מבוא לתכנות מערכות – 02340124

תרגיל בית 5

סמסטר חורף 2024-2025

תאריך פרסום: 28.1.25
תאריך אחרון להגשה: 4.2.2025 עד השעה 23:55
מתרגל אחראי לתרגיל: דניאל אלגריסי

1 הערות כלליות

- תרגיל זה מהווה 4% מהציון הסופי בקורס.
- התרגיל להגשה בזוגות בלבד.
- כל ההודעות והעדכונים הנוגעים לתרגיל זה יפורסמו באתר הקורס ב-Grades
- מענה לשאלות בנוגע לתרגיל יינתן אך ורק בפורום התרגיל בפיאצה (קישור באתר הקורס) או בסדנאות. אין לשלוח דוא"ל לגבי התרגיל.
- שימו לב – לא תינתנה דחיות למועד הגשת התרגיל.
- העתקות קוד בין סטודנטים תטופלנה בחומרה.
- בתרגיל זה, אין צורך לעמוד בקונבנציות ספציפיות. יחד עם זאת, יש לשמור על פורמט אחיד עבור מתן שמות (Naming) לכל אורך הפתרון שלכם.
- קבצי התרגיל מסופקים לכם ב-GitHub.
- החומר הנדרש כדי לפתור את התרגיל הינו כלל החומר הנלמד בשפת python, כלומר תרגולים 9 עד 11 כולל.
- תיקונים למסמך התרגיל, יסומנו בצהוב.

2 הקדמה

מטרת תרגיל זה הינה היכרות עם שפת python. לתרגיל זה שני חלקים, יבש ורטוב, בהם תתנסו בשימוש בתכונות שונות בשפה אשר נלמדו בכתה.

3 חלק יבש

בחלק זה, ינתנו לכם מספר קטעי קוד קצרים ב python.
בכל אחד מהסעיפים עליכם להסביר במילים מה קטע הקוד עושה. בנוסף, עליכם לכתוב oneliner שקול לקטע הנתון, כלומר שורת python בודדת המבצעת בדיוק את אותה פעולה (כולל הדפסה תקינה במידה וקיימת).
את הפתרון יש לכתוב בקובץ PDF בשם dry.pdf.

שימו לב – הקוד אותו אתם כותבים בחלק זה לא יורץ או יבדק אוטומטית, אלא יבדק ידנית בלבד.
יחד עם זאת, מומלץ להיעזר במפרש של python על מנת להגיע לפתרון.

1. לצורך קטע הקוד הבא, מוגדר עבורכם משתנה בשם my_str המכילה מחרוזת, **למשל**:

```
my_str = "The first one is the easiest!"
```

הסבירו מה הקוד הבא עושה וכתבו oneliner שקול:

```
res = []
my_str = my_str.split()
first, second, third = 1, len(my_str), 2
while first < second:
    res .append(my_str[first])
    first += third
```

2. לצורך קטע הקוד הבא, מוגדר עבורכם המשתנה n (למשל n=42) , הסבירו מה הקוד הבא עושה וכתבו oneliner שקול:

```
my_dictionary = {}
for x in range(100, 0, -3):
    if x % n == 0:
       my_dictionary[x] = f"{x} is divided by {n}.\n"
    else:
       my_dictionary[x] = f"the remainder of {x} divided by {n} is: {x % n}.\n"
print(*my_dictionary.values())
```

שימו לב – אין משמעות לסדר בו מודפסים האיברים , הניחו שבסוף קטע הקוד מסתיימת ריצת התכנית (ולכן אין צורך שהמשתנה my_dictionary יכיל מידע), השתמשו בשיטת ההדפסה בקטע הקוד המצורף וכן שימו לב שניתן לפצל את השורה עם היא ארוכה לצורך קריאות) .

3. קראו על הפונקציות **ord, chr** ב python. הסבירו מה הקוד הבא עושה וכתבו oneliner שקול:

```
for i in range(0, max(ord('9'), ord('z'), ord('Z'))+1):
    if chr(i).isalpha() or chr(i).isdigit():
        print(f"The ASCII number {i} represent the char '{chr(i)}'")
```

4. לצורך קטע הקוד הבא, מוגדר עבורכם משתנה בשם list_c המכיל רשימה של מספרים שלמים, למשל:

```
list_c = [80, 121, 116, 104, 111, 110, 32, 105, 115, 32, 102, 117, 110, 33]
```

הסבירו מה הקוד הבא עושה וכתבו oneliner שקול:

```
tmp_chr = ""
for num in list_c:
    tmp_chr += chr(num)
print(tmp_chr)
```

רמז – קראו על המתודה **join** לערכים מסוג string, תוכלו להשתמש בה לצורך הפתרון.

