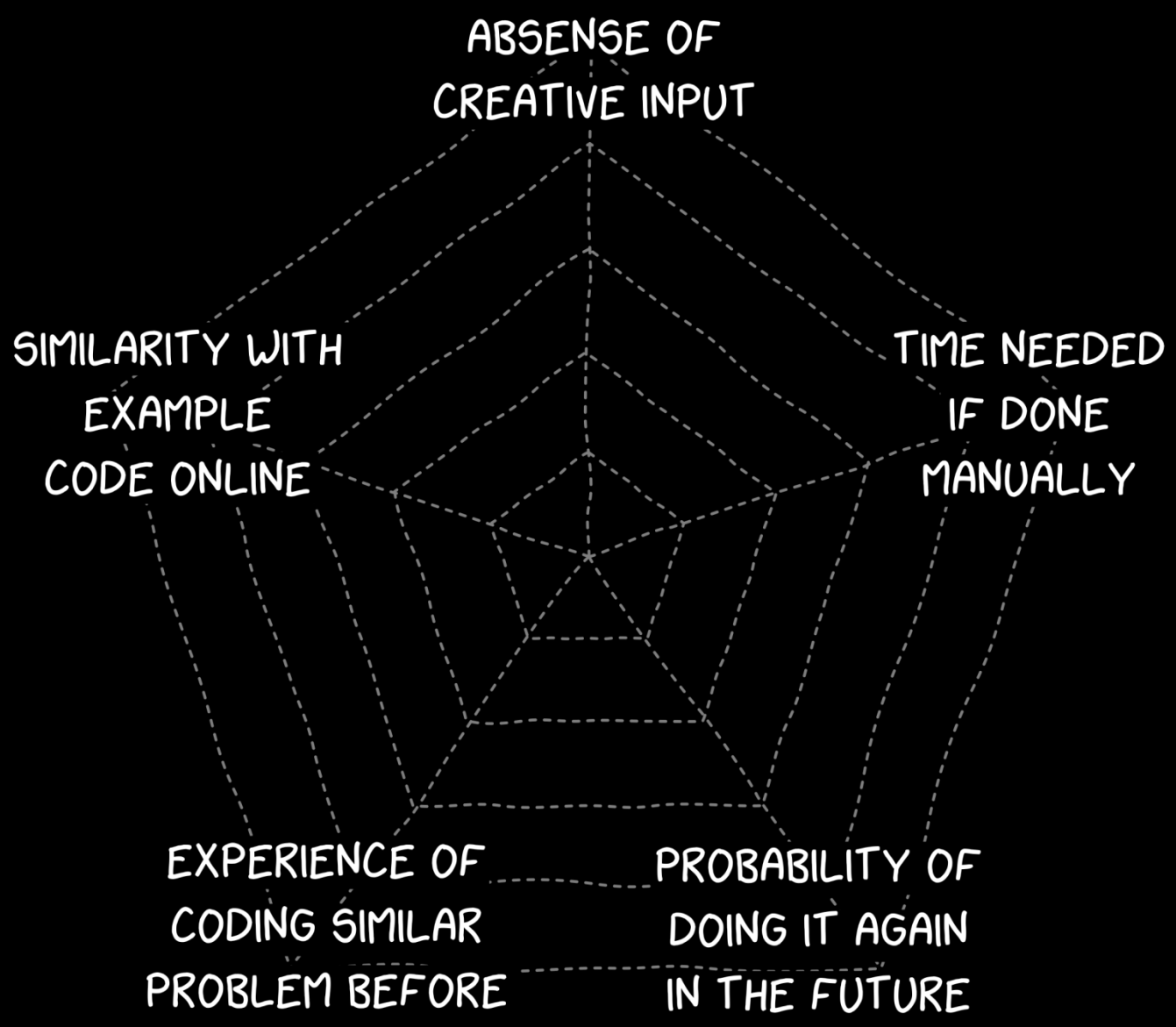
PRACTICAL SCRIPTING IDEAS FOR DATA MANAGERS WITH PYTHON

Why Automate?

1. Menial tasks are boring.
2. HSE concern. Wrist Pain.
3. Minimize mistakes.
4. Time Saving in the long run.
5. Writing code is fun and creative.

Real Examples

WHEN TO AUTOMATE?


ABSENSE OF CREATIVE INPUT
TIME NEEDED IF DONE MANUALLY
PROBABILITY OF DOING IT AGAIN IN THE FUTURE
EXPERIENCE OF CODING SIMILAR PROBLEM BEFORE
SIMILARITY WITH EXAMPLE CODE ONLINE

