

# פיתוח תוכנה מקצועית

חומר - 2025

מס' גרסה: 2

התרגיל מנוט בבלזון זכר, אך מכון לכל המגדירים והתחושים בצורה שווה

מרצה: [אביעד כהן](mailto:aviadco@mta.ac.il)  
בודק: [איתני כהן](mailto:itaych@mta.ac.il)

## תוכן העניינים

4.....	דרישות הקורס .....
4.....	כללי.....
5.....	אייר להגיש תרגילים באיחור, ולהישאר בחיים .....
6.....	הנחיות כלליות לכתיבת התרגיל.....
8.....	<b>Building the Enigma Machine</b>
8.....	מטרת התרגיל(ים) בקורס .....
8.....	מכונת האניגמה .....
11.....	מגבילות ומאפיינים של מכונת האניגמה בתרגיל זה: .....
12.....	הנחיות ספציפיות למימוש תרגיל "בנייה מכונת האניגמה".....
13.....	<b>תרגיל 1 – מימוש מכונת האניגמה כAPPLICATION Console (30%) – הasha: 14.12.25</b>
13.....	פרטים יבשים.....
13.....	דרישות.....
17.....	חלוקת למודולים.....
17.....	אייר מתחילה ? (המלצת...) .....
18.....	בונוסים.....
18.....	סבבה, סימתי. מה ואייר להגיש ? .....
19.....	<b>תרגיל 2 – מימוש מכונת האניגמה בתצורת Maven (30%) – הasha: 13.1.25</b>
19.....	פרטים יבשים.....
19.....	דרישות.....
21.....	חלוקת למודולים.....
21.....	אייר מתחילה ? .....
21.....	סבבה, סימתי. מה ואייר להגיש ? .....
22.....	<b>תרגיל 3 – מימוש מכונת האניגמה כAPPLICATION spring boot (35%) – הasha: 16.2.26</b>
22.....	פרטים יבשים.....
22.....	דרישות.....
23.....	חלוקת למודולים.....
23.....	אייר מתחילה ? .....
23.....	סבבה, סימתי. מה ואייר להגיש ? .....
24.....	נספח א' – פרטים לגבי מימוש מכונה .....
25.....	נספח ב' – פירוט תנועת המכונה (כולל לוח תקעים) .....
28.....	נספח ג' – מכונת אניגמה לגזירה ותרגול על נייר .....
29.....	נספח ד' – תיאור בקשות של Postman Collection (תרגיל 3) .....
29.....	Load Machine from XML .....
29.....	Get Current Machine Status .....
30.....	Manual Code Selection .....
30.....	Automatic Code Setup .....
31.....	Reset to Original Code .....
31.....	Process Input .....

31 .....	Get Machine History
32 .....	נספח ה' – תיאור מבנה המערכת באמצעות קובץ XML
33 .....	סכמה תרגיל 2
34 .....	נספח ו' – תרשימים סכמטיות XML
34 .....	סכמה תרגיל 1
35 .....	סכמה תרגיל 2

כללי

1. בהקורס אין בחינה אך חובה להגישתרגילים (סה"כ 4).
2. המלצתני היא להגיש את התרגילים ביחידים. אולם אם אין ברירה - את מרבית התרגילים (3) ניתן להגיש בזוגות, אך לא בשלישיות / רביעיות / חמישיות או יותר (כן, גם אם מדובר בשלישיה / רביעיה / חמישיה הצועדת היחידו לאורך שנים מאז גיל הגן והגישה עד עכשו את כל הפרוייקטים ביחד).
3. בעבודה משותפת על תרגיל יש להקפיד על מעורבות אקטיבית של כל המתGISים בכל חלקו התרגיל.
4. במידה והוגדר בונוס לתרגום מסוים, ציון הבונוס יתווסף לציוון התרגום בלבד (ולא לציוון הסופי של הקורס כולו).
5. התרגילים יגשו דרך מערכת Mama. מוגדר רכיב 'מטלה' נפרד לכל תרגיל.
6. לפני שליחת התרגום יש לבדוק שהוא עובד ומכל את הקבצים המעודכנים ביותר, על מערכת "נקיה".  
בצעו את סט הפעולות שאותם מצפים מן הבודק לבצע וודאו כי הכל מתנהל כראוי וכסדרו.
7. ניתן להחליף את השותפה/ה בכל תרגיל, ללא צורך באישור או הודעה למרצה.  
הניקוד על כל תרגיל נזקף לזכות הסטודנטים שביצעו אותו בלבד.
8. יש להעלות את התרגום רק עבור אחד מבני הזוג ולהוסיף את שם בת/בן הזוג ומספר תעודה זההות שלה/ו בקובץ ה *readme* המצורף (פרטיים בהמשך).

1. ניתן להגיש תרגיל עד חמישה ימים איחור, כאשר עברו כל 24 שעות איחור – תורד נקודה אחת מצוין התרגיל; תרגיל שיוגש באיחור של יותר מחמשה ימים (ללא סיבה מוצדקת) – **פשט לא בדק**.  
בכל מקרה עיקבו אחר ההוראות המדויקות שינטנו בלוח ההודעות של הקורס. בכל מקרה של כפילות – han הקובלות !

2. במידה והגשתם תרגילอลם הבודק נתקל בבדיקהתו במצב שפיט לא אפשר את המשך הבדיקה (למשל CISLON בטיענת קובץ הבדיקה) – הרי שאתם מוגדרים כתקלת 0 level. במקרה של תקלת שכזו הבודק יידע אתכם ויאפשר לכם לבדוק, לתקן ולהגיש מחדש את התרגיל כדי שאפשר יהיה לבדוקו אחרי הכל.  
שים לב כי **בכל בהגשה חוזרת** בגל 0 level – הציון לתרגיל יתחל מ 90, ללא שום קשר לאופי הבעה /או התיקון (גם אם התיקון היה "קטן". גם אם התיקון היה בגל הבלבול בהגשה של גרסה קודמת יותר של הקבצים. גם אם הכלב אכל לכם את שיעורי הבית)

3. בתרגילים השונים ניתן למשבוןם (פרטים בהמשך).  
המטרה של הבונוס היא לעזור לכם להעלות את הציון ולא להורידו !  
רוצה לומר: אל תגיבו באיחור רק בשביל להספיק לפתח בונוס.  
בונוס מפתחים **אם ורק אם** סיימתם את כל דרישות הבסיס להגשה, יש לכם עוד מספר ימים, וברצונכם לנשות ולהגדיל את הציון ע"י בונוס.  
**היות וכך, ולמען הסר כל ספק: לא בדקם** הבונוסים עבור תרגילים שהוגשו באיחור (**שאינו מוצדק**).  
בהתאם לכך, אני שומר לעצמי חירות רבה יותר בשינוי צזה או אחר של מי מסעיפי הבונוס, גם במהלך התרגיל עצמו.

4. עומס לימודיים, בעבודה, בחיים, בגל הילדים או ההורים, שכנים וחברים (או בכלל תחום אחר) **אין נחשב** כסיבה לגיטימית לבקשת הארכה.

5. במקרה של בקשה להארכה (מכל סיבה שהיא, לרבותAMILAOIM ומחלת) יש לפנות למרצה מראש על מנת לקבל אישור. הפניה תבוצע במיל.

6. לאנשי הקבע – הישאות של שבת בסיס אינה נחשבת כסיבה לגיטימית להארכה (מכיוון שזה חלק מהסדר העבודה בצה"ל);  
יציאה לאבט"ש כן נחשבת כAMILAOIM ויש להגיש אישור ממפקד הבסיס.

7. ניתן לערער על ציהון של תרגיל על פי הוראות הבודק בעט פרסום הציונים במע' הממא.   
כדי לערער יש לשוחח מייל לבודק בצריף כל הסיבות והטענות שלכם.

- במהלך הקורס יוצגו דוגמאות והסבירים מbossים על כתיבה בסביבת הפיתוח (IDE) – IntelliJ IDEA. אתם מוזמנים (ומעודדים בזאת) לפתח גם כן את התרגיל בסביבת העבודה העובדה IntelliJ. ניתן לקבל רישיון חינם לשימוש בגרסה ultimate, רק בשל היוכם סטודנטים מכללה (כבר שווה!) יחד עם זאת, כל אחד רשאי לבצע לעבוד בסביבת העבודה הנוחה והモכרת לו. כך או אחרת הגשת התרגיל אינה כוללת את סביבת הפיתוח אלא אף ורק הריצה ידנית מ cmd (כמו פום...). שימוש לב: מבחרינטכם, לבדוק פשוט אין IntelliJ (או כל ide אחר) ולכן זו אפלו לא אופציה. חיסכו ממוני (ומכם) את כתיבת המיל המבוקש זאת.
  
- יש להגיש את התרגילים בתור קובץ zip/rar (לא Z!) הקובץ יכול:
  1. כל הקבצים הרלבנטיים להפעלת התרגיל (jar/jar – פרטיים בגוף התרגיל).
  2. קובץ אצווה ( == batch) שיכיל את הפקודה שמריצה את התרגיל.
  3. קובץ readme שיכיל את פרטי המגיש/ם, כמו גם הנחיות כליליות להריצה התרגיל וכל הנחות שלקחו חלק במהלך התרגיל ואינם סבוריים שחשיבותם כיר. דמיינו כי בכל שאלה/תקלה שיתקל בהן הבודק, ימודד לרשותו רק קובץ ה readme שלהם. דאגו להבהיר ולהסביר את כל הדברים שיכולים להשتبש ו/או שבעתיים יתכו בעיות/שאלות/תתיות וכיוצא'.
 כמו כן, כל הנחה שאתה מניחים בעצמכם לגבי אופןนำไปים התרגיל (בין אם בלוגיקת התרגיל ובין אם בהנחה טכנולוגית) צריכה להיות רשומה בקובץ.
  
- על קובץ ה readme להיות בפורמט word או pdf (**לא notepad!**). (חי נפשי – אם מישחו מגיש readme כקובץ טקסט פשוט – ירד לו ניקוד מהתרגיל...)
 קובץ ה readme צריך לפתוח בעמוד השער המדויק כפי שניתן לכם במסגרת חומרה התרגיל. אין לחרוג משער זה.
  
- דוקא בגלל שאין זהו קורס שבו יכנסו לנכסי הקוד ויבדקו כל שורה ושורה, יש להקפיד ביתר שאת על קוד נקי, מסודר, קרייעיל. בפרט:
  - הימנעו משכפל קוד
  - פונקציות ארוכות מדי (בדרכ'ך יותר מוגדל עמוד)
  - בחירת שמות גראפים למחלקות, לפונקציות ולמשתנים
  - הזהה (אינדנטציה) נכונה
  - imports מיותרים
  - יש להקפיד להשתמש ב-modifiers ב-classes בצורה נבונה:
  - מחלקה שלא אמרום לבנות אובייקטים שלה אמרה להיות מוגדרת כ-abstract
  - קבועים יש לסמן כ-final
  - משתנים של המחלקה רצוי להגיד כ-private אלא אם יש סיבה לגיטימית לבחירה אחרת.
 יש להקפיד על מוסכמות בסגנון הכתיבה – שמות מחלקות יתחלו באות גדולה, שמות חבילות, משתנים ומונקציות באות קטנה, שמות קבועים יהיו מורכבים רק מאותיות גדולות וכו'. ראו מסמך **java coding conventions** שהועלה למאמה.
  
- התמודדות עם קלט שאינו תקין (במקומות הרלבנטיים) היא חלק בלתי נפרד מחווית המפתח (լטוב ולרע...).
 יש לוודא קלט תקין מהמשתמש בכל שלב ולהציגו הודאות שגיאה קרייאות, אינפורטטיביות במידה והקלט אינו תקין. (למשל: לא להגיד **הקובץ לא תקין** – אלא **מה לא תקין בקובץ** בצורה מפורטת...)

- 

כל הקלט והפלט בתרגילים השונים יהיה באנגלית בלבד.

אין להציג או לתרmor בקלט קלט /או הצגת פלט בעברית או בכל שפה אחרת.

כל הקלטים באנגלית יהיו case insensitive, כלומר אין חשיבות ל caseInsensitive. דוגמא: OmOmOmOm=MomOmOm

- 

הוראות שגיאות שייגרמו לאפליקציה שלא לרווח יורידו נקודות, וכך רצוי מאוד שתנסו להתקין את האפליקציה בעצמכם לפי ההוראות שתכתבו.

- 

זהו תרגיל מתגלל. המטרה היא לבנות בסיס ראשון בתרגיל הראשון, ולהמשיך ולהשתמש בו, ככל האפשר (ואפשר!). במהלך התרגילים הבאים. השקיעו חשיבה ותכנון בעיצוב הפתרון תוך מחשבה על איך מה שתעשו היום ישרת אתכם מחר. (זה כלל נכון לחיים, לא רק לתרגיל זה).

- 

חלק מהעבודה בתרגילים היא קבלת החלטות בנושאים שאינם מפורטים במדויק. מטרת התרגיל היא לתרגם את הנושאים המרכזיים הנלמדים בקורס. על כן, בכל מקום לא מופיעה דרישת מדיוקת – מוטל عليיכם לבחור בדרך הגיונית ביותר שנראית לכם ולציין את בחירתכם בקובץ **Readme** אשר מוגש עם התרגיל. אם יש ספק לגבי אופן פעולתכם אתמים מעודדים לשאול האם הפתרון שלכם חوابים לתת לסוגיה מסוימת הוא קביל ולגייטימי (שאלות בפורום, מייל למרצה וכו')

- 

התרגיל מתקיים כולם במסגרת ג'אווה גרסה 21. הקפידו להוריד, לעבוד, לkiemפל ולהריץ עם הגרסה המתאימה בלבד.

- 

ו/ידוא הגשת התרגיל טרם הגשתו:

- 

יש לוודא כי ההגשה שלכם רצה היטיב על מע' נקייה, באופן שבו גם הבודק יירץ אותה, על מערכת נקייה ולא תוצר!