1 חלק רטוב

בחלק זה נממש מודול ב-python המדמה פעולה של מכונת האניגמה. אניגמה (מיוונית: αἴνιγμα - תעלומה, חידה) היא משפחה של מכונות להצפנה ולפענוח של מסרים טקסטואליים, ששימשו את הכוחות הגרמנים והאיטלקים במלחמת העולם השנייה (לקריאה נוספת לחצו כאן).

פיצוח האניגמה היה מאמץ מתמשך שבו עסקו אנשי מודיעין ומדענים פולנים, בריטים ואמריקאים החל משנות השלושים ועד סוף מלחמת העולם השנייה כדי לפצח מסרים שהוצפנו בדגמים שונים של המכונה, אחד המדענים הבולטים בצוות מחקר הפיצוח היה אלן טיורינג, מתמטיקאי בריטי, ממניחי היסודות למדעי המחשב.

את הפתרון לחלק זה יש לכתוב בקובץ enigma.py.

4.1 צופן האניגמה

כעת נסביר איך האניגמה (בתרגיל שלנו לפחות) מצפינה הודעות.

למכונת האניגמה מוגדרת קונפיגורציה לפיה מוצפנות ההודעות. הקונפיגורציה מורכבת משלושה חלקים:

1. hash_map – מיפוי בין אותיות (קטנות) באנגלית למספרים בטווח 0 עד 25 כולל. המיפוי הינו הפיך, כלומר לכל אות קיים מספר יחיד מתאים, ולכל מספר בטווח מתאימה אות יחידה.
2. wheels (גלגלים) – שלושה מספרים שלמים המסומנים W_1, W_2, W_3 כאשר W_1 בטווח 1 עד 8 (כולל), ו-W_3 הוא אחד הערכים 0, 5 או 10 ועל W_2 אין הגבלה.
3. reflector_map – מיפוי בין אותיות קטנות באנגלית לאותיות קטנות אחרות. המיפוי הינו סימטרי, כלומר אם אות c_1 ממופה לאות c_2, אזי גם c_2 ממופה ל-c_1. בנוסף, אף אות לא ממופה לעצמה.

האניגמה מצפינה כל בהודעה בנפרד מההתחלה ועד סוף ההודעה, לאחר הצפנת כל תו מתעדכנים ערכי הגלגלים איתם מוצפן התו הבא.

4.1.1 הצפנת תו בודד

הצפנה של תו בודד, נסמנו ב-c, פועלת לפי האלגוריתם הבא:

1. נאתחל i עם הערך המספרי הממופה ל-c לפי ה-hash_map.
2. אם הערך $\left(\left((2 * W_1) - W_2 + W_3\right) mod\ 26\right)$ אינו שווה ל-0, נוסיף אותו ל-i. אם ערך זה שווה 0, נוסיף ל-i את הערך 1.
3. נציב ב-i את שארית החלוקה שלו ב-26 (כלומר $i\ mod\ 26$)
4. נאתחל c_1 עם הערך אליו ממופה הערך i שקיבלנו לפי ה-hash_map.
5. נאתחל c_2 עם האות אליה ממופה c_1 לפי ה-reflector_map.
6. נציב ב-i את המספר הממופה ל-c_2 לפי ה-hash_map.
7. אם הערך $\left(\left((2 * W_1) - W_2 + W_3\right) mod\ 26\right)$ אינו שווה ל-0, נחסיר אותו מ-i. אם ערך זה שווה 0, נחסיר מ-i את הערך 1.
8. נציב ב-i את שארית החלוקה שלו ב-26 (כלומר $i\ mod\ 26$).
9. נאתחל c_3 עם האות אליה ממופה הערך i שקיבלנו לפי ה-hash_map.

c_3 שהתקבל הינו התו אליו נצפין את התו c.

שימו לב – ההצפנה מתייחסת לאותיות קטנות בלבד. כל תו שאינו אות קטנה יעבור **הצפנה באופן ריק**, כלומר, בכתב המוצפן נשאיר אותו ללא שינוי.