1. GENERATE FOLDERS FROM LIST

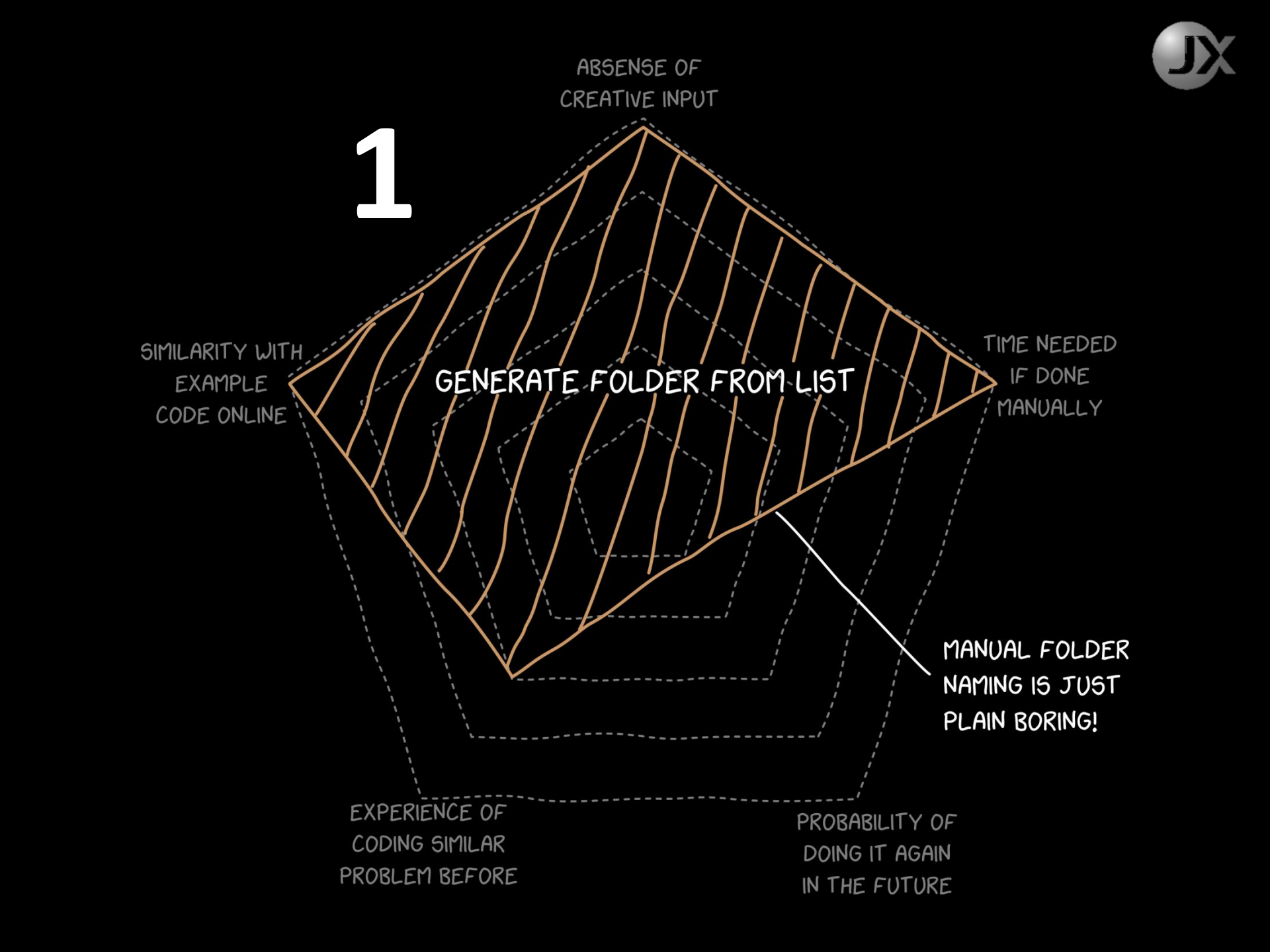
2. COPY EXCEL SHEET

3. EXTRACTING DATA FROM LAS FILES

4. CONTROLLING MOUSE & KEYBOARD

5. DIGITIZE OLD LOGS

Increasing Complexity

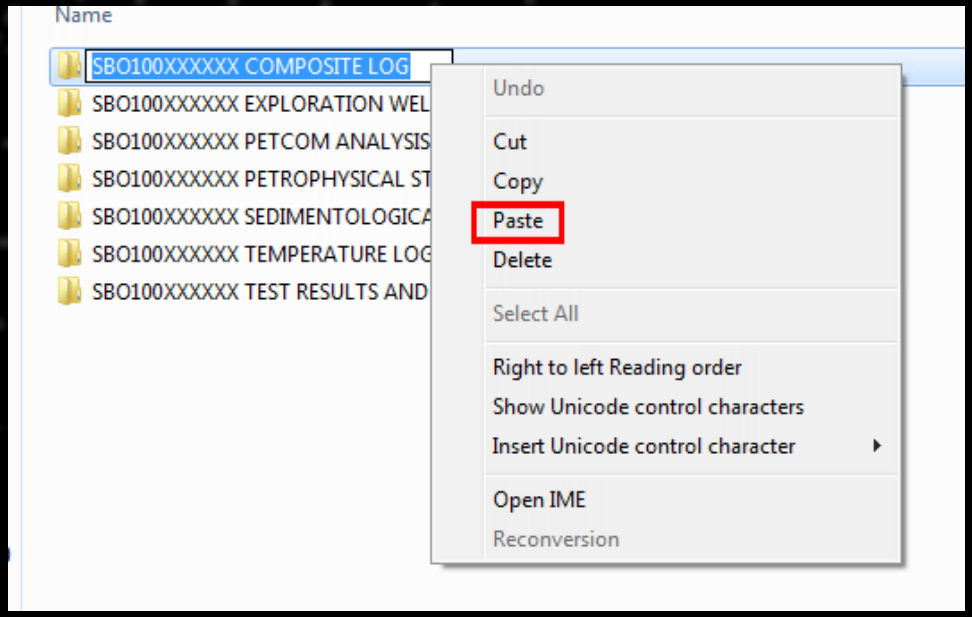

1
ABSENSE OF CREATIVE INPUT
SIMILARITY WITH EXAMPLE CODE ONLINE
GENERATE FOLDER FROM LIST
TIME NEEDED IF DONE MANUALLY
MANUAL FOLDER NAMING IS JUST PLAIN BORING!
EXPERIENCE OF CODING SIMILAR PROBLEM BEFORE
PROBABILITY OF DOING IT AGAIN IN THE FUTURE

List_of_things.txt

```
SBO100XXXXXX COMPOSITE LOG
SBO100XXXXXX EXPLORATION WELL
SBO100XXXXXX PETCOM ANALYSIS
SBO100XXXXXX PETROPHYSICAL STUDY ( ENCLOSURE 10)
SBO100XXXXXX SEDIMENTOLOGICAL FEATURES
SBO100XXXXXX TEMPERATURE LOG
SBO100XXXXXX TEST RESULTS AND PETROPHYSICAL INTERPRETATION SUMMARY
```

