תרגיל בניית Shell

# שלב א: הכרת המודול subprocess והפעולה Popen

# מטרה

* לימוד הפרמטר args
* לימוד הפרמטר cwd
* לימוד הפרמטר env

# הפרמטר args

יכול להיות רשימה של מחרוזות או מחרוזת בודדת (לא בתוך רשימה)

הרץ את הפקודה:

subprocess.run(args="cmd /c dir c:\windows")

האם הצליח?

הרץ את הפקודה:

subprocess.run(args=["cmd /c dir c:\Program Files"], shell=True)

האם הצליח?

הרץ את הפקודה:

subprocess.run(args=["cmd /c", "dir", "c:\Program Files"], shell=True)

האם הצליח?

הסבר את התוצאות

מומלץ להעזר התיעוד של המודול בפייתון. בפסקה Frequently Used Arguments וכן בפסקה Popen Constructor

תשובה:

# הפרמטר cwd

הדגמת והבנה של המושג current working directory

1. ייצרו סקריפט פייתון פשוט. לדוגמא סקריפט המדפיס למסך Hello
2. מקמו (שמרו) את הסקריפט בספרייה כלשהי, לדוגמא בספרייה c:\temp\folder1
3. פתחו shell (cmd.exe) בספרייה שבה נמצא הסקריפט.
4. הרצו interpreter של פייתון כשאתם נמצאים בספרייה הזו

לצורך הפעולות הבאות נדרש לייבא מספר מודולים: subprocess, os

הרצו את הפקודה:

subprocess.run(args=["python", "example1.py"])

האם הפקודה רצה כמצופה?

הרצו את הפקודה:

os.getcwd()

מה היא החזירה? מה היא עושה?

הרצו את הפקודה:

os.chdir(r'C:\temp\folder2')

ואחריה את הפקודה:

os.getcwd()

הסבירו מה היא עושה:

הרצו את הפקודה:

subprocess.run(args=["python", "example1.py"])

האם הפקודה הצליחה? הסבירו.

הרצו את הפקודה:

subprocess.run(args=["python", "example1.py"], cwd=r'C:\temp\folder1')

הסבירו:

# הפרמטר env

יש לקרוא את ההסבר מה הפרמטר הזה עושה ולהדגים.

את ההדגמה והסבר שלכם יש לפרט פה.

# העמקה והבנה של הפקודות run, Popen

מה לדעתך ההבדל בין 2 הפעולות?

subprocess.run(args="cmd /c dir c:\windows")

subprocess.Popen(args="cmd /c dir c:\windows")

הסבר:

יכול להיות שלא שמתם לב להבדלים בין 2 ההרצות של הפקודות הנ"ל.

אולי שני קטעי הקוד הבאים יעזרו להם להבין.

ראשית, מה ההבדל בין שני קטעי הקוד הבאים לבין שתי השורות הקודמות

f = open("dir.txt", 'w')

subprocess.run(args="cmd /c dir c:\windows /s", stdout=f)

f.close()

f = open("dir.txt", 'w')

subprocess.Popen(args="cmd /c dir c:\windows /s", stdout=f)

f.close()

הסבר:

ועכשיו הסבר מה ההבדל בין שתי קטעי הקוד האחרונים:

איך ניתן לדעת מתי הפקודה Popen הסתיימה? קראו והסבירו wait, poll, communicate

תשובה:

קראו על הפקודות הבאות והסבירו אותן. אם יש צורך שלבו בהסברים ודוגמאות.

**send\_signal**

**terminate**

**kill**

תהליכים (process) (תוכניות (exe) או סקריפטים) יכולים להחזיר תשובה (return code) בסיום הריצה של התהליך.

כתבו סקריפט הכי פשוט שמחזיר תשובה עם ערך 0 ופעם עם ערך 1

הראו איך אתם יכולים לתפוס את התשובה והדפיסו אותה.

עזרה: Popen.**returncode**

בעצם עליכם לכתוב 2 סקריפטים קצרים. אחד שפשוט מחזיר 0 או 1 כתגובה לפרמטר כלשהו. נניח אם הסקריפט קיבל בתור פרמטר את הערך ok הוא יחזיר 0 ואחרת יחזיר 1

הסקריפט השני יריץ את הסקריפט הראשון עם subprocess.Popen ויקבל את התשובה שתחזור מהסקריפט (חישבו איך, התשובה נמצאת כמה שורות למעלה)

תנו דוגמא לשימוש עם הפרמטר stdin והסבירו מה קורה אם מספקים אותו ואם לא.