- 

לוואי אחרים של הפעולות קודמות שלהם. הבודק יבצע את הבדיקה על מע' windows 10. כל מי שפתח על גבי mac/axunia – זכותכם – אבל גם חובתכם לוודא כי אתם רצים היטיב על windows. למען הסר ספק, לא תבצע בדיקה על מע' הפעלה אחרת. כמו כן לא תהיה התחשבות בתקלות שמקורם רק בשל פעולה על מע' הפעלה שונה (ולא שאמורות להיות תקלות כאלה..)

- 

יש לוודא כי כל קבצי הבדיקה השונים שהועלו ל mama נטען בהצלחה ע"י המע' שלכם טרם הגשתה. בדיקת הבודק תתחיל לבדוק בסיסית המבוססת בקרה גסה על קבצים אלה. חבל ליפור Level 0 על שיטות שיכלתם לעלות אליה בשנייה עוד בשלב הפיתוח.

- 

כאמור, הגשה חוזרת בשל תקלות 0 level מראש מצינו של 90. בלי שום יוצא מן הכלל. הקדימו תרופה למכה.

- 

בחלק מהתרגילים ניתנת אפשרות לימוש דרישות בונוס.

ישנם 2 סוגי בונוסים:

- 

1. בונוס בגין טווח התרגיל - יכול להביא אתכם לכל היוטר לציון 100, ולהפוך במרקחה והורדו לכם נקודות בשל תקלות.  
2. בונוס מחוץ לטווח התרגיל – יכול להעלות את ציונכם אף מעבר ל 100 (וכן, יש כפלי מבחנים לטובות הליקוח).  
בכל מקרה יש לבצע את הבונוס **אם ורק אם** סימתם את כל דרישות הבסיס הכרחיות לתרגיל.

- 

חלק מהבונוסים בתרגילים השונים הם ככל שנוועדו "להקדים תרופה למכה" – שימוש דרישת בתרגיל ח אשר בכל מקרה

תגיא כדרישה חובה בתרגיל 1+ח.

- 

הדבר נדרש לפחות ארכם להוריד את העומס הכספי בתרגיל 1+ח, מתוך הנחת יסוד שתרגיל ח הוא קל יותר ומרוחק יותר. תכננו את בעודתכם בהתאם ו שאפו "להקדים תרופה למכה", במידת האפשר. (וגם זה כלל חשוב לחיים, בלי קשר לתרגיל ולקורס).

- 

פירוט הבונוסים, משקלם ונקודותיהם מפורט בגוף התרגיל הספציפי.

- 

אם כבר ממשים בונוס, יש למש את כלו, **על** דרישתו כדי לזכות במלוא הניקוד שהוא מקנה. בכל מקרה ההחלטה על ניקוד הבונוס היא בידי הבודק/מרצה בלבד (המגמה היא להיות נדיבים ככל האפשר...)

- 

כאמור, למען הסר כל ספק – ניקוד הבונוס מתווסף לניקוד התרגיל הספציפי שבו הוא מומש ולא לניקוד הסופי של הקורס. לא ניתן לקבל ציון סופי בקורס שהוא מעל ל 100 (גם אם בזכותו הבונוסים הגעתם לציון צה).

## מטרת התרגיל(ים) בקורס

חלק א' – מימוש מכונת האניגמה בקוד. הפעלה ותפועלה באמצעות ממשק `soleso.console`. ההוראות והתמונות עם אבני הבניין הבסיסיות של הטכנולוגיה.

חלק ב' – הכלכלה והרחבת יכולות המכונה, והמרת המימוש של חלק א' למבנה עבודה מתאים עם `maven`.

חלק ג' – הטענת וחשיפת אניגמה במסגרת שירות `Tor` עבודה עם `postman client` נגנון.

### מכונת האניגמה

במהלך מלחתת העולם השנייה, השתמשו הכוחות הגרמניים [במכונת האניגמה](#) כאמצעי להצפנה ולהעברת תשלומות בין הצלולות למטה הקרב לティיאום התקפות על כוחות בעלות הברית.

מכונת האניגמה הינה מכונה אלקטרו-מכנית המאפשרת להצפין הודעות טקסט. באמצעות המכונה ניתן להצפין (Encrypt) ולפענחו (Decrypt) את הטקסט וכן יכולות הגרמניות להעיבר בינהן מסרים, בהתאם להתקפות השונות ולקבל הוראות ביצוע מהדרג הפיקודי ביבשה, כל זאת מבלי לחשוף את תוכנותיה לעולות הברית. בעלות הברית מצין ביצעו מאמצים רבים וניכרים (שנשאו פרי לבסוף.. ! T. A. Go) לפענחו ולפצח את מכונת האניגמה ולהבין את תוכניות הגרמנים מעוד כדי לסכל ולמנוע אותן לעיתים, וכך להציג את הגרמנים במידע שווה לעיתמים אחרים (בתחרבות תעשה לך מלחמה...).

במהלך תרגיל זה נדמה את השלבים השונים בעבודה עם מכונת האניגמה. ראשית נבחן את מבנה המכונה ובנונה צו בעצמיינו (בקוד כמובן...). לאחר מכן נרחיב את מבנה המכונה למוכנה אוניברסלית שאינה כובלה למוגבלות המכונה הפיזית.

### מבנה מכונת האניגמה:

מכונת האניגמה הומצאה (ברסתה הראשונית) כבר בשנות ה-20, אולם היא שוכלה עוד ועוד, עד להגיעה לשיא הפעולות במהלך מלחתת העולם השנייה. המכונה נראהית כ"סוג" של מכונת כתיבה של פעם, רק שיש בה "מקלדת" נוספת שבה יש נורה לכל אות (מקלדת הנורות). בתהילך ההצפנה המפעיל לווחץ על אות במקלדת הראשית, וכORBזק רגע נדלקת נורה של האות המוקודדת (מקלדת הנורות). המפעיל רושם את האות שנורתה דליה, ומתקדם לאות הבאה בקלט. כך, אותן אחר אותן, מתקבלת מחרוזת מוצפנת. כדי לפענחו את המחרוזת המוצפנת ולקבל חזרה את המחרוזת המקורי, המפעיל (מצד השני) חוזר על אותו תהליך בדיקון, רק שהפעים המחרוזת המוצפנת מהוועה קלט (לקלדת הראשית) והמחרוזת המפעונחת תופיע במקלדת הנורות. במובן זה אין שימושות ל"פענוח" או "הצפנה" של טקסט במכונה – מדובר על עיבוד הקלט שנכנס והפקת פלט מתאים לו. "הציפנה" או "פענוח" נקבעים עפ"י ההקשר של מחרוזת הקלט (אם היא טקסט מקורי או מוצפן שנכנס למכונה).

מאפיין חשוב של מכונת האניגמה הוא העובה של אדי להחזיק את המכונה עצמה ולדעת את המבנה שלא כדי לפענח את הצופן, אלא יש לדעת את ה"קוד הסודי" והיחודי שבאמצעותיו הוצפנה הודעה כדי לפענח חזרה. (במציאות, אכן תפסו הבריטים והפולנים מספר מכונות אניגמה – אולם לא היה די בכך כדי לפענח את ההודעות והו צריים לעبور עוד כברת דרך נירת עד שהגיעו אל היעד הנכוף..).



מכונת האניגמה מורכבת ממספר חלקים מכניים ואלקטרוניים. בלבת המכונה ניצבים מספר גלגלי המררה ("רוטורים"). גלגל המררה הוא גלגל בעל X נויסות המחברות ל-X יציאות. הנויסות והיציאות מסומנים אותיות ה-ABC – ובמכונה הקלאסית יש 26 כאלה (אצלנו, בהמשך, יהיו הרבה יותר). גלגל המררה למעשה ממיר כניסה X ליציאה Y: לדוגמה הנויסת המזוודה עם האות A יכולה להיות מחוברת ליציאה המזוודה עם האות K. המררה היא דו כיוונית, כלומר-A->K ו-K->A. המכונה בניה על שלילוב של מספר גלגלי המררה הניביים אחד אחריו השני (בדרכ' 3 גלגים שכאלה, אצלנו יהיו הרבה יותר), כך שהיציאות של אחד מהם מחוברות לכניות של הבא אחריו. זרם חשמלי עובר בין הנויסות והיציאות של הגלגלים השונים בכיוון "הLOOR" (מהגלגל ימני ביותר אל השמאלי ביותר, דרך כל הגלגלים שבדרך) ואז גם בסיוון חוזר (מהגלגל השמאלי ביותר דרך כל הגלגלים שבדרך), כך שבסיום התהליך עבר כניסה E' ניתן לקבל יציאה Z'.

את הרוטורים השונים ניתן לשובב במקומות ולקבוע מקום ראשון מיקום ראשוני שוננה בכל פעם. כניותות הזרם החשמלי המ廟ופות למקשי המקלדת נותרות קבועות. ככלומר אם במצב רוטור מסוים, הקלדה על אות E שושגירה בנויסת 5 אל הרוטור ימני ולכן נפלת על מיפוי 0 מסוים, לאחר סיבוב הרוטור לפוזיציה אחרת, אותה הקלדה על האות E, עדין מגיעה לכניותה 5 אל הרוטור ימני – רק

שהפעם היא טיפול על מיפוי אחר (ח) וכו'. מכונת האנigmaה הכללה לכל רוטור מעין "חלונית" קטנה שאפשרה לראות מהו מיקומו ולסובב את הרוטור למקום מסוים ע"י הזרמו עד שמתאפשרת האות המבוקשת בחולונית (הפוזיציה הנדרשת).

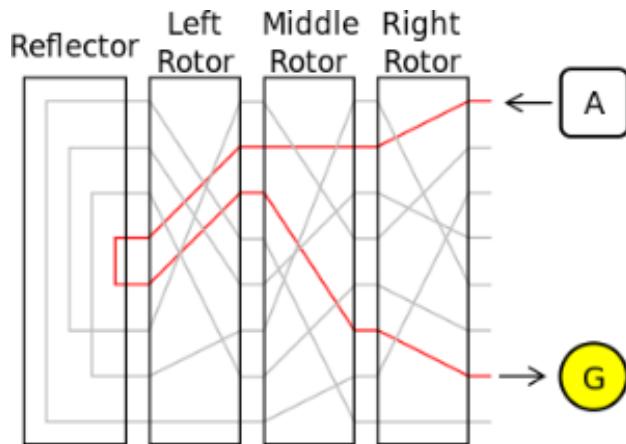
היות וכל רוטור מחוץ לאחרת (כלומר מיפויים שונים בין "אותיות", כלומר בין הכניסות ליציאות), קל להבין כי לקביעת סדר הרוטורים (מי הימני, מי באמצע, מי בשמאל), כמו גם למקום הראשון של כל רוטור (אילו אותיות ניבוטות מהחולונית) יש חשיבות מכרעת, קריטית, בהצפנה ובפענוח ההודעה.

אם זה לא מספיק, מכונת האנigmaה מגיעה עם סט של מספר רוטורים, אבל משתמשת בפועל רק **בחלקם**, בסדר מסוים, ובמיוחד התחלתי מסוים.

המכונה מורכבת מרכיב נוסף הנקרא **רפלקטור** (" משקף"). הרפלקטור ניצב אחרי סדרה הgalils (מצד השמאלי של המכונה) ותפקידו לקבל זרם חשמלי בכניסה ולהחזיר אותו ביציאה חזרה לסדרת הgalils (reflect - לשקף). הוא האחראי לתהיליך ה"חזר" בהצפנה. גם כאן המכונה מגיעה עם מספר רפלקטורים, ויש צורך להכירו במירפלקטורים משתמשים.

**שימוש לב:** ברגע לרוטור, אין משמעות ל"מקום" הרפלקטור – הוא לא רוטור שניtan לשוב ולשנות את מיקומו.

אני ממליץ בחום לראות את [הסרטון](#) הזה (אנימציה; 20 דקות) המסביר לפוטוטו את המבנה המכני/אלקטרוני של מכונת האנigmaה. זה יעזר לכם "להיכנס לעסוק" 😊



אם כן, ממה מרכיב "הקוד הסודי" של מכונת האנigmaה ?

המכונה (הבסיסית) מגיעה עם מספר רוטרים אפשריים לשימוש, ממוספרים בספורות 1..ח. יש לבחור 3 רוטרים לשימוש מהם, ולקבוע את סדר הרוטרים (ימני;אמצע;שמאל). לאחר מכן יש לקבוע את המיקום ההתחלתי של כל רוטר. על כל רוטר יש את 26 אותיות הABCijk שפשוט קובעים מי האות המהווה את המיקום הראשון של כל רוטר. לסימן, יש לבחור באיזה רפלקטור משתמשים (יש כמה מהם. מסומנים בכתב רומי).

אולי זה נשמע מסובך, אבל בעולם הקRIPTוגרפיה – אין כאן אלא צופן החלפה פשוט (כל אות מוחלפת באות אחרת), ואת זה קל מאוד לפענה גם אם לא יודעים את קוד המקור (זהה גם קל מאוד במהלך מלחמתWWII).

כדי להקשות על פענוח הצופן עוד יותר, מכונת האנigmaה השתמשה בטריק פשוט (אר זדו): בכל הקשת אות כקלט, רגע לפני שילוח הזרם דרך הרוטרים בכיוון "ולוך"-> "חזר"-< " משקף ". המכונה הייתה מקדמת את הgalil הימני ביותר צעד אחד קידימה ("פסיעה"). כשהgalil הימני היה מגיע לפוזיציה מסוימת, זיז קטן (notch) המקובע על הרוטר הימני היה גורם לgalil הבא אחריו להתקדם גם כן בפסיעה אחת. בדומה לכך, כשהgalil המרכזי היה מגיע לפוזיציה מסוימת, זיז קטן המковע על הרוטר המרכזי היה גורם לgalil הבא אחריו להתקדם גם כן בפסיעה אחת. (חישבו על מבנה של ספידומטר).