Before Automation

1. Open List
2. Copy one item
3. Create Folder
4. Rename and paste
5. Repeat 2 to 4 until finish



The list

List_of_things.txt

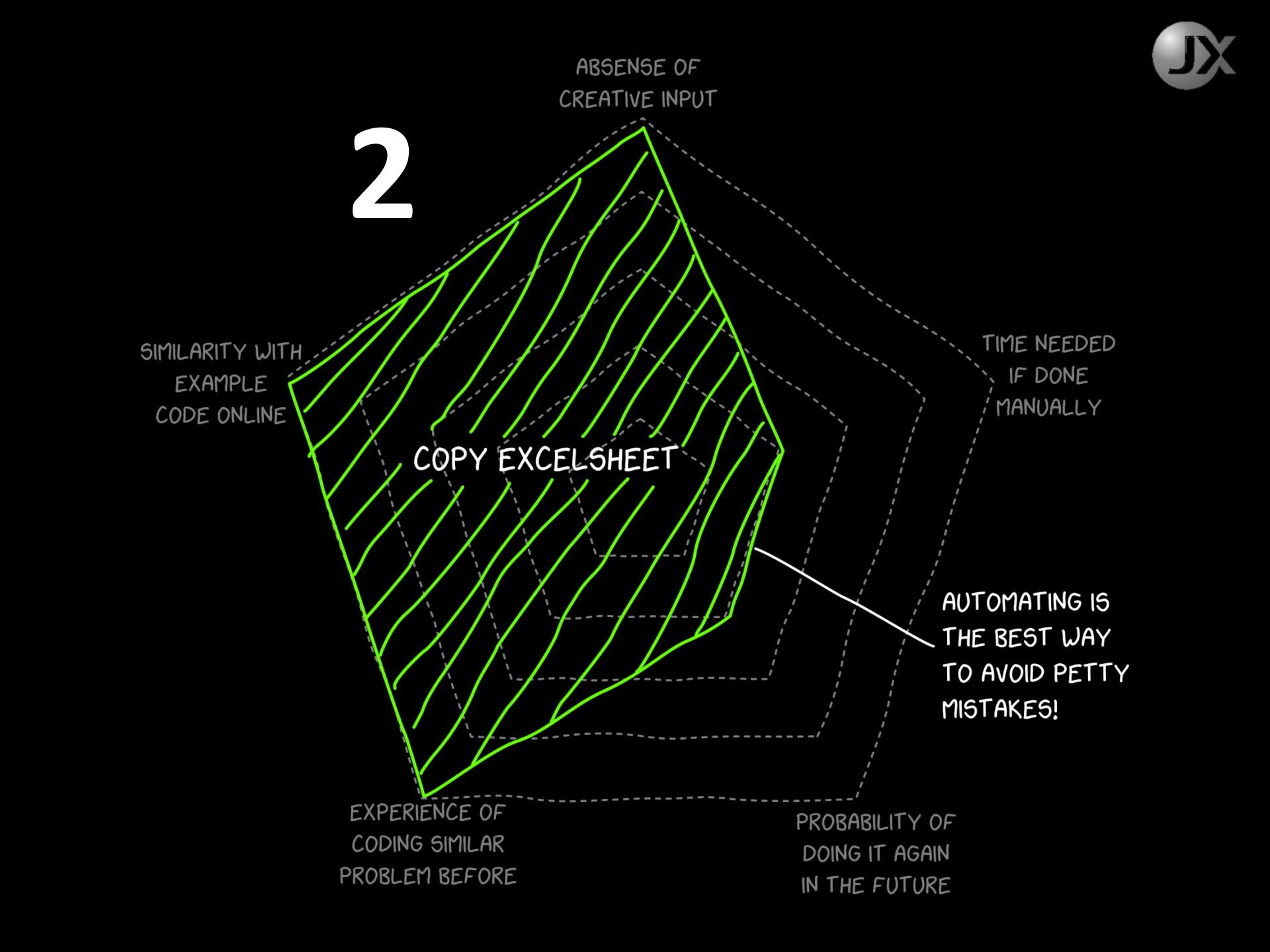
```
SBO100XXXXXX COMPOSITE LOG
SBO100XXXXXX EXPLORATION WELL
SBO100XXXXXX PETCOM ANALYSIS
SBO100XXXXXX PETROPHYSICAL STUDY ( ENCLOSURE 10)
SBO100XXXXXX SEDIMENTOLOGICAL FEATURES
SBO100XXXXXX TEMPERATURE LOG
SBO100XXXXXX TEST RESULTS AND PETROPHYSICAL INTERPRETATION SUMMARY
```

The 5 lines of code

```python
import os
with open("list/List_of_things.txt","r") as lines:
    for line in lines:
        if not os.path.exists(line.strip()):
            os.makedirs(os.path.join(line.strip()))
```

Generated folders

- SBO100XXXXXX COMPOSITE LOG
- SBO100XXXXXX EXPLORATION WELL
- SBO100XXXXXX PETCOM ANALYSIS
- SBO100XXXXXX PETROPHYSICAL STUDY (ENCLOSURE 10)
- SBO100XXXXXX SEDIMENTOLOGICAL FEATURES
- SBO100XXXXXX TEMPERATURE LOG
- SBO100XXXXXX TEST RESULTS AND PETROPHYSICAL INTERPRETATION SUMMARY


2
ABSENSE OF CREATIVE INPUT
SIMILARITY WITH EXAMPLE CODE ONLINE
TIME NEEDED IF DONE MANUALLY
COPY EXCELSHEET
AUTOMATING IS THE BEST WAY TO AVOID PETTY MISTAKES!
EXPERIENCE OF CODING SIMILAR PROBLEM BEFORE
PROBABILITY OF DOING IT AGAIN IN THE FUTURE