תשובה:

# הפניית פלט מהמסך לקובץ

אם יש לכם סקריפט שמבצע הדפסות למסך (print) ועכשיו אתם מעוניינים להפנות את הפלט לקובץ, כיצד תעשו זאת? מבלי לעבור על כל פקודות ההדפסה ולשנות אותם.

במודול sys יש משתנה stdout

אם תתנו לו ערך של קובץ (handle) אז הפלט יופנה לשם

כשתהליך מתחיל לרוץ אז מערכת הפעלה נותנת למשתנה הזה את הערך של המסך

לפני שאתם דורסים ומשנים את הערך הזה, דאגו לשמור אותו בכדי שתוכלו לשחזר אותו ולשוב להדפיס למסך.

import sys

org\_stdout = sys.stdout

f = open("my\_output.txt", 'w')

sys.stdout = f

print(“something 1”)

sys.stdout = org\_stdout

f.close()

print(“something 2”)

הריצו את הקוד או משהו דומה לו. הבינו טוב טוב מה קורה שם והסבירו

אחרי שהבנתם, עשו דבר דומה עם stdin ופרטו את מה שעשיתם כולל צילומי מסך

# שלב ב: בניית shell

שימו לב: לשלב זה יש מחוון בסוף המסמך. כמו כן יש להגיש בנוסף לקוד, מסמך word עם צילומי מסך כפי שמפורט במחוון.

# מטרות התרגיל

* הבנת המושג STDIN, STDOUT ודריסה (override) שלהם
* הבנת המושג pipe והבנה שהתהליכים רצים במקביל (ולא סדרתי)
* הכרת המודול subprocess והממשק שהוא מחצין (מייצא)
* הבנת ההבדל בין *internal commands* לבין *external commans*
* הבנה ומימוש משתנה סביבה **path** והדרך שמערכת ההפעלה מוצאת את התהליך הנכון להפעיל

לשם כך עומדות בפניכם לפחות המסמכים/מאמרים הבאים:

* מאמר טוב (לדעתי)

<https://www.datacamp.com/tutorial/python-subprocess>

* התיעוד של פייתון. יש לקרוא ולהבין אותו.

<https://docs.python.org/3/library/subprocess.html>

יש להתעמק בעיקר בפקודות popen ו- run ולהבין את ההבדלים ביניהם

* מומלץ להיכנס לקוד עצמו של המודול subprocess שמגיע כחלק מההתקנה של פייתון ונמצא בספריית lib, אצלי הוא מותקן ב-

C:\Python310\Lib\subprocess.py

* מומלץ (ואפילו רצוי) להסתכל ולהשתמש בדוגמאות (סקריפטים) שהוצגו בשיעור

countLines.py

printToScreen.py

pipes.py

בתרגיל זה נממש Shell פשוט (בדומה ל- CMD) משל עצמנו ב- .Python ה- Shell שלנו יתמוך ברוב הפקודות Shell הסטנדרטיות של Windows ויאפשר לנו להוסיף בצורה קלה ובטוחה פעולות Shell משל עצמנו.

בשיעור ראינו שיש פעולות (פקודות) שהן חלק מהפקודות של ה- CMD של Windows ולהן קראנו internals command (כמו cd, dir, set, help ועוד רבות אחרות) ופעולות נוספות שהן הפעלת תהליכים חיצוניים (כמו find, nslookup, ping ועוד) וקראנו להן externals command.

הפקודות הפנימיות לא צריכות להפעיל תהליך (process) חיצוני נוסף, לעומתן הפעולות החיצוניות כן מפעילות תהליך חיצוני נפרד (בעזרת המודל subprocess שהודגם בשיעור).

בהינתן פעולה להריץ ובדומה למנגנון של windows, גם ה- shell שלנו יבצע חיפוש במספר ספריות בכדי לאתר ולמצוא את הפעולה להריץ אותה ובמידה וימצא אז יריץ אותה עם הפרמטרים שניתנו בשורת הפקודה (prompt). הפעולות שנחפש אותן יהיו סקריפטים ב- python אותם אנחנו נכתוב ונשמור בתיקיות שיהיו שמורות אצלנו כרשימה, בדומה למשתנה path של windows

במידה ולא מצא אז ייתן למערכת ההפעלה למצוא את הפקודה (יוסבר בהמשך).