למעשה כל הקלדת אות קידמה את אחד (או יותר) המgalלים בשלב אחד קידמה ובכך למעשה שינתה את המיקום של הgalלים, וכפועל יוצא את כל החיווט עבור כל אות שהופנה (!)

כלומר מכונת האנigmaה מצפינה בצווף שאיןו 'צפן החלפה' פשוט: אותה אות שתוקלד כמה פעמים רצוף תוצפן בכל פעם לפחות אחת (!!) הדבר מוסיף ממש קושי נוספת לניסיונות לפענוח את הצפנים בטכניקות שהו ידועות עד אז, המבוססות על ניתוח תדריותן של אותיות וכו'.

מיקומו של הזיז בכל galil הוא שרירותי (כלומר לא כל הזיזים ממוקמים בפסיעה ה 26, כפי שallow היו הם מצפים...), ועם כל הזיז של כל רוטור – הזיז שלו "זז" ייחד אליו (נסו להגיד את זה כמה פעמים רצוף ותראו מה יוצא ...).

כל הgalלים מגיעים עם הגדרת הזיז שלהם, ולאחר מכן הוא מקובע במקומו הרלוונטי בכל רוטור ורוטור.

פרט נוסף לשים לב אלו הוא שבשל קיומו של המשקף, אותן לעולם לא יכולה להיות מוצפנת עצמה (חישבו מדוע?). (בדיעבד, פרט זה דוקא יצא בעוכריה של המconaה שכן הוא סיע (בمعט) לכוחות הבריטיים בפענוח הצופן אחרי הכל...).

הספריך לכם? עוד לא? אז...

הרכיב האחרון היה לוח התקעים:

לוח התקעים הכיל כניסה לכל אותן מהאב' (26 אותיות ה-ABC) ואפשר למשה להגדיר צמדים של אותיות מתחלפות. המפעיל היה מחבר כבל עם 2 פינים לתוך התקעים המסומנים באותיות ובכך יוצר צמד אותיות להחלפה (נגיד K ו-D או למשל Z ו-O) וזה אומר שכל פעם שאותיות אלה מגויות אל לוח התקעים – הן מתחלפות.

אות הגעה אל לוח התקעים פעםיים:

ב"הלו", מיד אחריו שהמפעיל הקיש על אותן (נגיד K), היא זרימה אל לוח התקעים. במידה והיא הייתה חלק מצמד מתחלף, הרי שעכשו היא הייתה מתחלפת בהתאם לצמד שלה (במקרה זה D). אותן D הייתה זו שמאיה עכשו אל החלק הארי (הבנייה?) בדיחה!) של המconaה (רוטורים -> משקף -> רוטורים) והוא זו שהיא מוצפנת (נגיד ל-Z).

בדרכה "חזרו" אל מкладת הנורות, היא הייתה עוברת שוב בלוח התקעים, ואולי פוטנציאליות מוחלפתשוב (למשל במקרה שלנו מוחלפת ל-O) ורק אז התוצאה הייתה זורמת למkładת הנורות.

אות שמאיה לוח התקעים ואיננה חלק ממיפוי – נותרת בעינה ויוצאת ממנה שם שבאה. (נגיד בדוגמה שלנו אותן A נשארת A)

מאפייני שימוש בלוח התקעים:

- במבנה האניגמה נהוג היה להשתמש עד 10 צמדים, מטור 13 אפשרים סה"כ. (אצלנו השתמש בכמה שרק נרצה.)
- בוגוד לשאר הרכיבים – אין חובה להשתמש בלוח התקעים.
- אותן לא יכולה להופיע לצמד עם עצמה.
- אותן יכולה להופיע לכל היתר לצמד אחד (או בכלל לא)

פשוט ונאיyi ככל שהוא נושא – לוח התקעים הוסיף מימד שהכפיל בכמה סדרי גודל (!) את הקושי לפצח את מבנות האניגמה (ראו הפניה לסרטון בהמשך המפרט את הסוגיה).

אם כן, מבנה הקוד במבנה צריך להכיל את הפרטים הבאים (בתצורה הבאה):  
בහינתן מבנה עם 5 רוטורים אפשריים, אך צזו משתמש רק בשלושה בפועל, ו-3 משקפים אפשריים, אז הינה מספר דוגמאות לפענוח הקודים הבאים:

#	מבנה קוד	משמעות
1	<2,3,1><A(1),I(5),C(15)><II><A D,K E>	בחרי את רוטורים המסומנים 1,2,3 הצב אותן קר ש 1 בימין, 3 במרכז ו-2 בשמאלי. קבעי את המיקום ההתחלתי של הרוטורים קר ש 1 ממוקם ב C (זהזיז שלו מרוחק 15 פסיועות מחלונית היציאה), 3 ממוקם ב I (זהזיז שלו מרוחק 5 פסיועות מחלונית היציאה) ו 2 ממוקם ב A (זהזיז שלו מרוחק פסיועה אחת מחלונית היציאה). השימוש במשקף מספר 2 (II). השימוש ב-2 תקעים: בין A ו-D ובין K ו-E
2	<I><5,1,4><D(5),O(2),W(0)>	בחרי את רוטרים המסומנים 5,1,4 הצב אותן קר ש 4 בימין, 1 במרכז ו-5 בשמאלי. קבע את המיקום ההתחלתי של הרוטרים קר ש 4 ממוקם ב W (זהזיז שלו נמצא מימין מחלונית היציאה), 1 ממוקם ב O (זהזיז שלו מרוחק 2 פסיועות מחלונית היציאה) ו 5 ממוקם ב D (זהזיז שלו מרוחק 5 פסיועות מחלונית היציאה). השימוש במשקף מספר 1 (I). אין שימוש בתקים.

בתרגילים זה, כל הגדרות המconaה יוגדרו בקובץ בתצורת XML שմבנהו ופרטיו מסווגים בהמשך. לאורך התרגיל אנו נבנה מבנה איניגמה "גנרטית" – צזו שאינה מציאות למוגבלות המconaה הפיזית ושניתן להרחיבה ככל העולה על רוחינו (ואכן עלה על רוחינו להרחיבה...).

על מנת לשמר על איחדות ותאימות, כניסה קבועות יקבעו על פי סדר הופעת האב' (מוסיפות מ-1 עד ח'). הגדרת רוטור נעשית באמצעות הגדרת 2 "טוררים" של אותיות האב' (המופוזרים רנדומלית בין לבין עצמם). הטורים נקראים right ו-left והם מדירים את "המייפויים" של כל רוטור ורוטור הנקבעים ע"י מציאת האות эта ב right וב left.

מחלונית היציאה נקבעת להיות במיקום מס' 1. כאשר ה-2 מגיע למקום זה, או אז הגעה העת לבצע את מהלך הפסיעה של הרוטור הבא בתור. (מהלכי הפסיעות מתקדמים אך ורק מימין לשמאלי; רוטור פועל רק בעקבות הגעת ה-2 של הרוטור מימין לו לחلونית היציאה)

כאשר מקבעים רוטור מסוים לפוזיציה מסוימת – המדובר הוא על הופעת האות/מיקום הרלוונטי על פי עמודות ה-right. יש להבהיר את הרוטור קר שהאות בעמודה right הנדרשת תופיע בחلونית היציאה מול מקום מס' 1.

יעקבו אחר [נספח א'](#), [נספח XML](#) כדי לקבל את מלאה הפרטים בהקשר זה.

כמה אפשרויות מתקיימות לקוד במכונת האנigma?

לדוגמא: במכונת אנigma סטנדרטית, המגעה עם 5 רוטורים, אבל פעולה רק עם 3, על כל רוטור יש בדיק (אך ורק) את 26 אותיות ה ABC, וברשותינו 2 משקפים לבחירה, ולא תקעים (ובנהנזה שאנו ידעים זאת), ניתן לחשב את כמות האפשרויות בצורה הבאה:

כמות האפשרויות לבחירת 3 גללים מתוך 5 גללים אפשריים, בלי חזרות (הbinsom של ניוטון - ח' מעל א') X

כמות האפשרויות לסידור 3 גללים בין עצם (בל' חזרות) X

כמות המשקפים שברשותינו X

כמות האפשרויות לבחירת קוד התחלתי על פני 3 גללים

ובנוסחה:

$$5!/[3!(5-3)!] = 17576 = 2,109,120$$

([בסרטון זהה](#) ניתן לראות את החישוב המלא כולל את רכיב לוח התקעים. אם חשבתם שוגול זה מספר גדול...)

קל לראות שהרכיב המרכזי הוא כמות האפשרויות לבחירת קוד התחלתי, ואלו תלויים בכמות האותיות האפשריות לקידוד. סט האותיות האפשריות לקידוד במכונה יקרא אלף-בית של המילה (Alphabet). אצלנו בתרגיל, האלף-בית של המכונה יכול להכיל גם ספרות ותווים מיוחדים ויהיה גדול הרבה יותר מ 26, כך שמספר האפשרויות מגיעה בקצבות מיליונים אם לא יותר.

בקיצור, פענוח ההצפנה הוא:

## משימה קשה במיוחד !

מגבליות ומאפיינים של מכונת האנigma בתרגיל זה:

1. סט התווים הסופי שנייתן לקוד במכונה יכול להשתנות ולהכיל תווים שונים ככל שנגיד לך זאת. התווים יכולים להיות אותיות ה ABC, ספרות, תווים מיוחדים (!#\$%^&@#%), تو הרווח ועוד.

מובטח כי לא יכולו התווים הבאים:

טאב (\t), enter (\n), esc (027)

2. גודל האב' של המכונה חייב להיות זוגי (בעיקר בשל מגבלות המשקף ולוח התקעים)

3. המכונה תוגדר לכל היותר עם 5 משקפים לבחירה (מהם צריכים לבחור בדיק אחד)

4. המכונה תוגדר לכל היותר עם 99 רוטורים בשימוש (שים לב: יכולים להיות מוגדרים יותר מ 99 רוטורים, אולם לכל היותר 99 יהיו בשימוש)

5. משקף לעולם לא יכול לבצע מיפוי בין אותן לעצמה.

6. בהתחלת יתכן מיפויים ברוטורים בין אותן לעצמה.

7. מיפוי כל רוטור הוא מיפוי דו כיווני ("הלאן": left to right ו"חזרה": right to left).

8. על מנת לישר קוו, כיוון עיבוד התווים יהיה מימין לשמאל – כלומר הרוטור הראשון שנבחר הוא הימני והאחרון שנבחר הוא השמאלי. מקלדת הקולט ומקלדת הפלט (הנורות) נמצאות מימין לרוטור הראשון והמשקף נמצא ממשאל לרוטור האחרון:

מקלד(ו)ת -> רוטור 1 -> רוטור 2 -> ... -> רוטור ח -> משקף

כרגע במתן והגדרת קוד המכונה (ראו פרטים בהמשך).

9. נוכל להשתמש במקרה תקעים האפשרי, או בכלל לא. ככל שנרצה.

## הנחיות ספציפיות למימוש תרגיל "בנייה מכונת האניגמה"

1. המטרה היא לבנות מנווע מערכת גנרי, כזה שידע לקבל את הפרטים לגבי אופי המע' (מכונת האניגמה) מתוך קובץ נתונים בפורמט XML (עובדת עם XML'ים תילמד במהלך הקורס כמובן).  
מנוע המערכת הגנרי יocr וישתכלל מתרגיל לתרגיל, בהתאם לפיצרים השונים. כך תוכלו לחווות מהלך של מוצר החל מרעיון קטן במימוש בסיסי וכלה במנוע מע' המונע אפליקציית ווב שלמה.
2. חלק מהמע' תצרכו לחשב ולבחור לבד את מבני הנתונים השונים שיישרתו את הצרכים של דרישות המע'. זהו לא קורס במבנה נתונים או באלגוריתמים, ובמבנה הנתונים/אלגוריתמים שתבחרו למש לא חיבטים להיות העילים ביותר או האופטימליים. מספיק שהם יעבדו בצורה **נכונה** (לא טעויות) ובזמן סביר.
3. ממשק המשמש השונים יפעלו מול מנווע המע' שיפורת מתרגיל בהתאם לדרישות. המנווע יכול, בין היתר, את מימוש המכונה עצמה, את היכולת לפענה את ההודעות המוצפנות.
4. כל המשקיקים הפונים אל המשמש החיצוני ומציגים לו מידע /או מבקשים ממנו מידע מסוים ספירה המתחילה מ 1. גם אם פנימית אתם ממשים את מי מהרכיבים כמערך או רשימה (از בסיס הספירה מתחילה מ 0) – עליהם להקפיד ולודא כי ככלוי חוץ "תדברו" אך ורק במקרים של בסיס 1.
5. המחרוזות והתוויים שנמצאים/נענחו יכולים להתקבל גם ב lower case, אולם תוצאה העיבוד (הצפינה או פענוח) יוחזרו תמיד כ Uppercase. יש לדעת להתמודד עם 2 המ מצבים.
6. המע' יכולה תותואר בשפה האנגלית בלבד, עם ממשק משתמש המתנהל משמאלי לימין (במקומות הרלבנטיים)
7. המע' יכולת תיכתב ותורץ בסביבת העבודה של ג'אווה 21.