1.xlsx

	A	B
1	Name	Value
2	A	4
3	B	10
4	C	8
5	D	2
6	E	6
7	F	4
8	G	1
9		

Sheet1

2.xlsx

	A	B	C
1	Name	Value	
2	A	2	
3	B	0	
4	C	10	
5	D	4	
6	E	3	
7	F	9	
8	G	9	
9			

Sheet1

3.xlsx

	A	B	C
1	Name	Value	
2	A	10	
3	B	5	
4	C	2	
5	D	1	
6	E	4	
7	F	5	
8	G	2	
9			

Sheet1

5.xlsx

	A	B	C
1	Name	Value	
2	A	9	
3	B	8	
4	C	9	
5	D	10	
6	E	10	
7	F	0	
8	G	2	

Sheet1

4.xlsx

	A	B	C
1	Name	Value	
2	A	8	
3	B	2	
4	C	0	
5	D	7	
6	E	4	
7	F	2	
8	G	5	

Sheet1

Before Automation

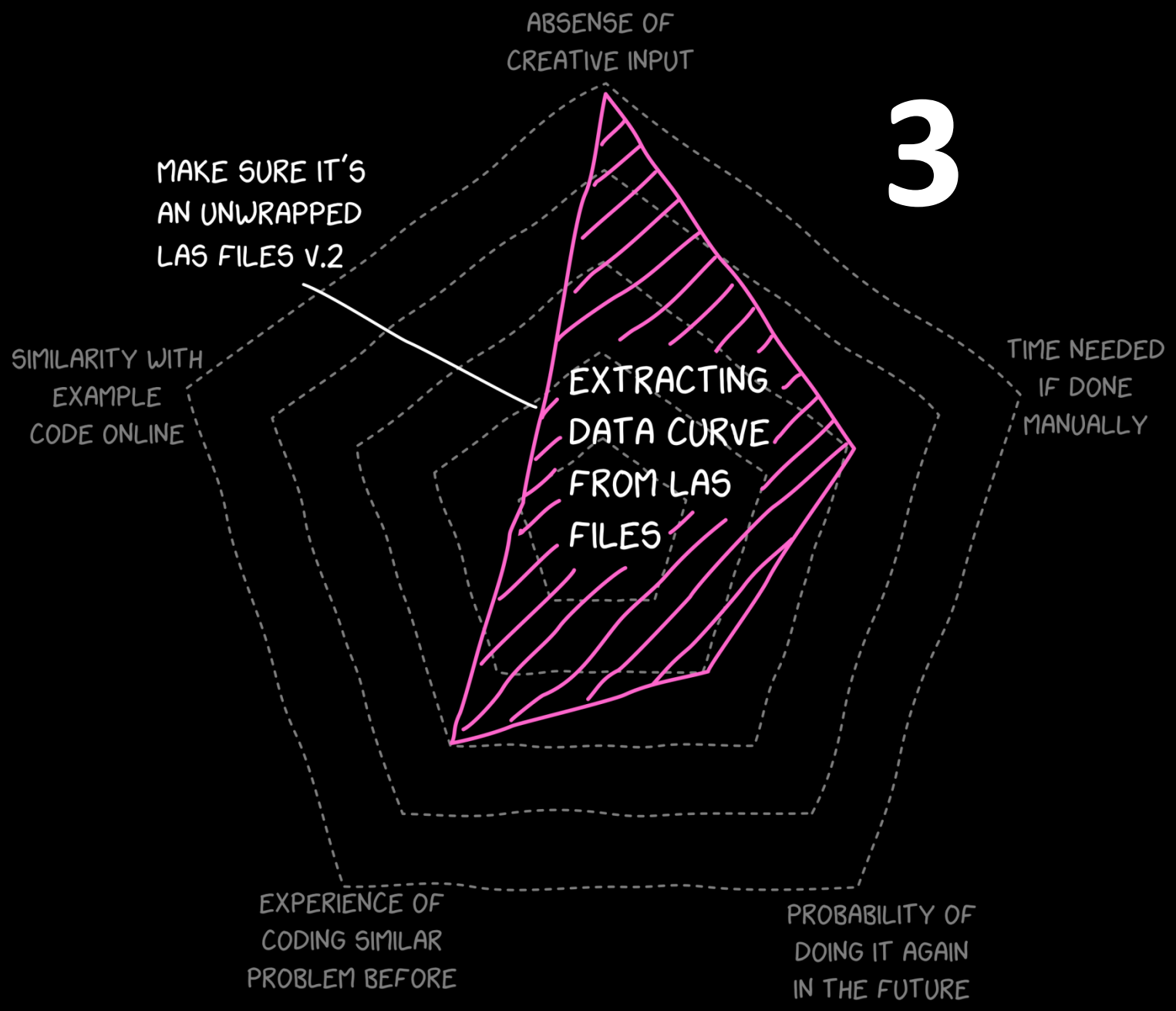
1. Create Destination Excel
2. Open Origin Excel
3. Copy target cells
4. Paste into Destination
5. Close Origin Excel
6. Repeat 2 – 5

Python Script Automation

```python
import openpyxl as xl
from os import listdir

# create new output workbook
out_wb = xl.Workbook()
out_ws = out_wb.active
out_ws.title = "Combined"

# Inputs
files = [(in_folder + f) for f in listdir("Input/")]

in_Col = 2
in_startRow = 2
in_endRow = 8

for f in files:
    in_wb = xl.load_workbook(f, read_only=True)
    in_ws = in_wb.get_sheet_by_name("Sheet1")
    col_name = f.split('/')[1]
    out_col = files.index(f)+2
    # writing the header
    out_ws.cell(row=1,column=out_col).value = col_name

    for i in range(in_startRow,in_endRow+1):
        # copying first column if its first file
        if not files.index(f) and i == in_startRow:
            out_ws.cell(row=1,column=1).value = in_ws.cell(row=1,column=1).value
            out_ws.cell(row=i,column=1).value = in_ws.cell(row=i,column=1).value
        # writing the data rows
        out_ws.cell(row=i,column=out_col).value = in_ws.cell(row=i,column=in_Col).value

# Save the output workbook
out_wb.save("Output/Combined.xlsx")
```

	A	B	C	D	E	F
1	Name	1.xlsx	2.xlsx	3.xlsx	4.xlsx	5.xlsx
2	A	4	2	10	8	9
3	B	10	0	5	2	8
4	C	8	10	2	0	9
5	D	2	4	1	7	10
6	E	6	3	4	4	10
7	F	4	9	5	2	0
8	G	1	9	2	5	2

3


ABSENSE OF CREATIVE INPUT
MAKE SURE IT'S AN UNWRAPPED LAS FILES V.2
SIMILARITY WITH EXAMPLE CODE ONLINE
EXTRACTING DATA CURVE FROM LAS FILES
TIME NEEDED IF DONE MANUALLY
EXPERIENCE OF CODING SIMILAR PROBLEM BEFORE
PROBABILITY OF DOING IT AGAIN IN THE FUTURE

```
#=================================================================
~Version information
VERS.   2.0                    :
WRAP.   NO                     :
#=================================================================
~Well
STRT .ft      2308.0000000  :
STOP .ft      6428.0000000  :
STEP .ft     0.50000000     :
NULL .        -999.250000   :
COMP.                       : COMPANY
WELL.  ---                  : WELL
FLD.                        : FIELD
LOC.                        : LOCATION
SRVC.                       : SERVICE COMPANY
DATE.  2018-05-29 16:19:58  : Log Export Date {yyyy-MM-dd HH:mm:ss}
PROV.                       : PROVINCE
UWI.   42                   : UNIQUE WELL ID
API.                        : API NUMBER
#=================================================================
~Curve
DEPT .ft                    : DEPTH
y_Perm .mD                  : y_Perm
~Parameter
#=================================================================
~Ascii
 2308.0000000  -999.250000
 2308.5000000  -999.250000
 2309.0000000  -999.250000
 2309.5000000  -999.250000
 2310.0000000  -999.250000
 2310.5000000  -999.250000
 2311.0000000  -999.250000
 2311.5000000  -999.250000
 2312.0000000  -999.250000
 2312.5000000  -999.250000
 2313.0000000  -999.250000
 2313.5000000  -999.250000
 2314.0000000  -999.250000
```