להלן הסבר יותר מפורט.

ראשית, ל- shell שלנו אסור לקרוס. יש להגן מפני חריגות.

דבר נוסף, יש לתמוך בפעולות ללא קשר לאיך שהוקלדו עם אותיות גדולות וקטנות. לדוגמא כל הפקודות הבאות תקינות: dir, DIR, diR, Dir...

יש צורך לבצע לולאה כללית המבקשת קלט לאחר הצגת prompt ייחודי שתבחרו שיכלול את המיקום הנוכחי (ספרייה), שם המשתמש (השם שלכם) ומשהו נוסף (כל דבר אחר שתבחרו). הקלט הינו פקודה לביצוע ל- shell והוא יכול להיות אחד משש הקבוצות הבאות לפי סדר קדימות.

1. פעולה פנימית (internal command).   
   עליכם לממש את 4 הפעולות הבאות cd, set, help, exit ולהוסיף עוד 4 פעולות שתבחרו (מתוך אלו שה- shell של windows תומך כמו type, dir, time ועוד רבים אחרים).   
   שימו לב שהפקודות האלו יכולות לקבל פרמטרים ולהפעיל בצורות שונות. לדוגמא set יכול להיות מופעל רק ע"י הקלדה של הפקודה set (ללא פרמטרים) או בתוספת שם המשתנה (set path) או בתוספת שם המשתנה וערך לעדכון (set USERNAME=Israel), באותו האופן גם help יכול לקבל פרמטר וגם לא.
2. הפעלת סקריפט פייתון (external command). חלק מהסקריפטים יתמכו בקבלת פרמטרים בשורת הפקודה. סקריפטים אלו יהיו ממוקמים בספריות שמשתנה הסביבה שלנו בסקריפט יחפש.  
   כמו שהוזכר קודם, נתחזק משתנה בסקריפט שלנו (אפשר גלובאלי, לא חייב) שיפרט רשימה של ספריות ששם הסקריפט שלנו יחפש את הסקריפטים האלו. נניח נקרא לו MY\_PATH  
   דוגמא לשורת פקודה יכולה להיות: hexdump.py abc.exe

אין צורך לציין python.exe  
במידה ונמצאה התוכנית באחד מהספריות אז יש להפעיל אותה עם python

על מנת להראות שה--scriptים שלנו באמת נטענים ועובדים כפי שצריך, נכתוב סקריפטים לדוגמא:

* + Hello - סקריפט שידפיס למסך Hello ואת שם המשתמש שהריץ אותו. סקריפט זה ישמש בעיקר לצרכי debug
  + HexDump - סקריפט המקבל נתיב לקובץ בינארי ומדפיס אותו בצורה הקסדצימלית, כאשר כל בית בקובץ מומר למספר הקסדצימלי.
  + **בונוס:** grep - סקריפט המקבל 2 פרמטרים – נתיב לקובץ ו- Regex ומדפיס שורות אשר יש בהן התאמה ל- Regex המתקבל כפרמטר. על מנת למממש את הפעולה בצורה הטבעית ביותר יש לתמוך גם בקבלת נתיב לקובץ וגם בקבלת הקובץ עצמו דרך ה- stdin (כפי שקורה כאשר מעבירים מידע ב- pipe)

1. הפעלה של פעולות של shell אחר (CMD). דוגמה dir, copy, ren בפעולות שמופעלות עם ובלי פרמטרים (לדוגמא dir) יש להראות תמיכה ב-2 האפשרויות אבל אין צורך לתמוך בכל האופציות.  
   לדוגמא ניתן להפעיל את dir ללא פרמטרים וניתן להפעילה כך: dir \*.txt או dir /s   
   יש לבחור לפחות ב- 2 פעולות שמקבלות פרמטריםץ  
   עזרה: הסתכלו על סעיף 4 ב"עזרה למתקשים" בהמשך המסמך
2. הפעלה של תוכניות exe מספרייה נוכחית ו/או מכל מסלולי ה- path , כאן הפעולה run/Popen במודול subprocess כבר מוצאת את התוכנית ואין צורך לבצע חיפוש ב- path
3. הפעלה עם שימוש ב- redirect in (>)
4. הפעלה עם שימוש ב- redirect out (<)
5. מאוד חשוב: הפעלה עם תמיכה במסנן pipe (מסוג |) לדוגמה: HexDump a.txt | Grep 8F