פרטים יבשים

**14.12.25**

תאריך הגשה:

ציון אפשרי מקסימלי: 105

כפי תחילת עבודה:

כפי זמן לביצוע: 4 שבועות

משקל התרגיל: 30%

## מטרת התרגיל העיקרית

1. הקמת מנוע המע' הבסיסי'
2. יצירת ממשך console לתפעול המערכת

דרישות

1. בתרגיל זה תקימו את מכונת האניגמה ותעמלו על יכולת התפעול התקינה שלה. את המכונה "תתפעלו" באמצעות ממשך console פשוט המציג תפריט פקודות שדרכו מפעילים את המכונה. בתרגיל תמשכו את מנוע המערכת, אשר יודע לקבל נתונים על המכונה ויודע להציג לכל פניה שмагעה משכבות ממשך המשמש, לעבד את הקלט ולהחזיר פלט רלבנטי.
2. פרטיה הטכניים של המערכת ינתנו מקובץ XML (כמפורט [בנושא ג](#)).  
מגבילות טכניות למכונה:
  - עובדה עם 3 רוטורים בשימוש ברגע נתון (אך יכולם להיות מוגדרים יותר מ 3)
  - עובדה ללא לוח תקעים
3. יש לוודא תקינות קלט כחלק מכל אינטראקציה עם המשתמש, ובכל מקום שבו זה רלוונטי: אם אתם מצפים לקבל מספר – לא לקרוס כי הכוינו לכם בטעות (או בכוונה) טקסט וכו'. בכל מקרה של תקללה יש להיות מאד ברורים במסר שمبرירים חזרה למשתמש: מה קרה? מה הייתה מהות התקלה? היכן שזה רלבנטי, איך לתקן וכו'.  
חייבים אף להיות ידידותיים למשתמש ולעולם אל תניחו כי מי שמשתמש באפליקציה שלכם הוא מתכוון בעצמו או מישחו שמדובר במתחום ו"מכיר" איך דברים עובדים בלבד. (זה הזמן לחשב על...)
4. אין צורך להשתמש בצעדים שונים במהלך תרגיל זה בעט ההדפסה ל console.  
יתריה מזאת, ישנו צפי רב (ניסיונות מהסטודנטים הקודמים) כי ניסיון לעשות כן תור שימוש בספריות צד שלישי קורס אצל הבודק, מעווית את כל תצוגת המסך וגורם לחוסר יכולת לבדוק את ההגשה.  
גם אם בדקתם את זה אצלם וזה עבד.  
גם אם בדקתם במחשב של השכנה וזה עבד.  
כמו כן אין לנוקות את המסך בין פקודה לפקודה.

5. עליכם לכתוב ממשק משתמש בתצורת `soiso.so`.  
משק המשתמש יכול סט סופי של פקודות שדרכו ניתן יהיה להפעיל את המערכת.  
אחרי הצגת תפריט הפקודות יש לחכות לקלט מהמשתמש באשר לפעולהו הוא רוצה לבצע. לאחר ביצוע הפעולה (שאול תגורור בקשה קלט נוספת מהמשתמש) יש להציג את הפלט החוזר ממנה (לכל פקודה יש פלט החוזר ממנה) ואז להציג שוב את התפריט וחוזר חלילה.

#### שימו לב:

- ישן פקודות שאין הגיון לבצע אותן אם לא קדמו להן פקודות אחרות. במידה זה קורה יש להציג הודעה שגיאה רלבנטית למשתמש ולאפשר את המשך מהלך פעילות המע'.
- בכל המקרים שבהם מציגים רשימות של דברים וצריך לאפשר למשתמש לבחור פריט(ים) מסוימה – יש לאפשר בחירה זו ע"י הגדלת מספר לכל אחת מהאפשרויות ולאפשר לו לבחור על פי המספר המזהה של האפשרות מהרשימה (או באמצעות כמה מספרים במקומות הרלבנטיים).  
כולומר, אין להניח שהמשתמש הולך להקליד לכם מלל חופשי של תיאור האפשרות!  
זכרו כי הסFOROR כלפי חוץ (המשתמש) תמיד מתחילה מ 1 (גם אם פנימית אתם סופרים מ 0)

רישימת הפקודות שיש לתמוך בהן:

#### 1. קריית קובץ פרטי המע'

פקודה זו טעונה את פרטי המערכת מתוך קובץ נתונים בפורמט XML.  
קובץ דוגמא מתאים המעלוי מבועוד מועד לאתר הקורס ואתם מוזמנים להורידם ולחזור אותם בהתאם.  
(אתם מעודדים ליצור לעצמכם קבצי בדיקה נוספים כדי לבדוק את המע' בצורה יסודית וטובה יותר כאותו נפscanfם).

יש לבצע מהמשתמש נתיב מלא לקובץ XML אותו הוא רוצה לטען למע'.  
הנתיב יכול להכיל רוחcis בתוכו (למשל "program files") ויש לוודא כי הדבר לא מכשיל אתכם (ולא שהוא אמרו).  
הנתיב יכול רק אותיות באנגלית (לא גיבריש של אותיות בעברית וכו')

הקובץ יהיה קובץ XML שפרטיו וחוקיו המפורטים מובאים בנספח ג' לתרגום זה.  
אתם מצופים לעבור על פרטיים אלה ולהתיחס אליהם כחלק מהגדרת התרגום.

עליכם לוודא בדיקת קלט לקובץ XML ולוודא כי הקובץ מכיל מידע תקין ואמין.  
(מובטח כי הקובץ יהיה תקין schema-wise אבל לא בהכרח תקין application-wise)...)

בפרט יש לוודא את הפרטים הבאים:

- הקובץ קיים, והוא מסוג XML (די לבדוק לשם כך כי הוא נגמר בסימנת .xml).
- גודל ה ABC הוא זוגי.
- מוגדרים לפחות 3 גלגלים (רוטורים) במסגרת הקובץ.
- לכל רוטור יש po'יחודי. כל ה po'ים של הרוטורים השונים צריכים בסוף להוות ספרור רץ המתחילה מ 1, ולא חורים.
- רוטורים לא חייבים להופיע על פי הסדר.
- אין מיפויים כפולים באותו רוטור (כלומר אותה אות מיפויה פעמיים בעמודת ה right או ה left)
- זיז הפסיעה (tch) בכל רוטור מוגדר בטוח גודל הרוטור (פונקציה של גודל ה ABC)
- לכל משקף יש מספר סידורי בשפה הרומנית. כל ה po'ים השונים צריכים בסוף להוות ספרור רץ של הספרות הרומיות 1-5: V, VII, III, II, I). הם אינם חייבים להופיע על פי סדר הספירה הטבעי.
- אין מיפוי בין אותה הכנסה לעצמה ביציאה באף לא אחד מן המשקפים המוגדרים

במקרה והקובץ לא תקין יש לדוח זאת למשתמש בצורה ברורה כך שניתן יהיה להבין מה לא תקין בקובץ.  
אין לקרוס על exception במידה וקובץ מתגלה כאינו תקין; יש לאפשר למע' להמשיך לפעול במצב זה.  
(כחלק מבדיקה התרגום יטענו למערכת קבצים שאינם חוקיים כדי לבדוק מהי התגובה).

במידה והקובץ נמצא תקין – יש לטען את פרטיים למע' וליצור מופיע של המכמה הנוצר ממנו. יש לדוח על כך למשתמש.

### דgesim:

- יש לאפשר למשתמש לטעון כמה קבצים אחד אחריו השני (כלומר להפעיל את הפוקודה כמה פעמים רצוף). כל קובץ תקין "דروس" לחלוון את כל פרטיה הקובץ (התקן) שהיה טעון לפניו במאן (כל שהוא צה). כל נסوان טעונה של קובץ תקול לא דרס את פרטי הקובץ (התקן) האחרון שהיה במאן (כל שהוא צה)  
פוקודה זו מוצגת ומאפשרת תמיד. אפשר לבחור בה בכל רגע נתון במאן.

## 2. הצגת מפרט המcona

פוקודה זו מציג את מפרט המcona המוגדרת בהתאם לקובץ התקין האחרון שנטען בהצלחה.  
יש להציג את הפרטים הבאים:

1. כמות רוטורים במאן (סה"כ הכמות האפשרית לבחירה)
2. כמות משקפים
3. כמה הודיעות עובדו במכונה עד כה, סה"כ, מאז שנטען הקובץ, עבור דרך כלל הקודים שהוגדרו בה (פרק' 3 או 4)
4. תיאור תצורת קוד מקורי (במידה וקיימת ; לאחרונה שנקבעה ע"י פוקודה 3 או 4)
5. תיאור תצורת קוד הנוכחי (במידה וקיימת ; יתכן שינוי מתצורת הקוד המקורי בעקבות עיבוד קלט – פוקודה 5)

את תצורת קוד המcona (סעיפים 4 ו 5) יש להציג בפורמט הבא (זהו הפורמט והוא מחייב):

<מספר משקף בספרות רומיות><מייקום גלגלים ומרחקי זיזים מהחלונית><בחירה וידור גללים בחריצים>

דוגמא (ראו הסבר צבעוני בהמשך):  
(הצבעים נועדו להבהרה בלבד. אין צורך להשתמש בצבעים במסגרת תרגיל console !):

<45,27,94><A(2),O(5),(!)><|||>  
<198,346,123,959><|(4),=(10),3(52),K(3)><\V>

זכרו כי מיוקמי הרוטורים נקבעו מימין לשמאלי, על אף שمدפיים את הנתונים הנ"ל משמאלי לימין...  
(בדוגמא הראשונה, הרוטור הימני ביותר הוא 94 והוא מכון לתו '!'. הרוטור השמאלי ביותר הוא 45 והוא מכון לתו 'A')

הסביר על הרכיבים השונים:

1. **מספר גללים שנבחרו + סדר ביניהם – סדרת מספרים מופרدة בפסיקים (,).** רוטור לא יכול להופיע פעמיים.
2. **מיוקמו של כל רוטור – נתון סדרת תווים חוקיים מהאלפ-בית של המcona, מופרדים בפסיקים.**  
ליד כל אחת, בתוך סוגרים עגולים, מופיע מרחק הזמן שלו מחלונית ההוצאה. 0 אומר שהוא ממש בחלונית ההוצאה.
3. **מרקף נבחר – נתון בספרה רומיות: V, IV, III, II, I,**
4. **כל אחד מהמקטעים השונים (1-3)عطוף בתוך סוגרים משלוחים <>**

מליצ' ללמידה לעבוד עם StringBuilder כאמצעי לבניית המחרוזת (המתוסבכת משהו) צעד אחריו צעד בצורה יعلا  
(ובעיקר חביבה)

### דgesim:

- פוקודה זו מאפשרת רק לאחר שטען קובץ תקין במאן

### 3. בחירת תצורת קוד נוכחת (בצורה ידנית)

פקודה זו מאפשרת להגדיר את תצורת הקוד שיישמש להצפנה ופענוח במצבה הנוכחי.

תצורה זו תורכב מ 3 חלקים, כפי שוגדר בפקודה 2 סעיף 5

יש לבקש מהמשתמש את הקלטים בכמה מנת, בסדר זהה (אחרי כל קלט, מתקדים לקלט הבא):  
א. הרוטורים ממושפרים בספרות עשרוניות. יתקבלו כרשימה שבה הם מופרדים באמצעות התו פסיק (.) .

דוגמא: 23,542,231,545

על אף שהקלט מכונס משמאלי לימין (מתחלים מ 23) הכוונה היא כי הרוטור اليمنי ביותר הוא 545 וכו'

ב. מיקומי הגלגים הראשונים הם חלק מה-אב' של המע'. יש להכניסם כרצף תווים צמודים.

דוגמא: 4D8A

בקטל הנ"ל, על אף שהconoנס משמאלי לימין (4 הconoנס ראשון) – האות A היא המיקום של הרוטור اليمنי ביותר (545)

ג. מספר המשקף יציין בספרה רומיות (V, II, III, IV).

יש להציג למשתמש רשימה מספרית ולקבל ממנו מספר עשרוני, המציין את המשקף הרלוונטי בספרה הרומיות.

אין לצפות מהמשתמש להכניס לconiנס ספרות רומיות!

דוגמא: 3 פירשו III + 5 פירשו V וכו'.

יש לבצע בדיקת קלט למשתמש ולהתמודד עם מצבים שבהם הוא מקליד אותיות היכן שמצופים מספרים או להיפך, או כל ניסיון אחר שלו להזין קלט שאינו תקין ואינו עומד בziehtypם. במידה וזה יקרה (זהה יקרה לבטח!) יש להודיע על כך בצורה מסודרת למשתמש – תוך הסבר מפורש מה הבעיה שנתקלתם בה וכיצד הוא יכול לטקינה.

במידה ואיתרתם תקלה ודיווחתם אותה למשתמש, תוכלו לבחור איך להתקדם מכאן:  
להחזיר את המשתמש לתפריט הראשי (שייחיל מההתחלת) או לאפשר לו להישאר "בתוך הפוקודה" ולתקן רק את הרכיבים) הלא תקנים. כך או אחרת אין "לכלוא" את המשתמש בתוך בפקודה (עד שיכניס קוד תקין) ויש לאפשר לו לחזור חזרה לתפריט הראשי אם ירצה.

לאחר קביעת הקוד הסודי יש לכוון את המכונה אליו, כך שעיבוד הקלט שיגיע בהמשך יתרחש בהקשרו. שימוש לב Ci יש למקם את הרוטורים בהתאם למיקומיהם כמו גם את הпозיציות הראשונות של הרוטורים יש למקם על פי מה שהוגדר בעמודות ה right .

**דגשים:**

- פקודה זו מאפשרת רק לאחר שטען קובץ תקין במע'

### 4. בחירת תצורת קוד נוכחי אוטומטית

פקודה זו תבצע בחירת קוד לצורה אוטומטית עבור המשתמש. כל פרטיה הקוד יוגלו לצורה רנדומלית לחלוין. בסיום הפקודה תוצג התצורה הנבחרת למשתמש והיא תיקבע כתצורה האקטיבית במצבה.

ممליץ לקרוא על המחלקה [Random](#) אשר מאפשרת לבדוק את זה...

**דגשים:**

- פקודה זו מאפשרת רק לאחר שטען קובץ תקין במע'

### 5. עיבוד קלט

בפקודה זו המשתמש יזין קלט ויקבל בסופו את המחרוזת שהזין כשהיא מעובדת (מצפנת או מפוענחת – הכל תלוי בהקשר).