Before Automation

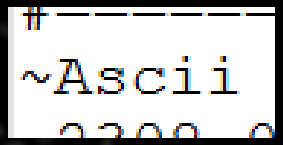
Manual Extract Data From LAS

1. Open LAS using text editor
2. Paste to excel
3. Copy one column
4. Paste into another excel
5. Repeat for all files

…or you can use some software that can read LAS files

Essentially the code do the following,

1. Look for the “~Ascii” keyword



2. Copy data as table into intermediary format
3. Copy data only at specific depth zones
4. Save data table as desired format

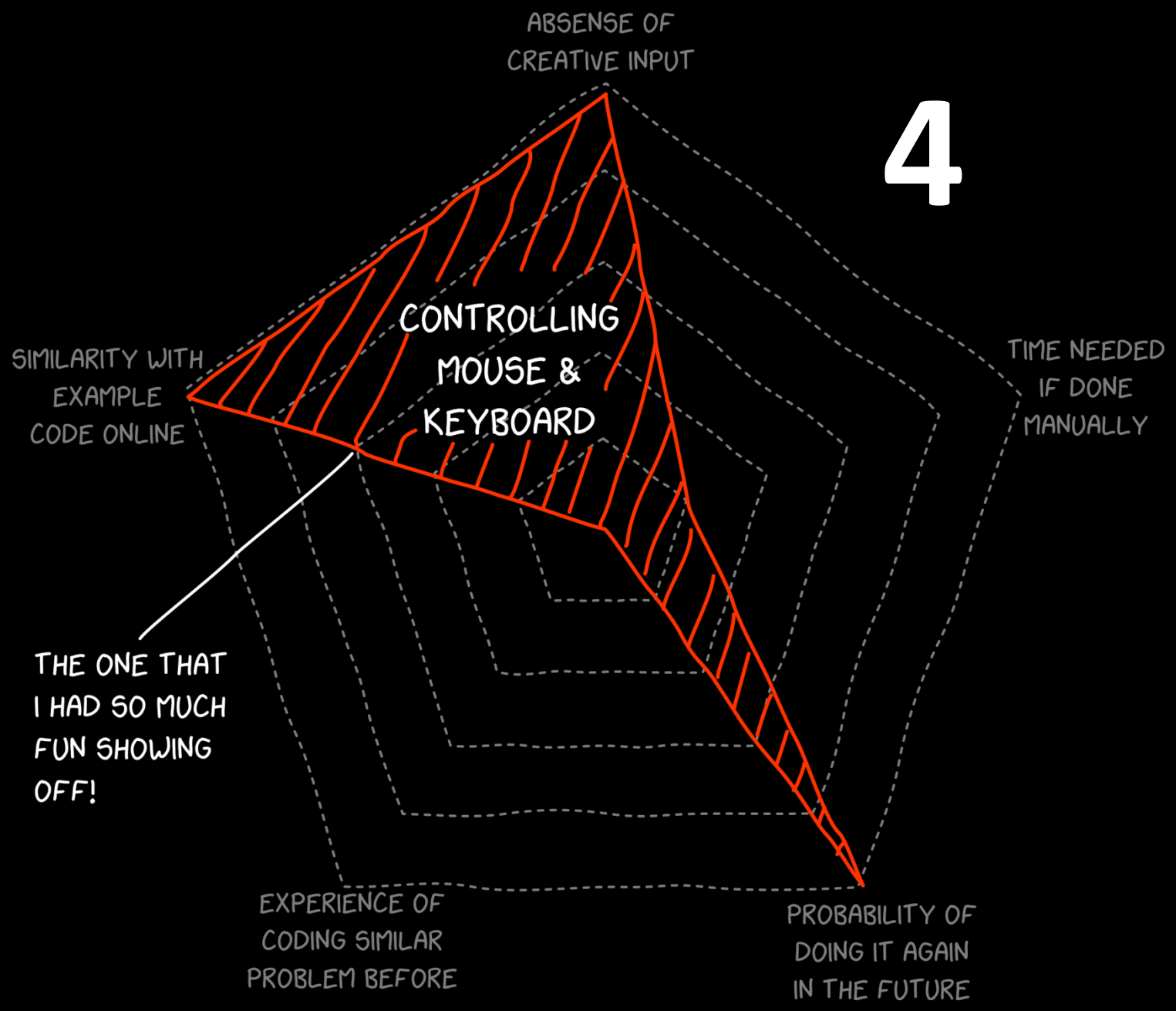
```
#########################################
####          HELPER METHODS          ####

def keyword_line_no(filename,keyword='~Ascii'):
    """
    Returns line number of the first keyword encountered.

    Keyword arguments:
    String -- full path to the ascii file
    String -- search keyword (default '~Ascii')
    """
    count = 1
    with open(filename, "r", encoding="utf-8") as file:
        for line in file:
            if keyword not in line:
                count += 1
                continue
            else:
                break
    return count

def las_df(filename):
    """
    Returns pandas dataframe of las ascii values with depth as index

    Accepts one argument:
    string -- full path to the ascii file
    """
    skiprows = keyword_line_no(filename)
    return pd.read_csv(filename,delim_whitespace=True,skiprows=skiprows,header=None,names=['Depth','Value'])

def zone_x(las_df,top,base):
    """
    Returns pandas dataframe of zone data extracted from LAS dataframe

    Keyword arguments:
    DataFrame -- dataframe of LAS
    float -- top value
    float -- base value
    """
    top = math.floor(top)
    base = math.ceil(base)
    #print("Top: {}, base: {}".format(top,base))
    df = las_df[(las_df.Depth >= top) & (las_df.Depth <= base)]
    df.reset_index(drop=True,inplace=True)
    return df
```

Method to look for keyword

Method to copy data

Method to copy specific depth zones

4


ABSENSE OF CREATIVE INPUT
SIMILARITY WITH EXAMPLE CODE ONLINE
CONTROLLING MOUSE & KEYBOARD
TIME NEEDED IF DONE MANUALLY
THE ONE THAT I HAD SO MUCH FUN SHOWING OFF!
EXPERIENCE OF CODING SIMILAR PROBLEM BEFORE
PROBABILITY OF DOING IT AGAIN IN THE FUTURE