(אפשר עם ריבוי מסננים רקורסיבי אבל לא חובה)

# הנחות מקלות

* ניתן להניח שאם בפקודה מופיע גם redirect in > וגם redirect out < אז הסימן < יהיה לפני >

לדוגמא: find.exe "xml" < input.txt > output.txt

עבור תמיכה ללא מגבלת הקדימויות (הסדר) ינתן בונוס

* נדרש לתמוך בלפחות pipe אחד (עבור תמיכה ב- pipe ללא הגבלה ינתן בונוס)

# עזרה נוספת למתקשים,

1. רצוי להבין לעמוק את השימוש בפונקציות המימשק של המודול

<https://www.datacamp.com/tutorial/python-subprocess>

<https://docs.python.org/3/library/subprocess.html>

שימו לב, הפעולה subprocess.run מחליפה את subprocess.call ואין צורך להשתמש בה. כלומר צריכה להיות סיבה טובה למה כן להשתמש בה ולא בממשק המתקדם יותר.

בכיתה הודגם והוסבר השימוש ב- popen וניתנו סקריפטים המדגימים את השימוש. מותר ורצוי להשתמש בסקריפטים שהודגמו.

2. כדאי לארגן את הרשימות של הפעולות שנתמכות פנימית ברשימות כמו :

cmd\_commands = ["dir","type","md","copy","ren"] # can add here all "dos" commands

inner\_commands = ["exit","help","cd","set" [

3. הצעה לביצוע ls

items = listdir(mypath)

for item in items:

full\_name = join(mypath,item)

if isfile(full\_name):

print "%s\t\t%s \t%s" %(time.ctime(path.getmtime(full\_name)), str(path.getsize(full\_name)) ,item)

else:

print "%s\t<Dir>\t\t%s" %(time.ctime(path.getmtime(full\_name)), item)

4. בסעיף 3 כדאי להשתמש

הפעלה של aa עי פנייה ל CMD

args = "cmd.exe /c " + " aa"

res = subprocess.run(args, stdin=None, stdout=None, stderr=None, shell=True)

5. בסעיף 6 (מימוש pipes) נדרש להשתמש ב- subprocess.popen

תרגיל בניית Shell – מחוון

תמיכה ב- stdin וגם ב- stdout

* צריך לתמוך בכך שכל אחד מהם עובד בנפרד וגם ביחד

תמיכה ב- pipe (חובה שימוש ב- Popen)

חיפוש ב- path פנימי של המערכת (מתקשר לנושא Scripts command)

פקודות הן case insensitive

* ניתן להקליד פקודות עם אותיות גדולות וקטנות ומעורבב והפקודה צריכה להתקבל

Internal commands כולל תמיכה בדגלים (2 לפחות). יש להראות לפחות 4 פקודות כאלו

Scripts command

* אלו סקריפטים בפייתון שימוקמו בספריות שונות.
* יש להפעיל אותן רק בשמן (כולל הסיומת py) וללא python.exe

Externsl commands

* כל תוכנית ש- windows יודע להריץ (calc, ping, notepad, nslookup)

**יש להריץ את הפקודות הבאות ולהגיש צילומי מסך של ההרצה:**

1. להדגים את כל הפעולות הפנימיות (internal command) כולל הדגמה של 2 מקרים לפחות ריצה עם פרמטרים ודגלים. והרצה עם redirect in

לדוגמא:

dir \*.txt

dir /s

dir /s \*.exe

dir \*.txt > c:\temp\dir\_result.txt

1. להדגים הרצה של סקריפטים פייתון שימוקמו בתתי ספריות שנמצאות במשתנה סביבה path הפנימי של my\_shell

לדוגמא:

printToScreen.py | countLines.py

printToScreen.py | countLines.py > res.txt

1. להדגים הרצה של פקודות shell שלא נתמכות אצלכם כפעולות פנימיות. נניח שלא בחרתם לממש את הפקודה copy

copy a.txt b.txt

1. להדגים הרצה של פקודות חיצוניות

להדגים את הפקודות הבאות:

ping [www.google.com](http://www.google.com)

ping [www.google.com](http://www.google.com) > ping\_result.txt

ping [www.google.com](http://www.google.com) | countLines.py > res.txt

tasklist | find "Teams"

find “Python” < c:\temp\a.txt

calc

notepad

notepad c:\temp\a.txt (הקובץ קיים)

ולהוסיף עוד 3 דוגמאות שלכם