השימוש יתרחש עם תצורת הקוד שהוכנסה אחרונה למערכת.

בסיום פקודה זו הרוטורים לשארים במקומם הנוכחי וains מוחזרים אוטומטית לקוד הראשוני שאוחד במצבה (פרק' 3 או 4).

**דגשים:**

- פקודה זו מאפשרת רק לאחר שטען קובץ תקין במע'
- אין לאפשר הפעלת פקודה זו באם טרם בוצעה פקודה 3 או 4
- יש לוודא כי המשתמש לא מזין אותיות שאין חלק מה-אב' של המע'. (להודיע לו בטענה מסודרת כמפורט לעיל אם זה קורה)

## 6. ביצוע איפוס קוד נכון

זכרו כי בכל הקלדות אוט, הרוטוריים השונים מבצעים פסיעות (גלאי ימי תמיד מבצע פסיעה; אחרים לעתים). על מנת לקודד מספר הודעות ברצף אשר כולם מתחילה מאותה תזרחה ראשונית – יש לאפשר לאחר כל הצפנה/פענו. פקודה זו תאותחל את הרוטוריים לקוד החוקי האחרון שנבחר ע"י פקודה 3 או 4 (הוא "הקוד המקורי")

### dagshim:

- פקודה זו מאפשרת רק לאחר שטען קובץ תקין במע'
- אין לאפשר הפעלת פקודה זו באם טרם בוצעה פקודה 3 או 4

## 7. הסטוריה וסטטיסטיקה

פקודה זו תציג את הסטוריה הפעילות במצבה הנוכחי כמו גם מספר נתונים סטטיסטיים לגבייה:  
א. יש להציג את כל תוצאות הקוד השונות המקוריות (כלומר אלה שנקבעו באמצעות פקודה 3 ו 4) אשר בוצעו במע' עד כה (על פי הפורמט המוגדר בפקודה 2.4)  
ב. עבור כל תוצאה קוד שכזו, יש להציג את כל המחרוזות שעברו עיבוד (פקודה 5), כמו גם את משך הזמן שהקח לבצע את התהליך.

הפלט יוצג בפורמט הבא:

# המחרוזת מעובדת <> --> המחרוזת מקורה  
מספר סידורי רץ המתחליל מ 1, והמחרוזות מוקפות בסוגרים משולשים <>

### dagshim:

- פקודה זו מאפשרת רק לאחר שטען קובץ תקין במע'

## 8. יציאה מהמערכת

פקודה זו מסייםת את פעולה התוכנית.

## חלוקת למודולים

בתרג'il זה חובב לייצר (לפחות) 2 מודולים (מהם תפיקו בהמשך 2 jar'IM):

1. ממשק ה צו, המציג את התפריטים השונים, אחראי על קליטת קלט מהמשתמש והחזרת הפלט למשתמש. שמו לב זה המודול "האקטיבי", המניע את כל המע'. הוא זה האחראי על פניה ותפעול מגוון המערכת. כפועל יוצא, כל ההდפסות של מיידומים למשתמש (println(*System.out*)). מtbodyות אר ווּך מתוך מודול זה ; במודול זה יושבת מטודת *the main* ; מודול זה אחראי על לולאת תפעול המע' העיקרי, הצגת התפריטים, איסוף הקלט מהמשתמש, הצגת הפלטים למשתמש וכו'.

2. מנוע המערכת, האחראי על קבלת הפקודות (מודול ה צו), ביצוע והחזרת פלטים מתאימים. שמו לב שמודול זה "פסיבי", והוא רק מגיב לבקשתו ולפקודות המתקבלות ממוקורות בלתי ידועים לו (בתרג'il זה מודול 1#). בתרג'ילים הבאים מקורות נוספים יפנו אליו לקבלת מידע וחשוב מאוד להקפיד על כך **שמודול זה אינו מכיר/מודע למי פונה אליו**.

## איך מתחילה ? (המליצה...)

התחלו מהבנתה מבנה ופעולות המכונה בצורה מוחלטת. שחקו את זה על הנייר, חפשו סרטונים בYouTube, ריכשו מכונת אנימציה מאספן מציאות - העיקרי וודאו כי התהילר מובן והגינו לכם לאשרו.

צרו פרוייקט חדש ב IntelliJ IDEA שישמש כפרויקט האב לכל התרגילים. בתוך הפרויקט צרו מודול נפרד שיכיל את מימוש המכונה עצמה. פרקו את מבנה המכונה לחלקים המרכזיים אותה (רוטוריים, משקף, מקלחת וכו') והתחלו למשם אותם ואת הקשרים ביניהם.

בידקו את תפעול המכונה בצורה מסודרת דרך מטודת *main* קטנה כדי לוודא שהמכונה **לכשעצמה** עובדת כהלכה.

צרו מודול נוסף מעור מע. המנוע יחשוף סט של יכולות (הפקודות השונות בתפריט), יחזק מופע של המכונה ויתוור את הפקודות השונות מול המכונה, הולך וছזר. התחלו בימוש פקודה 1 (על כל היבטי הטעינה), ופקודה 2 שתאפשר לכם לראות כי ביצתם את המידע החדש. עיברו למשם את פקודות תפעול המכונה עצמה (3,6,5,4) ורק בסוף התעסוקו עם סוגיית ההיסטוריה והסטטיסטיקה (פקודה 7)

לסיום צרו מודול נוסף מודול שכבת ה UI (console). זה המודול שבו תשב בסופו של דבר מודודת ה `main` הראשית שתתפעל את כל המערכת. זה המქם היחיד שבו מוצג פלט (System.out.println) ונאסף קלט (scanner) מהמשתמש. זה המქם המכיל את לולאת התפריט הראשית המניה את כל המערכת. שכבת ה UI תכיל הפניה (reference) למופיע המנווע (שבתוכו מכיל הפניה למופיע המכונה) וכן תוכל להעביר וlator גם לו את הפקודות הנאספות מהמשתמש ולהציג חזרה את הפלטים החזרים מהmenoוע.

#### בונוסים

#	סוג	מהות	למה שווה לי ?	כמה שווה לי ?
1	מגביב ! לאלה !	שמירה וטינה של מצב נתון של מכונה. במצב זה המכונה הנוכחית תשמר לקובץ חיצוני (באייזה פורמט וטכנייה שתבחרו), כולל תצורת הקוד הנוכחי, כל ההיסטוריה והסתטיסטיקה שנאספה עד כה וכו'. יש להוציא פקודה שמאפשרת לשומר את מצב המכונה וגם פקודה המאפשרת לטעון מכונה קיימת (מקובץ שנשמר זה לא מכבר), בנויגוד לטיעינה רגילה מקובץ ה XML של התרגיל. יש לאפשר למשתמש לבחור את הנטייה המלא כולל שם הקובץ (בל' הסיום) שהוא היה מעוניין לשומר את המכונה אליו (ולטעון אותה ממנו)	5 נקודות (מעל ל 100)	Ci עם תכנון נכון זה אמרור להיות משוחה כמו 5-4 שורות...

#### סבירה, סימתי. מה ואיך להגיש ?

יש להגיש קובץ `zip` המכיל:

1. 2 `jar`'ים (לפחות) שהם כל הקוד שלכם, בציরוף קובץ אצווה (batch) שהפעלתו תריץ את התוכנית (כלומר תבצע `<class name=`-jar `java`).
2. קובץ ההגשה יכול גם קובץ readme שיכיל הסבר על המערכת, בחירויותם השונות במקומות בהם הוא לכם בחירה, או כל דבר נוסף העולה על דעתכם חשוב שחשיבותה ידוע. הקובץ צריך להיות "בשער" שקיבלום במסגרת חומרה הומרי התרגילים.
3. יש לכלול בקובץ ה readme גם תיעוד וסביר כללי (וממצאה) של המחלקות הראשיות ותפקידיהם.
4. יש לכלול בקובץ ה readme קישור ל Github שלכם המכיל את קוד הפרויקט.

#### בונוס שימוש אבל לא יתועד – לא יבדק !

הגשה באיחור, שאינה באישור, תבטל כל שימוש בונוס. אין להגיש באיחור בשביל להוספיק לעשות בונוסים.  
תכנו את הזמן בהתאם.

פרטים יבשים

**13.1.25** תאריך הגשה:

ציון אפשרי מקסימלי: 100

כפי תחילת עבודה:

כפי זמן לביצוע: 4 שבועות

משקל התרגיל: 30%

## מטרות התרגיל העיקרית

1. מידול והעברת המע' לתוכה של maven
2. הוספה יכולת לעבוד עם יותר מ 3 רוטורים
3. הוספה תמייה בלוח התקעים

דרישות

1. במסגרת תרגיל זה נורחיב את יכולת המכונה לעבוד עם מספר דינמי של רוטורים, (ולא רק 3 כפ' שהייתה בתרגיל הראשון). כמות הרוטורים בשימוש המכונה תפיע בדמות נתון חדש בקובץ ה XML (עיקבו אחר השינוי בסכמה). שימוש לב Ci שינוי קובץ ה XML גורר שינוי בסכמה ומכאן גם הצורך לג'רט מחדש את המחלקות באמצעות הסכמה ו CXX. עלייכם לבצע בקבצי הקלט את כל הבדיקות שביצעתם במסגרת תרגיל 1 ובנוסף יש לוודא כי כמות הרוטורים בשימוש אינה גדולה מכמות הרוטרים המוגדרת במכונה.

2. תמייה בעבודה עם לוח התקעים:  
עליכם למשש את התמייה בעבודה עם לוח התקעים.  
لتמייה זו יש כמה השלכות על התנהלות המכונה:  
בדרכו הloor הקלט מהמשתמש עובר ראשית אל לוח התקעים, פוטנציאלית מוחלף בכך יש תקע מתאים ורק אז מגיע אל הרוטרים.  
בדרכו חוזר הפלט מהगלאל הימני עבור שוב אל לוח התקעים, פוטנציאלית מוחלף בכך יש תקע מתאים ורק אז מוחזר אל המשמש כפלט הקידוד.  
התמייה בלוח התקעים באה לידי ביטוי בכמה נקודות לאורך המע':

1. בפקודה 1 (טעינת קובץ) יש לבצע בנוסף לכל הבדיקות גם וידוא ובדיקה כי גודל האב' הוא זוגי.
2. בפקודה 2 (הציג מצב המע') יש להציג גם את התקעים שנעשה בהם שימוש (סעיף 2.4 ו 2.5).  
צורתה הקוד תראה מעכשו כך:  
**<צמד> תקע</צמד><מספר מסקלף בספרות רומיות><מקום גלמים ומרחקי זיזים מהחולנית><בחירה וסידור גלמים בחריצים>**

הנה דוגמא:

<45,27,94><A(2),O(5),!(20)><|||><A|Z,D|E>  
<198,346,123,959><|(4),=(10),3(52),K(3)><\V>

כל צמד התקעים יוגדר באמצעות זוג אוטיות מופרדות בסימן ה pipe (|). אם יש יותר מתקע אחד הוא יופרד בפסיק.  
(בדוגמה הראשונה יש תקע בין A ל Z ובין D ל E. אם A מגיע אל לוח התקעים הוא מוחלף ל Z ולהיפך  
זכרו כי אין חובה להשתמש בתקנים במסגרת הקוד (כפי שנראה בדוגמה השנייה)

3. בפקודה 3 יש לדרש מהמשתמש להכנס גם את הפלגים (לאחר שלב הכנסת המשקף).  
יש להכנס מחרוזת רציפה של תווים המהווים את כל הצמדים (הפלגים) הנדרשים במע'.

הצמדים יופיעו בצורה צמודה, ללא שום מפריד צזה או אחר. הקשה על `enter` תסימן את הכנסת המחרוזת.  
עליכם לוודא כי המחרוזת היא באורך זוגי (כי עסוקין בצמדים) ולחلك אותה צמדים על פי הסדר.

שימוש לב: מחרוזת ריקה היא קלט חוקי - פירושו שאין פלגים בכלל בע' כמו כן יש לוודא כי לא משתמשים באותה האות ביותר מפעם אחת, וכן אין מיפוי מאות לעצמה.

[dk49 !]

בקטל הנ"ל הוגדרו 3 פלגים:

D | K a

4 | 9 b

c. ! | (שימוש לבתו הרווח הואתו חוקי באב' והוא חולק את הפלג שלו עם התו !)

4. פקודה 4 יכולה גם לבחור רנדומלית את כמהות התקעים, את שילובי הצמדים וכו'.

5. בפקודה 5 יש כמובן להתייחס לתקעים במסגרת עיבוד הקטל במכונה. עיקבו אחר [נספח ב'](#) לתיאור מפורט זה

3. מעבר לארכיטקטורת פיתוח maven:  
יש לעדכן ולשנות את המימוש כך שהוא יתאים לצורת maven כפי שנלמד והסביר בכיתה.  
ה Group הראשי של המע' יקרא [patmal.course.enigma](#) וכל המודולים יחלקו group זה.

יעקבו אחר הדרישות הבאות בהקשר זה:

1. חלוקה למודולים

על המע' לתרמו בחלוקת (לכל הפחות) למודולים הבאים:

a. מימוש מכונת האניגמה. שם הארטייפקט: [enigma-machine](#)

b. מימוש מנוע המע' על כלל יכולתיו. שם הארטייפקט: [enigma-engine](#)

c. מימוש יחידת טעינת הקבצים בצורה עצמאית. שם הארטייפקט: [enigma-loader](#)

d. מימוש ה UI. שם הארטייפקט: [enigma-console](#)

"תכנו מודולים נוספים כרצונכם (גניך ל DTO ?) – אבל חובה להשתמש לפחות ארבעת המודולים הנ"ל תחת ה GAV  
שהוגדר כאן.