Manual Mouse Movement

1. Movement of mouse depends solely on our hands which is often less than perfect and irregular.

Python Script Automation

```python
import pyautogui
import time
from math import sin,pi
```

```python
# start point
x1,y1 = pyautogui.position()
print(x1,y1)
```

```python
# make a square
pyautogui.moveTo(x1,y1)
pyautogui.click(x1,y1)
basevalue = 200
pyautogui.dragRel(basevalue,0,duration=0.2)
pyautogui.dragRel(0,basevalue,duration=0.2)
pyautogui.dragRel(-basevalue,0,duration=0.2)
pyautogui.dragRel(0,-basevalue,duration=0.2)
```

Perfect square

Python Script Automation

```python
import pyautogui
import time
from math import sin,pi
```

```python
# start point
x1,y1 = pyautogui.position()
print(x1,y1)
```

```python
# make as spiral
pyautogui.moveTo(x1,y1)
basevalue = 30
for i in range(10):
    m = i
    if i%2:
        pyautogui.dragRel(basevalue * i,0,duration=0.2)
        pyautogui.dragRel(0,basevalue * i,duration=0.2)
    else:
        #print("Odd")
        pyautogui.dragRel(-basevalue * i,0,duration=0.2)
        pyautogui.dragRel(0,-basevalue * i,duration=0.2)
```

Perfect spiral

Python Script Automation

```python
import pyautogui
import time
from math import sin,pi
```

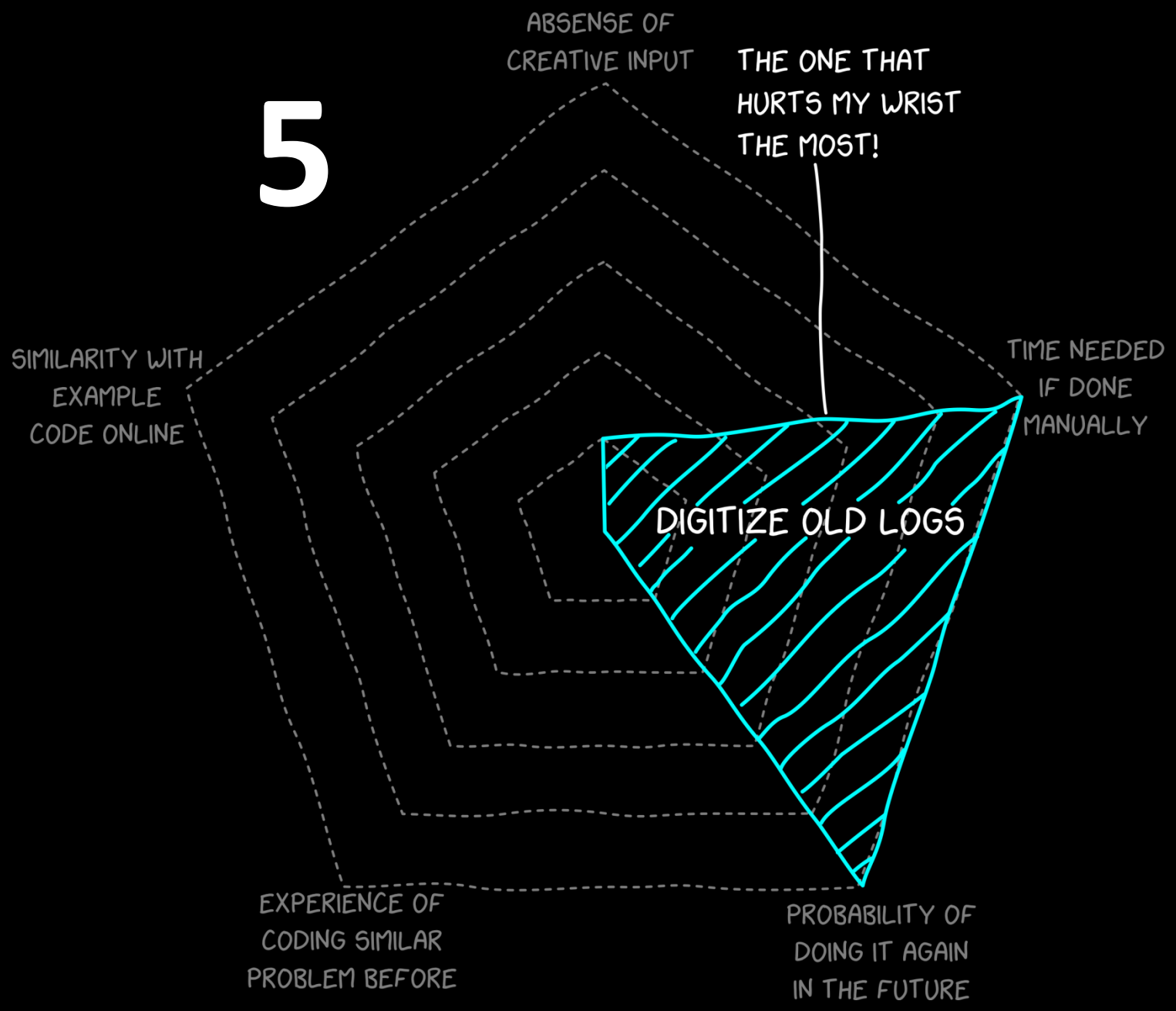
```python
# start point
x1,y1 = pyautogui.position()
print(x1,y1)
```

```python
# Sine wave
pyautogui.moveTo(x1,y1)
height = 20
width = 100
cycle = 5
for x in range(width):
    y = height * sin((2*pi*x)/(width/cycle))
    pyautogui.dragRel(cycle,y)
```