4.1.2 קידום הגלגלים

לאחר הצפנת התו (**גם אם הוצפן באופן ריק**), יקודמו גלגלי האניגמה באופן הבא:

- W_1 – יקודם ב-1. אם הערך גדול מ-8, הגלגל יקבל את הערך 1.
- W_2 – אם מספר התווים שהוצפנו (לא באופן ריק) עד כה בהודעה הינו זוגי, ערך הגלגל יוכפל ב-2, אחרת, נוריד 1 מערך הגלגל.
- W_3 – אם מספר התווים שהוצפנו (לא באופן ריק) עד כה בהודעה מתחלק ב-10, הגלגל יעודכן לערך 10. אם מספר התווים שהוצפנו (לא באופן ריק) עד כה בהודעה מתחלק ב-3 (ולא ב-10), הגלגל יעודכן לערך 5. אחרת, הגלגל יעודכן לערך 0.

4.2 המחלקה Enigma

ממשו את המחלקה Enigma והגדירו בה שתי מתודות:

1. בנאי – מקבל שלושה ארגומנטים, לפי סדר:
   - hash_map – מיפוי הניתן בתור מילון בו המפתחות הם תווים והערכים הינם מספרים בין 0 ל-25 (כולל).
   - wheels – רשימה בה שלושה איברים המהווים את המצב ההתחלתי של הגלגלים.
   - reflector_map – מיפוי הניתן בתור מילון בו המפתחות והערכים הינם תווים.

   ניתן להניח שה-hash_map, סט הגלגלים וה-reflector_map המתקבלים תקינים.

2. encrypt – מקבלת ארגומנט בודד בשם message מסוג מחרוזת. המתודה מחזירה את ההודעה מוצפנת, לפי האלגוריתם שתואר בחלק 4.1.

   שימו לב – בסוף ריצת encrypt, האניגמה חוזרת למצבה ההתחלתי.

4.3 פונקציות נוספות

ממשו את הפונקציה הבאה, מחוץ למחלקה Enigma:

1. **load_enigma_from_path** – הפונקציה מקבלת ארגומנט יחיד המהווה path לקובץ בפורמט JSON. הקובץ מכיל מילון ובו שלושה מפתחות – hash_map, wheels ו-reflector_map.
   הפונקציה מחזירה אובייקט אניגמה המאותחל עם הערכים מהקובץ שהתקבל.

   במידה ויש שגיאה בקריאת הקובץ, יש לזרוק חריגה מסוג **JSONFileException** (עליכם להגדיר חריגה כזו בעצמכם).

4.4 הרצה כתסריט (script)

לעיתים נרצה להיות מסוגלים להשתמש ב-Enigma מה-Shell, כלומר להריץ את המודול שלנו כתסריט (Script).

התסריט יקבל קונפיגורציה של האניגמה וקובץ המכיל הודעות, יצפין את ההודעות באמצעות אניגמה עם הקונפיוגרציה הנתונה ויפלוט את ההודעות המוצפנות לקובץ או למסך.

כאשר התסריט יורץ משורת הפקודה (בלבד), הוא יוכל לקבל מספר דגלים, ולאחר כל דגל פרמטר מסוג מחרוזת. פורמט פקודת ההרצה:

```
python3 enigma.py -c <config_file> -i <input_file> -o <output_file>
```

הפרמטרים של התסריט:

- config_file (חובה) – קובץ בפורמט json ובו קונפיגורציה של מכונת האניגמה.
- input_file (חובה) – קובץ קלט המכיל הודעות שיש להצפין. כל הודעה רשומה בשורה נפרדת.
- output_file (אופציונלי) – קובץ אליו יש לכתוב את ההודעות המוצפנות, כל הודעה בשורה חדשה. במידה ולא התקבל הדגל o-, יש להדפיס את ההודעות המוצפנות למסך (פלט סטנדרטי).

במידה והתקבלו לתסריט פרמטרים לא מתאימים (למשל שימוש בדגל לא קיים) ,יש להדפיס לערוץ השגיאות את ההודעה הבאה ולצאת מהתכנית:

Usage: python3 enigma.py -c <config_file> -i <input_file> -o <output_file>

על מנת לצאת מהתכנית יש לקרוא לפונקציה exit(1)

שימו לב – הדגלים יכולים להתקבל בכל סדר שהוא .ניתן להניח שלאחר הדגל קיים פרמטר (יחיד).

בכל מקרה של שגיאה במהלך ריצת התסריט (למשל אם אין הרשאות לפתוח את אחד הקבצים) ,יש להדפיס את ההודעה הבאה ולצאת מהתכנית:

The enigma script has encountered an error

4.5 הוראות מימוש

- לאורך התרגיל, רצוי להימנע בשימוש ב"מספרי קסם", פרט ל-0 או 1. ב python אין קבועים, תוכלו במקום להגדיר משתנים גלובליים או מקומיים ששמם באותיות גדולות בלבד.
- מותר לכם להוסיף מתודות ופונקציות נוספות לקובץ התרגיל.
- וודאו כי אתם עובדים עם גרסה עדכנית של python (3.9 ומעלה).