2. עליכם לעבוד עם pom aggregator שירכז את (פחות) ארבעת המודולים הנ"ל וניתן יהיה לבנות את כל הפרויקט דרכו  
ובאמצעות.

שם הארטייפקט: [enigma-aggregator](#)

3. עליכם להגדיר chownpack שיבנה את כל המע' בתצורה של jar uber המכיל את כל התלוויות הנדרשות.  
(אפשר ומומלץ בהחלה להשתמש בplugin assembly plugin שהציג בכיתה, אם כי שווה לדעת כי יש עוד אפשרויות)  
השם שלjar הتسويי הוא [enigma-machine-ex2.jar](#) (שימוש לבאות קטנות בלבד !)

במסגרת בדיקת המע' תספקו לבודק קובץ batch פשוט שיבצע clone ל repository שלכם וירץ mvn clean install בראשיתה,  
תוך ציפיה לקבל jar uber.

אתjar זהה ניתן יהיה להפעיל באמצעות jar - java jar בצורה פשוטה.

במידה וממשתמש בKİודם בתרגיל הקודם אין הכרח לגרום להם לפעול גם בתרגיל זה, אולם אם זה מתאפשר זה יחමם את  
LIBI (למען הסר ספק, חימום LIBI אינו מתרגם להעלאת נקודות).

## [איך מתחילה?](#)

אני ממליץ לפתח פרויקט חדש ב IntelliJ (אם תרצו גם repository חדש בפייסבוק, אם כי לא חובה), כזה המבוסס על עבודה עם maven ולהציג את כל הקוד אליו.

ראשית הייתה מתחילה בלקחת ולהזכיר את האפליקציה הקיימת (תרגיל 1) במסגרת תוכנות maven. התהילך יכול העתקה פיזית של קבצים ועובדת (סימפטיות משזה) של עדכון ה package'ים וה import'ים הרלוונטיים כדי שייתאיםו לבנייה החדש.

אני ממליץ מאוד לנסוט ולהיעזר ביכולות הצלט ובעיקר בשבותו כדי לבצע את המטלות הנ"ל וכמוון לוודא את הביצוע בעצמכם

רק לאחר שכל הפרויקט עובד לכם בתוצרת maven התעסוקו עם הרחבת יכולות הלוגיות החדשה (כמוות הרוטורים ותמיינה בלחן התקנים) על השינויים השונים שניכרים בהם.

לסיום הוסיף את הפלגין אשר יודיע ליעץ uber jar ל mom הראשי וודאו כי ניתן לבנות את הפרויקט הראשיתו באמצעות mvn install clean על גבי המום הראשי בלבד והתוצר הסופי הוא uber jar.

## [סבבה, סימתי. מה ואיך להגיש?](#)

עליכם להגיש קז אשר כולל 2 קבצים:

1. קובץ `readme` אשר יידע ליעץ uber jar ל mom הראשי וודאו כי ישמש את הבודק בהרצאת והרמת הפרויקט.
  2. קובץ `bat` או `sh` אשר ישמש את הבודק בהרצאת והרמת הפרויקט.
- קובץ דוגמא הועלה לאתר הממא. [ממליץ בחום](#) להוריד אותו ורתק להחליף בו את ה repository שלהם.

קובץ `readme` יכול את הדברים הבאים:

1. הפניה ל `git repository` שלכם, אשר ממנו יבוצע ה `clone`. וודאו כי ה `repository` אינו `private` !
2. תיעוד והסביר כללי (וממצה) של המחלקות העיקריות ותפקידם.
3. הסבר על המערכת, בחירותיכם השונות במקומות שבהם היי לכם בחירה, או כל דבר נוסף העולה על דעתכם חשוב שהבודק ידע.

פרטים יבשים

## 16.2.26

תאריך הגשה:

ציוון אפשרי מקסימלי: 100

כפי תחילת עבודה: 19.1.26

כפי זמן לביצוע: 4 שבועות

משקל התרגיל: 35%

## מטרות התרגיל העיקרי

1. מימוש המע' בתצורת spring boot
2. מימוש המערכת אפליקציית client-server (באמצעות boot)

דרישות

1. בתרגיל זה נטמיע את מכונת האניגמה בתוך שרת בתצורת spring boot.

דבר זה יחייב המהה של כלל רכיבי המע' לעבודה עם spring.

כלל המע' תשב תחת ה URL הבא:

localhost:8080/enigma

עליו יתווסף שתי שמות המשאים כי שmagدرם בסעיפים הבאים.

תצורת המע' תהיה ב jar uber שנitin לבקשת ולהפעיל מידית.

2. קליינט:

המע' תיבדק באמצעות postman כקלינט אשר יגיש בקשות HTTP שונות כדי להנוט יכולות המע'.

הבקשות השונות יפעלו את המע' על היבטי השונים תוך העברת מידע וקבלת חזרה של מידע בתצורת json.

עליכם להציג לאגדות הבקשות ולמשם במלואם. הבודק הולך להשתמש ב collection זה כדי לבדוק את האפליקציה

שלכם.

במסגרת חומרה התרגיל קיבלتم Postman collection אותו תוכלו לטעון ולהשתמש.

בנוסף בידקו נספח ד' לפרטים על הבקשות השונות והערכים המוחזרים בהם.

כמו כן, למגדל הראש, סופק גם קובץ הגדרת ה API בפורמט של Open API 3.0. אפשר לחשב עליו בעל סוג של "סכמה" המתארת את ה API'ם השונים... רוצים לנחש מה אפשר לעשות עם סכמה ??

3. Controllers :

המע' כולל מספר controller'ים שדרcum יחשפו יכולותיה השונות:

/loader – אחראי על מלאכת הטינה של מכונה. (פקודה 1) נתיב: LoaderController .3.1

/config – אחראי על קבלת וקביעת תצורת מכונה קיימת. (פקודה 2-4,6 נתיב: ConfigurationController .3.2

/process – אחראי על הפעלה בפועל של מכונה קיימת. (פקודה 5) נתיב: MachineController .3.3

/history – אחראי על כל היביט ניהול ותפעול ההיסטוריה של המכונה. נתיב: HistoryController .3.4

4. Services :

עליכם לייצר service (@Service) של spring עבור כל אחד מה controller'ים שתוארו לעיל.

זה אמרור לבצע את העבודה הלוגית של המע'. קלומר אין לכתוב לוגיקה ישירות ב controller, גם אם היא " פשוטה ". ווק קוראת למטרודה אחת.

5. ארטיפיקט סופי:

עליכם לייצר ארטיפיקט סופי של spring boot שהוא למעשה השרת המכיל את כל הקוד הנדרש להרכבת האפליקציה.

שם הארטיפיקט הסופי: enigma-machine-server-ex3.jar

## חולקה למודולים

יש להוסיף מודול נוסף של app spring שיכיל את מעטפת השירות של spring boot .spring boot מודול זה יכיל את ה controller'ים והservice'ים השונים והוא יהיה תלי' בשאר המודולים שכבר יש לכם במע', זולת מודול ה console.

## איך מתחילה ?

אני ממליץ ראשית לעبور על המע' לבנות אותה כ spring application .spring application זה יקל عليכם מאוד את השילוב והטמעה אחר כך בתוך .spring boot application

התהיל'יך אומר לזרזות מי הם ה bean'ים העיקריים במע' ולהוסיף את האנטציטות המתאימות מעלייהם. אם אין לכם כליה עדין, אני ממליץ ליצור מחלקות (כלומר beans) שהן סוג של managers מהקלים השונים של המע' (למשל: loaderManager שאחראי על טיענת המע'; MachineManager שאחראי על החזקת המכונה הנוכחיית ומרכז את כל יכולותיה; HistoryManager וכו').

לאחר שתתנסתים בכך (אם אמ' לא תצליח להגיע ל 100% מע' spring) – צרו מודול חדש עבור app spring והוא יוכל כבר להתmeshק ביטר קלות עם ה managers שהכנתם בתרגיל הקודם (או'ן יכולים להיות פשוט פשוט @Service) אם וככל שהוא מסתדר לכם

## סבירו, סימתי. מה ואיך להגיש ?

עליכם להגיש zip אשר כולל 2 קבצים:

1. קובץ readme אשר י解释 קובץ קובץ.zip
  2. קובץ bat.chun אשר ישמש את הבודק בהרצה והרמת הפרוייקט.
- קובץ דוגמא הועלה לאתר הממא. ממליץ בחום להוריד אותו ורוק להחליף בו את ה repository git שלהם.

קובץ readme יכול את הדברים הבאים:

1. הפניה ל repository git שלכם, אותו הבודק הולך לעשות clone. וודאו כי ה repository אינו private !
2. תיעוד וסביר כללי (וממצאה) של המחלקות הראשיות ותפקידם.
3. הסבר על המערכת, בחירותיכם השונות במקומות שבהם הי' לכם בחירה, או כל דבר נוסף העולה על דעתכם חשוב שהבודק ידע.

המלצות וחידושים הנוגעים למימוש מכונת האניגמה:

1. הגדרו למcona שלכם "מצב debug" שבו היא תפלוט מידע רב ככל האפשר המתעד עברו עיבוד כל תז את כל הדרך שהוא עבר, כולל פסיעת הרוטור(ים), היכן הוא נכנס מהין הוא יצא, מה היה סטטוסו במשקף וכל הדרך חזרה וכו'. רק בדרך זו תוכלו לעקוב מבחן על הנעשה בקרים של המכונה ולהבין אם היא עבדת כהכליה.

2. תוכלו לבדוק את פעילותה המכונה באמצעות תיאור מפורט של מהלך הריצה שלה על קובץ קטן (all-small-(sanity-sanity), המכיל אלף-בית של 6 אותיות בדיק, עם 3 רוטרים.

להלן טבלה של מספר עיבודי מחרוזות המכונה הנבנית מ sanity-small, עם קוד ראשון של <1><2><3><CCC>

המחרוזות הוכנסו ממשמאליימין, תוך איפוס המכונה חזרה בראשית הקוד (CCC) לפני עיבוד כל מחרוזת:

#	מחרוזת קלט	מחרוזת פלט
1	AABBCCDDEEFF	FFCCCBEEEDAA
2	FEDCBADDEF	ADEBCFEEDA
3	FEDCBAABCDEF	ADEBCFFCBEDA
4	AFBFCFDFFFFF	FACABAEADAAA
5	AAAEEEEBBBDDDCCCFFF	FFFDDDCCEEAFEDCB

3. תוכלו לבדוק עצםם גם ע"י בדיקה שאתם יכולים להציג ולפענה מחרוזת תווים הולך ו诙זר באמצעות המכונה שלכם – קלומר התהילך הפיך לחלווטין.

4. תוכלו לבדוק את המכונה עם נתוני מכונה גדולה יותר המוגדרת באמצעות paper enigma המובאת בסוף ב'.  
 להלן טבלה של מספר עיבודי מחרוזות המכונה הנבנית מ sanity-paper-enigma, עם קוד ראשון של <1><2><3><ODX>

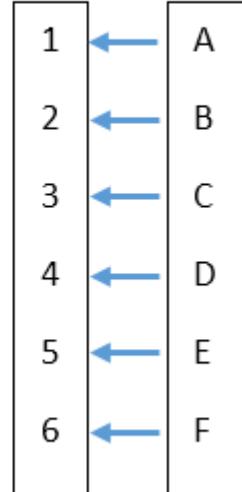
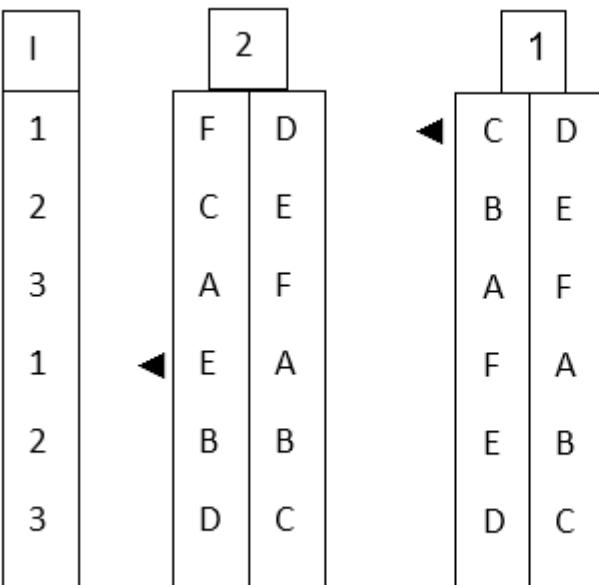
המחרוזות הוכנסו ממשמאליימין, תוך איפוס המכונה חזרה בראשית הקוד (ODX) לפני עיבוד כל מחרוזת.  
 שים לב: ב papernigma המקורי הצבת מיקומי הרוטורים נקבעת על פי העמודה השמאלית, בעוד שאצלנו במע' (ובדגימות מטה) ההצבה נקבעת דווקא על פי העמודה "הימנית" (ומבחןת ההגדרות בקובץ ה XML – על פי ה right).