Perfect sine wave

Hand drawn sine wave

5

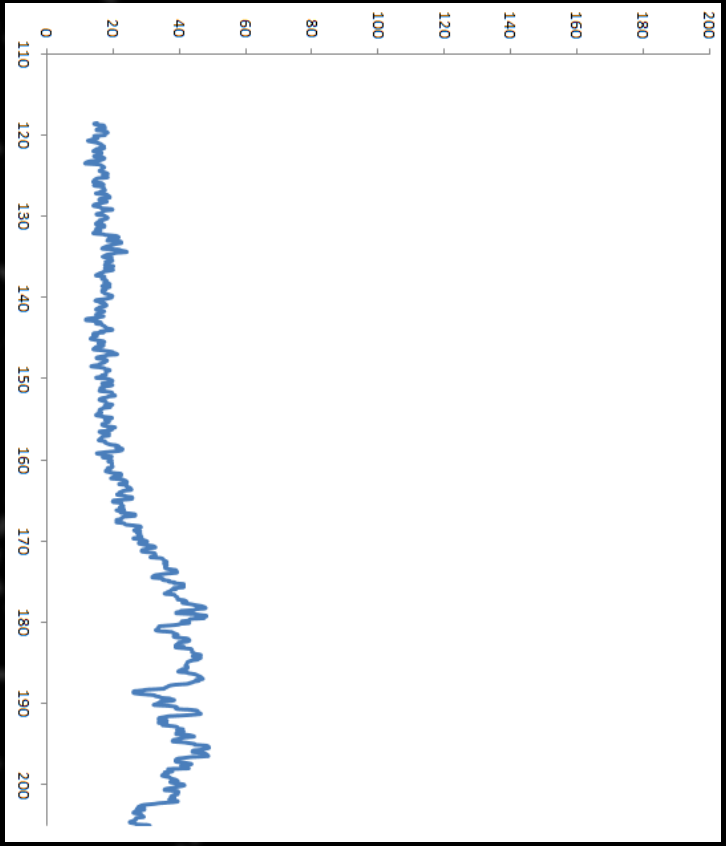

ABSENSE OF CREATIVE INPUT
THE ONE THAT HURTS MY WRIST THE MOST!
SIMILARITY WITH EXAMPLE CODE ONLINE
TIME NEEDED IF DONE MANUALLY
DIGITIZE OLD LOGS
EXPERIENCE OF CODING SIMILAR PROBLEM BEFORE
PROBABILITY OF DOING IT AGAIN IN THE FUTURE

Before Automation

Manually Digitize old Logs

1. Load image into software with
2. Draw to follow the lines as seen on screen

Python Script Automation



Essentially,

1. The script automatically detects the black as data
2. Calculate against the width of the image to determine the value
3. Calculate against the length of the image to determine the depth

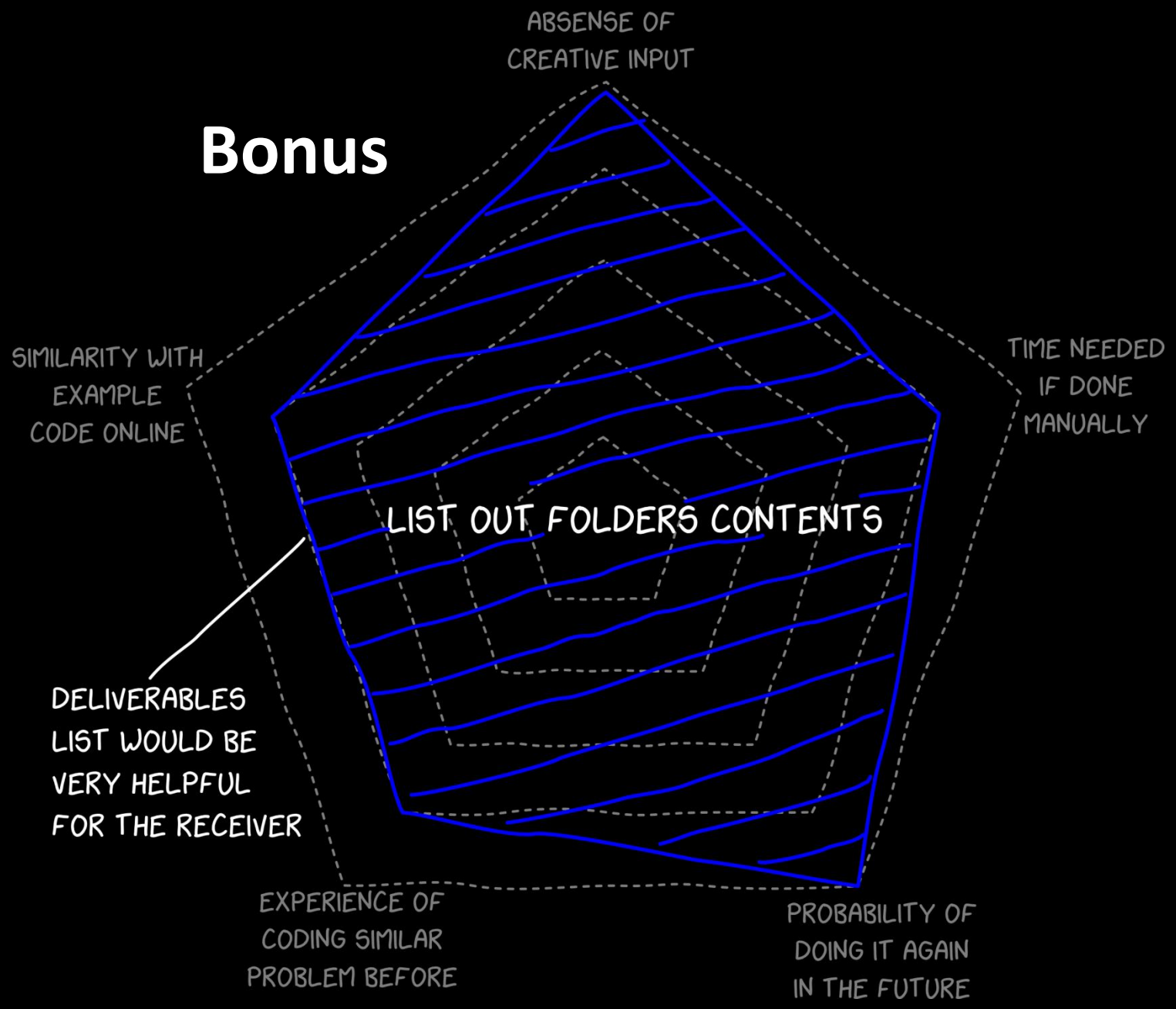
```python
from PIL import Image
import math
import pandas as pd
import numpy as np
```

```python
# Setup Variables
img_file = "Well_1/DTC_100_2730_-450_150.png"
output_file = img_file[:-4] + ".txt"
x_min = -450
x_max = 150
y_min = 100 # in depth unit
y_max = 2730 # in depth unit
step_size = 0.1 # in depth unit
multiples = 3 # lengthening of log
```