4.6 בדיקות מסופקות

בתיקיה tests תוכלו למצוא מספר קבצי קלט לדוגמה ופלטים מתאימים לכל קלט. ניתן להריץ את התכנית עם הקלטים המסופקים, כפי שנדרשתם בתרגיל, כלומר ע"י השורה:

```
python3 enigma.py -c config_file.json -i tests/test{i}.in -o tests/test{i}.out
```

תוכלו לבדוק את פלט התכנית אל מול הפלט המצופה למשל באמצעות הפקודה הבאה:

```
diff --strip-trailing-cr -B -Z test{i}.out tests/test{i}.expected
```

2 הגשה ובדיקה

5.1 הגשה

עליכם להגיש את הקבצים הבאים:

- dry.pdf
- enigma.py

את ההגשה יש לבצע במערכת ה-Gradescope.

שימו לב – על אחד השותפים (בלבד) להגיש את פתרון התרגיל במערכת. לאחר ההגשה יש להוסיף את השותף השני להגשה באמצעות לחיצה על כפתור "Group Members" ב-Gradescope.

הערות:

- ניתן להגיש מספר פעמים את התרגיל, ההגשה האחרונה (בלבד) היא זו שתיבדק ותקבל ציון.
- בכל הגשה נוספת שאתם מבצעים, יש להוסיף מחדש את השותף להגשה ב-Gradescope.
- וודאו שהקבצים אותם אתם מגישים (בפרט קבצי PDF) ניתנים לפתיחה ולצפיה.

5.1.1 הגשה באמצעות GitHub

בתרגיל זה, יש להגיש את הפתרון שלכם במערכת ה-Gradescope באמצעות GitHub. עליכם להגיש את ה-Repo שיצרתם עבור תרגיל זה (כפי שיצרתם בתרגילים הקודמים) אליו הוספתם את הפתרון לחלק היבש ולחלק הרטוב. לשם כך בדף הגשת התרגיל בחרו באפשרות ההגשה באמצעות GitHub, תנו ל-Gradescope הרשאות גישה לחשבון ה-GitHub שלכם ובחרו את ה-Repo אותו תרצו להגיש.

הקפידו על עבודה נכונה עם Git כפי שלמדנו בהרצאות ובתרגולים.

5.2 בדיקה אוטומטית

מיד עם ההגשה שלכם במערכת ה-Gradescope, יתבצעו מספר בדיקות אוטומטיות על קובץ ההגשה:

1. בדיקות שפיות – בודקות שכל הקבצים הדרושים נמצאים ב-zip.
   בדיקות אלו נועדו לוודא שפורמט ההגשה שלכם תקין.

2. טסטים אוטומטיים – הבדיקה האוטומטית מריצה את התכניות שהגשתם עם הקלטים שסופקו לכם בקבצי התרגיל. הטסטים האוטומטיים בודקים שריצת התכנית הסתיימה בהצלחה ונתנה פלט נכון.

בדיקות אלו נועדו עבורכם לוודא שההגשה שלכם תקינה והקוד עובד כראוי, קראו היטב את הפלט והמשוב מהבדיקות!

על מנת להשוות תכנים של קבצים בבדיקה האוטומטית, נשתמש בפקודת ה-diff הבאה (כאשר file1 ו-file2 הם הקבצים אותם נרצה להשוות:

```
diff --strip-trailing-cr -B -Z file1 file2
```

<u>שימו לב</u> – בנוסף לבדיקות שסופקו לכם, ההגשה הולכת לעבור בדיקות אוטומטיות נוספות לפיהן יקבע הציון על התרגיל. אנו מצפים שתבדקו בעצמכם את התרגיל שלכם מעבר לבדיקות המסופקות, האחריות על נכונות ההגשה היא שלכם בלבד.

5.3 תחרות מימז

בתרגיל זה מתקיימת תחרות מימז (Memes).
כדי להשתתף בתחרות, עליכם להוסיף מימ לתחילת הקובץ dry.pdf. הזוג שיזכה בתחרות יקבל בונוס של 5 נקודות לציון התרגיל (לכל אחד מהשותפים).

חוקי התחרות:
1. מותר לכם להגיש לתחרות מימ אחד <u>בלבד</u>.
2. נושא המימ חייב להיות קשור לחומר הקורס.
3. המימ הזוכה יפורסם בתרגול לאחר לפרסום הציונים.

בהצלחה !
بالنجاح !
GOOD LUCK!