#	מחרוזת קלט	מחרוזת פלט
1	THERAINISDROPPING	APZTICDXRVMWQHBHU
2	HELLOWORLD	DLTBQBVPQV
3	ENIGMAMACHINEROCKS	QMJJIDORMMMYQBVDVSBR
4	WOWCANTBELIEVEITACTUALLYWORKS	CVRDIZWDWQKUKBVHJILPKRNDXWIY
5	JAVARULES	MRUHFRRZZR

**נספח ב' – פירוט תנועת המכונה (כולל לוח תקעים)**

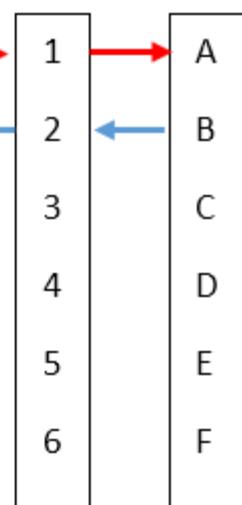
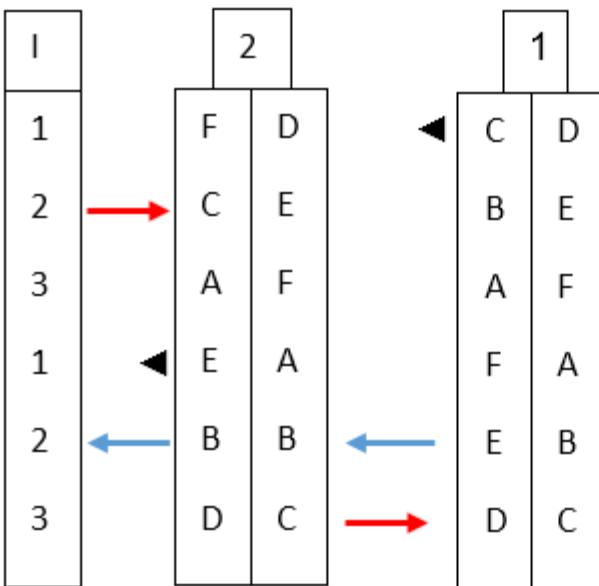
פירוט המכונה כפי שהוא בנוי מוקבץ small-sanity אך תוך שימוש בשני רוטורים בלבד (מפתח מקום) ובאמצעות הגדרת קוד ראשוני של **<B|C><A|C><2,1|CC>**

#	תיאור המצב	דיאגרמה
1	<p>מצב התחלתי בו נבחרו לשימוש רוטורים 1 (ראשון מימין) ו 2 (שני משמאל). כמו כן נבחר משקלף מס' 1.</p>	<p>Detailed description: The diagram shows four stages of two rotors.      - Stage 1: Rotor 1 has positions 1 through 6. Rotor 2 has positions A through F.      - Stage 2: Rotor 1 has positions 2 through 7. Rotor 2 has positions B through G.      - Stage 3: Rotor 1 has positions 3 through 8. Rotor 2 has positions C through H.      - Stage 4: Rotor 1 has positions 4 through 9. Rotor 2 has positions D through I.      Arrows indicate the progression from stage 1 to 2, 2 to 3, and 3 to 4. Blue arrows point from the right side of the rotors to the left side, indicating the direction of rotation."/&gt;</p>
2	<p>לאחר קביעת קוד סוד, הממקם את שני הרוטורים במקומות 3 (או ב'C') </p>	<p>Detailed description: The diagram shows four stages of two rotors.      - Stage 1: Rotor 1 has positions 1 through 6. Rotor 2 has positions A through F.      - Stage 2: Rotor 1 has positions 2 through 7. Rotor 2 has positions B through G.      - Stage 3: Rotor 1 has positions 3 through 8. Rotor 2 has positions C through H.      - Stage 4: Rotor 1 has positions 4 through 9. Rotor 2 has positions D through I.      Arrows indicate the progression from stage 1 to 2, 2 to 3, and 3 to 4. Blue arrows point from the right side of the rotors to the left side, indicating the direction of rotation. The final state is shown with a different rotor configuration compared to the first row, reflecting the key setting step mentioned in the text."/&gt;</p>



לפני כל הקלדתתו, המעל מבצעת פסיעה אחת קידימה החל מהרוטור הימני (1). במקרה הנוכחי הציג של רוטור 1 הגיע לחולנות ההצחה וכן הוא גורם גם לקידום רוטור מס' 2 צעד אחד קידימה

3



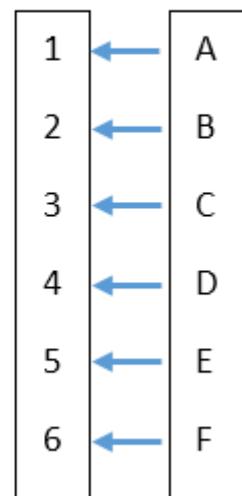
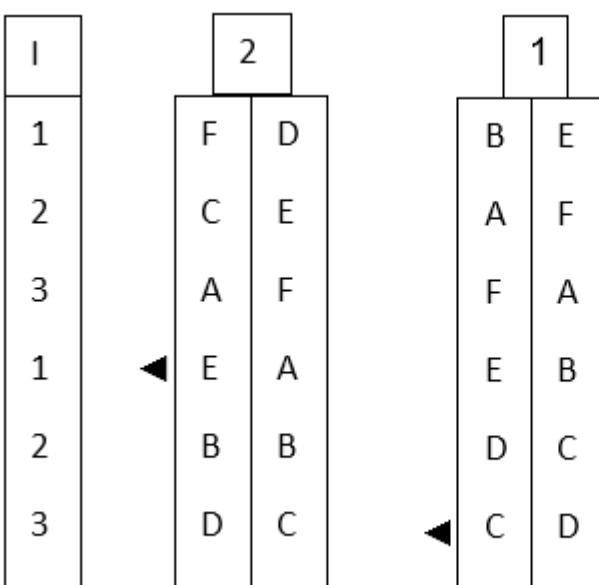
#### הצפנה האות C ← A:

הקלדת האות C פוגשת קודם כל אות לוח התקעים. יש מיקע המחליף בין C ו B ולכן הלכה מעשה הוגשה לכינית הרוטורים האות B.

האות B קודדה לאות A. תרשימים החיצים הבא מציג את כיוון הזרם בחיווט הרוטורים השונים. חץ כחול הוא כיוון ההלוך וחץ אדום הוא כיוון החזור.

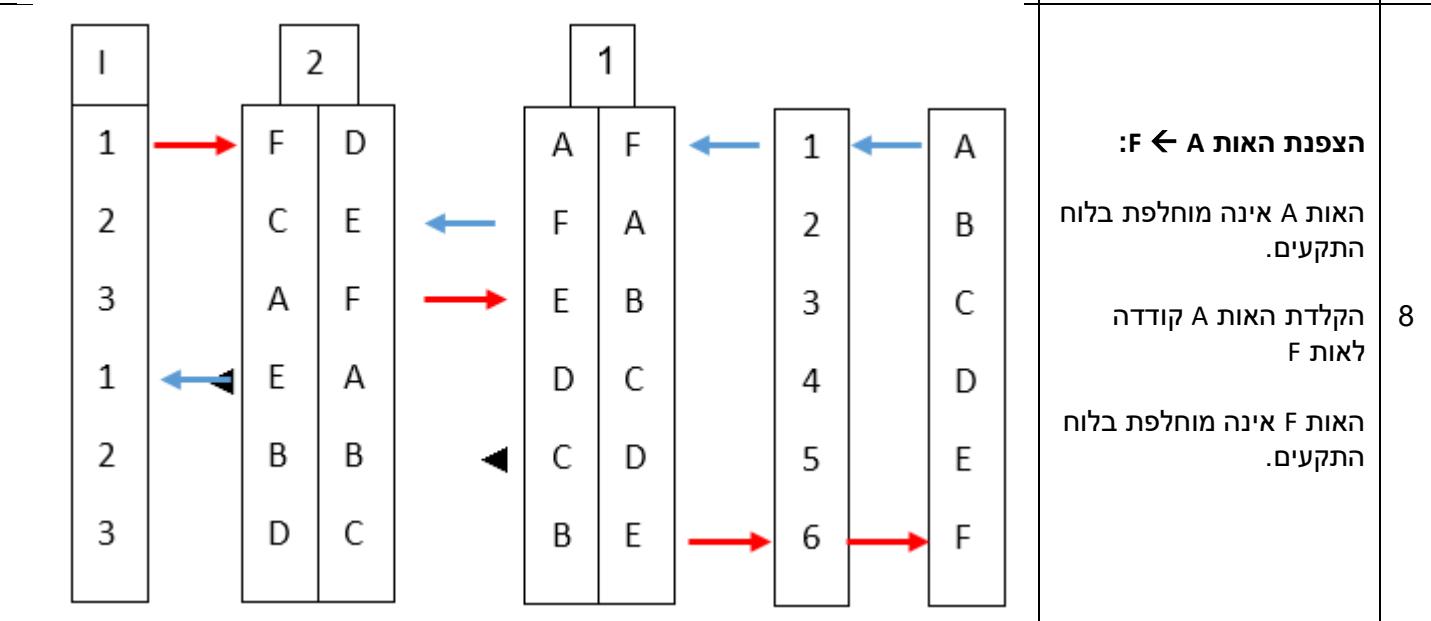
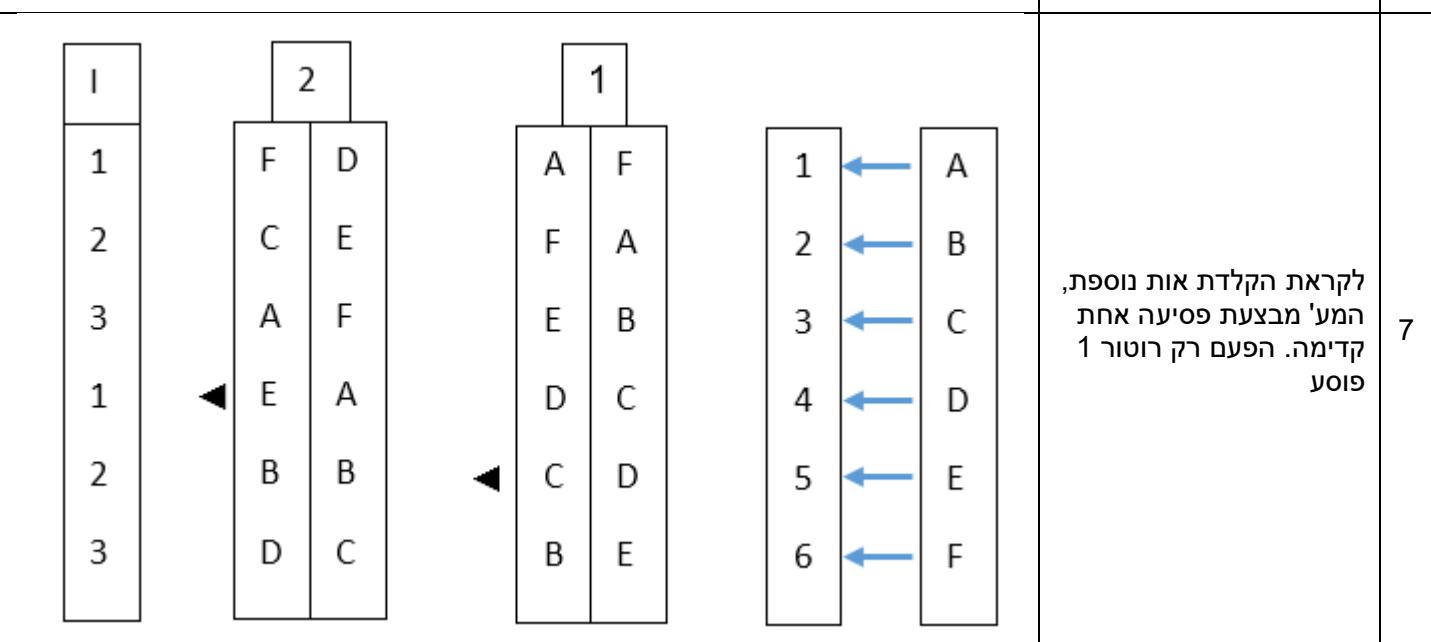
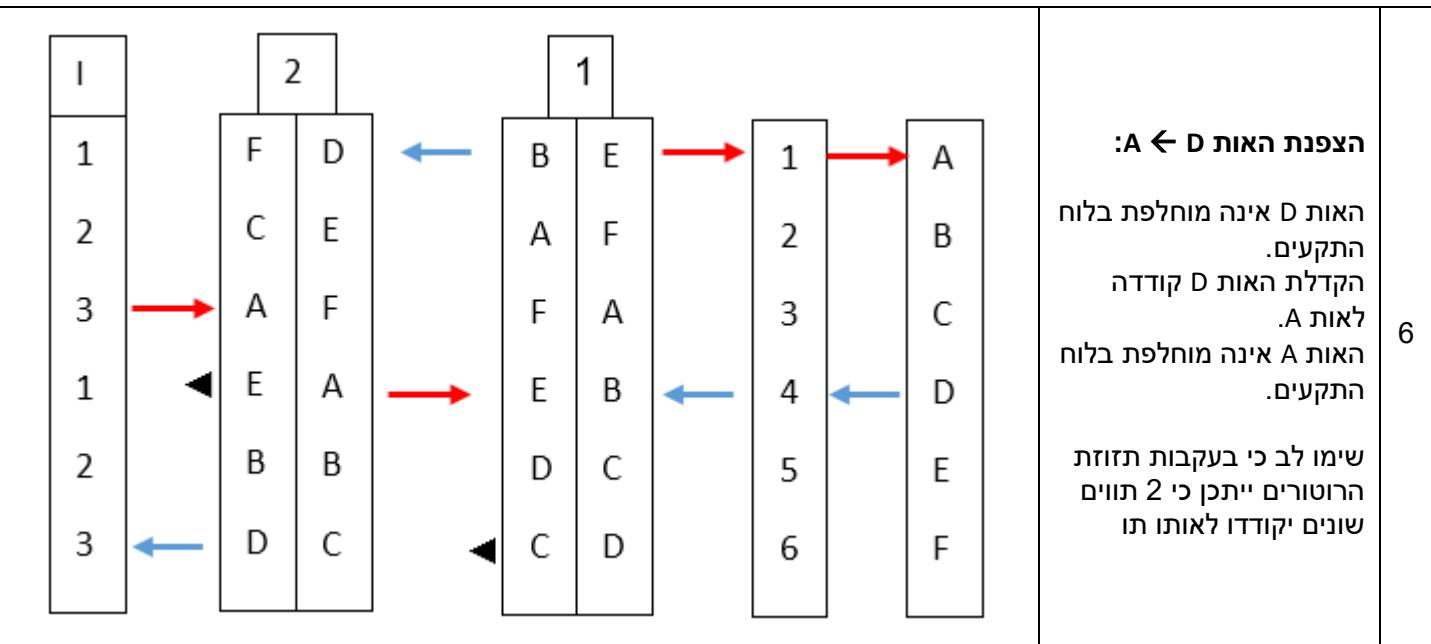
בחזור האות A ללוח התקעים היא יוצאת משם ISO AS מאחר ואין תקע מחליף לאות זו.