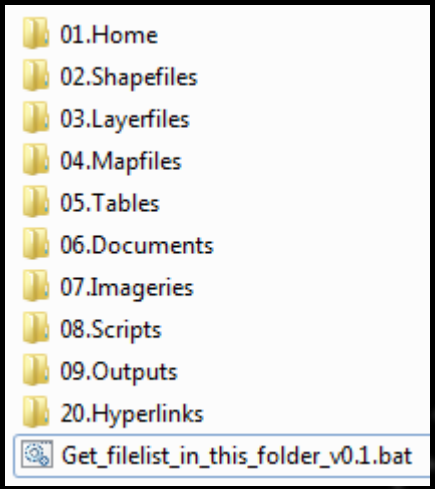
```python
img = Image.open(img_file,mode='r')
width = x_max - x_min
height = y_max - y_min
steps_count = math.ceil(height/step_size)
height = step_size *  steps_count
y_max = y_min + height #main calculation time
img = img.resize((img.width,steps_count * multiples))
pixels_img = [(y,x)
     for y in range(img.height)
     for x in range(img.width)
     if img.getpixel((x,y)) == 0]

df = pd.DataFrame(pixels_img)
dfx = df.groupby([0]).median().rename(columns = {1:'median'})
dfx['max'] = df.groupby([0]).max()
dfx['min'] = df.groupby([0]).min()
dfx['selected'] = np.nan
dfx['pixel_row'] = dfx.index
dfx = dfx[dfx.index % 3 == 0]
dfx = dfx.reset_index(drop=True)
for index, row in dfx.iterrows():
    if index != 0 and index < dfx.shape[0] - 1:
        before = dfx['median'][index - 1]
    else:
        before = dfx['median'][index]

    now = dfx['median'][index]

    if (before == now):
        dfx.set_value(index,'selected',dfx['median'][index])
    elif (before > now):
        dfx.set_value(index,'selected',dfx['min'][index])
    elif (before < now):
        dfx.set_value(index,'selected',dfx['max'][index])

dfx['Depth'] = (dfx['pixel_row'] * (step_size/3) + y_min)
dfx['Value'] = (dfx['selected'] * (width/img.width)) + x_min

dfx_output = dfx[['Depth','Value']]
dfx_output.to_csv(path_or_buf=output_file,sep="\t",index=False)
print("Finished")
```

Bonus Example

Bonus


ABSENSE OF CREATIVE INPUT
TIME NEEDED IF DONE MANUALLY
PROBABILITY OF DOING IT AGAIN IN THE FUTURE
EXPERIENCE OF CODING SIMILAR PROBLEM BEFORE
SIMILARITY WITH EXAMPLE CODE ONLINE
LIST OUT FOLDERS CONTENTS
DELIVERABLES LIST WOULD BE VERY HELPFUL FOR THE RECEIVER

Batch File Automation



Get_filelist_in_this_folder_v0.1.bat - Notepad

File Edit Format View Help

```
:: Get_filelist_in_this_folder.bat
:: This batch file is to output filelist of the current directory.
:: Just double click to use

:: Version 0.1 2017-09-20
@echo off
echo Listing the current files in this folder
echo Output to filelist.txt
echo Please do not close...
echo CTRL+C to cancel

forfiles /s /c "cmd /c if @isdir==FALSE echo @relpath" > filelist.txt
```

C:\Windows\system32\cmd.exe

```
Listing the current files in this folder
Output to filelist.txt
Please do not close...
CTRL+C to cancel
```

filelist.txt - Notepad

File Edit Format View Help

```
".\filelist.txt"
".\Get_filelist_in_this_folder_v0.1.bat"
".\01.Home\Default.gdb\a00000001.gdbindexes"
".\01.Home\Default.gdb\a00000001.gdbtable"
".\01.Home\Default.gdb\a00000001.gdbtablx"
".\01.Home\Default.gdb\a00000001.TablesByName.atx"
".\01.Home\Default.gdb\a00000002.gdbtable"
".\01.Home\Default.gdb\a00000002.gdbtablx"
".\01.Home\Default.gdb\a00000003.gdbindexes"
".\01.Home\Default.gdb\a00000003.gdbtable"
".\01.Home\Default.gdb\a00000003.gdbtablx"
".\01.Home\Default.gdb
\a00000004.CatItemsByPhysicalName.atx"
".\01.Home\Default.gdb
\a00000004.CatItemsByType.atx"
".\01.Home\Default.gdb\a00000004.FDO_UUID.atx"
```

Friendly Tools

- Anaconda Python Distribution
  - Help you manage data science packages for Python
    - For Python it's like
      - Android's "Play Store"
      - Apple's "App Store"

ANACONDA NAVIGATOR

Signed in as mathewlodge Sign out

Home

Environments

Learning

Community

Documentation

Developer Blog

Feedback

Applications on base (root) Channels Refresh

jupyterlab
0.31.12
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
Launch

notebook
5.4.1
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Launch

qtconsole
4.3.1
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Launch

rstudio
1.1.423
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.
Launch

spyder
3.2.8
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

glueviz
0.13.3
Multidimensional data visualization across files. Explore relationships within and among related datasets.

orange3
3.13.0
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

psyplot-gui
1.1.0

Friendly Tools

Jupyter Notebook

Full IDE is usually hard to learn while coding using text file is too tedious

Notebook Cells

- it is so much easy to edit & run specific lines of codes using Notebook cells.

LasDigitize/ | 20180125_Digitize_log_v2

localhost:8888/notebooks/LasDigitize/20...

jupyter 20180125_Digitize_log_v2

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [1]:
```
from PIL import Image
import math
import pandas as pd
import numpy as np
```

In [2]:
```
# Setup Variables
img_file = "Well_1/DTC_100_2730_-450_150.png"
output_file = img_file[:-4] + ".txt"
x_min = -450
x_max = 150
y_min = 100 # in depth unit
y_max = 2730 # in depth unit
step_size = 0.1 # in depth unit
multiples = 3 # lengthening of log
```

In [3]:
```
img = Image.open(img_file,mode='r')
width = x_max - x_min
height = y_max - y_min
steps_count = math.ceil(height/step_size)
height = step_size *  steps_count
```

Final Tips

Scripts should never not be general purpose

Write simple first, then combine

Writing scripts is fun, try it!

"Be **regular** and **orderly** in your life, so that you may be **violent** and **original** in your work."

– GUSTAVE FLAUBERT –

Thank you!

You can have all the codes here.

My Contact Info

Email: alvin@noex.com.my