4



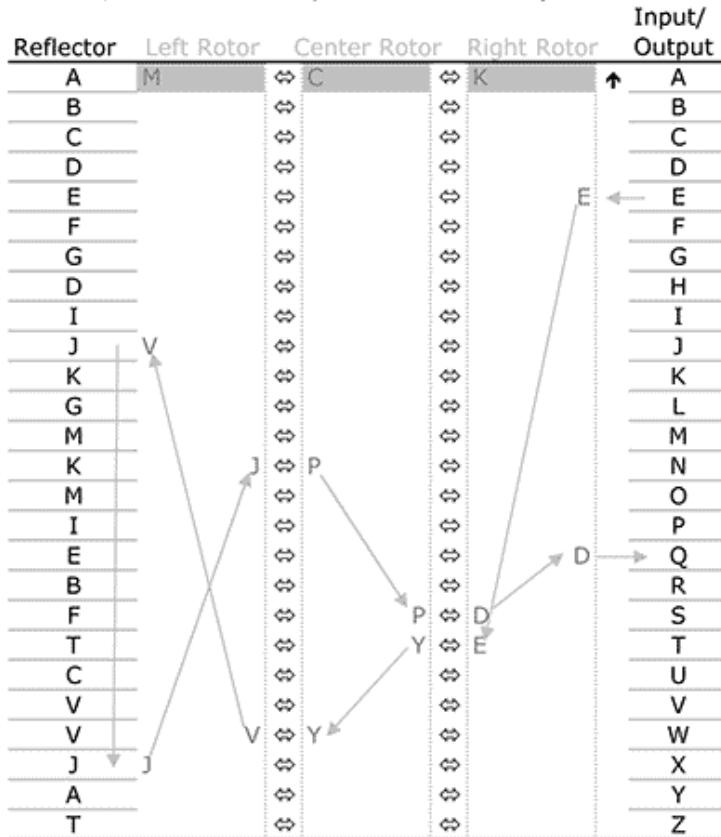
לקראת הקלדת אות נוספת, המעל מבצעת פסיעה אחת קידימה. הפעם רק רוטור 1 פועל

5



## Paper Enigma Machine

© 2003, Michael C. Koss (mike@mckoss.com)



### Setup

1. Select left/center/right rotors.
2. Position initial wheel positions by sliding the indicated window letter up to the first row.

### Operation

[Start at the input column at right, then work left to reflector, and then back to the right to the output column.]

1. If the ↑ notch appears in the window row, shift that rotor and the rotor to the left up one row (the Right Rotor is always shifted up one row before each letter is encoded/decoded).
2. Select letter to encode/decode in the Input column.
3. Read adjacent letter, X, in right hand column of the Right Rotor; select the letter X in the left hand column of the Rotor.
4. Repeat for Center Rotor.
5. Repeat for Left Rotor.
6. Read the adjacent letter, R, in the Reflector; select the other letter R in the Reflector.
7. Read adjacent letter, Y, in left hand column of the Left Rotor; select the letter Y in the right hand column of the Rotor.
8. Repeat for Center Rotor.
9. Repeat for Right Rotor.
10. Write down the adjacent letter, Z, in the output column.  
*Repeat for each letter of the message.*

Example: Initial setting: I-II-III: MCK, Letter E encodes to Q.

Sample Message: QMJIDO MZWZJFJR

Rotor I	Rotor II	Rotor III
A E	A A	A B
B K	B J	B D
C M	C D	C F
D F	D K	D H
E L	E S	E J
F G	F I	F L
G D	G R	G C
H Q	H U	H P
I V	I X	I R
J Z	J B	J T
K N	K L	K X
L T	L H	L V
M O	M W	M Z
N W	N T	N N
O Y	O M	O Y
P H	P C	P E
Q X	Q Q	Q I
R U	R G	R W
S S	S Z	S G
T P	T N	T A
U A	U P	U K
V I	V Y	V M
W B	W F	W U
X R	X V	X S
Y C	Y O	Y Q
Z J	Z E	Z O

✖ Cut here

## Enigma API - Developer Guide

In case of examples, these are just examples and are not meant to make sense or logic.

### Load Machine from XML

#### /enigma/load

Uploads an XML file containing the configuration of the Enigma machine. The server parses and loads this configuration for subsequent operations.

#### Request Details

- Method: POST
- URL: <http://localhost:8080/enigma/load>
- Body: Multipart form-data with one field named 'file' (the XML file to upload)

#### Successful Response

Returns a JSON object indicating whether the operation succeeded and, if not, what error occurred.

#### Example Response

```
{  
  "success": true  
}
```

In case of error:

#### Example Response

```
{  
  "success": false,  
  "error": "reason for file failure"  
}
```

### Get Current Machine Status

#### /enigma/config

Retrieves the current configuration and runtime status of the Enigma machine.

#### Request Details

- Method: GET
- URL: <http://localhost:8080/enigma/config>
- Query Parameter: "verbose". If true, adds to the result the attributes originalCode & currentRotorsPosition
  - Type: Boolean.
  - Default: false

#### Successful Response

A JSON object containing machine details, original and current code configurations, and compact representations.

#### Example Response (verbose)

```
{  
  "totalRotors": 5,  
  "totalReflectors": 3,  
  "totalProcessedMessages": 12,  
  "originalCodeCompact": "<198,346,123,959><l(4),=(10),3(52),K(3)><V>",  
  "currentRotorsPositionCompact": "<198,346,123,959><l(4),=(10),3(52),K(3)><V>",  
  "originalCode": {  
    "rotors": [{
```

```

    "rotorNumber": 3,
    "rotorPosition": "A",
    "notchDistance": 4
  }],
  "reflector": "V",
  "plugs": [{ // this is an optional attribute. It does not have to appear
    "plug1": "B",
    "plug2": "D",
  }]
},
"currentRotorsPosition": {
  "rotors": [
    {
      "rotorNumber": 3,
      "rotorPosition": "A",
      "notchDistance": 4
    },
    "reflector": "V",
    "plugs": [{ // this is an optional attribute. It does not have to appear
      "plug1": "B",
      "plug2": "D",
    }]
  ],
}
}

```

## Manual Code Selection

### /enigma/config/manual

Allows the user to manually set the Enigma machine's code, including rotor selection, reflector choice, and (optional) plugboard connections.

#### Request Details

- Method: PUT
- URL: <http://localhost:8080/enigma/config/manual>
- Body: JSON object specifying rotors, reflector, and optional plugs (same as originalCode structure)

#### Successful Response

A simple string holding the code set in compact structure, like in command 2.4

e.g. "<198,346,123,959><l(4),=(10),3(52),K(3)><V>"

#### Error Response

A string describes why it fails

## Automatic Code Setup

### /enigma/config/automatic

Automatically generates a random valid Enigma configuration and applies it to the machine.

#### Request Details

- Method: PUT
- URL: <http://localhost:8080/enigma/config/automatic>

#### Successful Response

A simple string holding the code set in compact structure, like in command 2.4

e.g. "<198,346,123,959><l(4),=(10),3(52),K(3)><V>"

## [Reset to Original Code](#)

### **/enigma/config/reset**

Resets the machine configuration back to the original code as it was when first loaded.

#### [Request Details](#)

- Method: PUT
- URL: <http://localhost:8080/enigma/config/reset>

#### [Successful Response](#)

A simple string holding the code set in compact structure, like in command 2.4

e.g. "<198,346,123,959><l(4),=(10),3(52),K(3)><V>"

## [Process Input](#)

### **/enigma/process**

Processes the given input text using the Enigma machine. This simulates encryption or decryption according to the current machine state.

#### [Request Details](#)

- Method: POST
- URL: <http://localhost:8080/enigma/process?input=<TEXT>>

#### [Successful Response](#)

Returns the output text and the updated rotor positions after processing.

#### [Example Response](#)

```
{  
  "output": "QWERTY",  
  "currentRotorsPositionCompact": "<198,346,123,959><l(4),=(10),3(52),K(3)><V>"  
}
```

## [Get Machine History](#)

### **/enigma/history**

Retrieves a detailed log of processed inputs grouped by code configurations. Each key in the JSON response represents a code setup, and its value is a list of past processed inputs with duration times.

#### [Request Details](#)

- Method: GET
- URL: <http://localhost:8080/enigma/history>

#### [Successful Response](#)

Returns a JSON object where keys are code descriptions and values are arrays of input-output records.

#### [Example Response](#)

```
{  
  "<198,346,123,959><l(4),=(10),3(52),K(3)><V>": [  
    { "input": "HELLO", "output": "XQFJM", "duration": 12 },  
    { "input": "WORLD", "output": "PQAZT", "duration": 9 }  
  ]  
}
```

תיאור המכונה והתנהלותה בתרגילים השונים נתון באמצעות קובץ XML.  
במהלך הבדיקה (של שלל התרגילים), נבדק המערכת באמצעות מספר קבצים שונים, חלקם חוקיים וחלקם תקויים, במטרה לראות אם וכייד המערכת מוגיבה לשגיאות.  
בחנו היטב את קבצי הדוגמא שהועלו למע' ה Mama וידאו כי אתם מבינים את פרטייהם ומבנהם.

היכן שמצויין **case insensitive** הכוונה היא שאין חשיבות ל**case** של האותיות באנגלית. במקרה זה הערך milk זהה לערך Milk  
היכן שמצויין שהmachrozeet יכולה להכיל רווחים – מדובר הוא רק על רווחים בתוך המחרוזת. אם מופיעים רווחים בתחילת/בסוף  
יש להעתלם מהם (רמז: המטודה () trim על המחלקה String).

מבנה המכונה מאוגד תחת האלמנט BTE-Enigma ומכל בתוכו מספר אלמנטים/מאפיינים נוספים:

```
<BTE-Enigma xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <ABC>
    ABCDEF
  </ABC>
  <BTE-Rotors>
    <BTE-Rotor id="1" notch="4">
      <BTE-Positioning right="A" left="F"/>
      <BTE-Positioning right="B" left="E"/>
      <BTE-Positioning right="C" left="D"/>
      <BTE-Positioning right="D" left="C"/>
      <BTE-Positioning right="E" left="B"/>
      <BTE-Positioning right="F" left="A"/>
    </BTE-Rotor>
    <BTE-Rotor id="3" notch="6">
    <BTE-Rotor id="2" notch="1">
  </BTE-Rotors>
  <BTE-Reflectors>
    <BTE-Reflector id="I">
      <BTE-Reflect input="1" output="4"/>
      <BTE-Reflect input="2" output="5"/>
      <BTE-Reflect input="3" output="6"/>
    </BTE-Reflector>
    <BTE-Reflector id="II">
  </BTE-Reflectors>
</BTE-Enigma>
```

#	שם	סוג	מזהה
1	BTE-Enigma	Element	אלמנט זה מכיל את כל פרטי ההגדראה של המכונה
2	ABC	Element	מתאר את האב' של המכונה. אוסף אותיות אפשריות לשימוש במהלך Case insensitive. שימו לב כי ישנים מספר תווים "לא חוקיים" ב XML ולכל אחד מהם יש חלופות. בפרט: '&' = '&lt;' ו '&' = '&gt;' יש לבצע trim לmachrozeet ה ABC כדי לנוקוטה מתווים מיותרם בראשיתה ובסיומה.
3	BTE-Rotors	Element	מכיל את הגדרות הרוטוריים השונים
4	BTE-Rotor	Element	מתאר מבנה של רוטור אחד בלבד.
5	id	Attribute	מספר הזאות של הרוטור. צריך להיות ייחיד בין כל הרוטוריים השונים
6	notch	Attribute	מייקומו של זיז הדחיפה על גבי הרוטור המסויים. המספר ניתן בסיס 1
7	BTE-Positioning	Element	מגדיר מיפוי אחד בתוך רוטור. כל רוטור יורכב מסדרה של מיפויים, כגודל ה ABC של המכונה סדרת המיפויים מתארת את המבנה של הרוטור כפי שהוא מוגדר ב Paper enigma

left   right	Attribute	8
מתארים את מיקומי האותיות בסדרת המיפויים.		
כל מיפוי אינו נחשב בפני עצמו אלא אך ורק בשבותו כחלק מסדרת המיפויים המרכיבה את הרוטור.		
האותיות יכולות להופיע ב case אחר יחסית לזו המתואר ב ABC.		
מכיל את הגדרות המשקפים השונים בשימוש המכונה	BTE-Reflectors	Element 9
מתאר מבנה של משקף בודד	BTE-Reflector	Element 10
מספר מזחה של המשקף הנ"ל. המספר מוגבל ל 5 תוים בספירה רומיות: V, IV, III, II, I	Id	Attribute 11
שיקוף (מיפוי) בודד של כניסה אחת ליציאה אחת.	BTE-Reflect	Element 12
מתארים את השיקוף שיש לבצע בין כניסה מסוימת (input) ליציאה מסוימת (output). המספרים ניתנים בסיס 1.	input\output	Attribute 13
כמויות המיפויים צפויות להיות חצי מאורכו של ה ABC		

## סכמה תרגיל 2

בתרגיל 2 מצטרף אטሪביוט נוסף לקובץ – rotors-count – המגדיר את כמות הרוטורים בשימוש בפועל. זהו מספר שלם. הוא יושב על גבי האלמנט הראשי BTE-Enigma:

```
<BTE-Enigma rotors-count="2">
```

סכמה תרגיל 1

