<u>הסבר עבור קוד האסמבלי</u>

הגדרנו את רגיסטר 4 כמצביע למערך 1 ואת רגיסטר 5 כמצביע למערך 2. את רגיסטר 13 Main ב – הגדרנו כמצביע לכתובת הראשונה במערך התוצאה, ואת רגיסטר 12 כמונה האיטרציות של הלולאה החיצונית (הסבר בהמשך).

ללואה חיצונית: מעבירים את הערך שרגיסטר 4 מצביע עליו מתוך מערך 1 לרגיסטר 6, ואת הערך שרגיסטר 5 מצביע עליו מתוך מערך 2 לרגיסטר 7. מאפסים את הערך שברגיסטר 10 כדי שרגיסטר זה יספור את מספר הסיביות התואמות בין כל שני מספרים מתאימים במערכים.
מאתחלים את רגיסטר 11 כמונה האיטרציות של הלולאה הפנימית – יכיל את הערך 16 עבור מעבר על 16 סיביות בכל איבר (כל איבר תופס בזיכרון מקום של מילה – 2 בתים).

לולאה פנימית: מאפסים את רגיסטר 8 ומזיזים אריתמטית את הערך שברגיסטר 6 מקום אחד ימינה – שהייתה ברגיסטר 6. מוסיפים את ביט הנשא לרגיסטר 8 LSBביט הנשא משתנה בהתאם לספרת ה – שהייתה ברגיסטר 6. לאחר מכן, באופן דומה: מאפסים LSBוכך הוא מכיל 0 או 1 בהתאם לספרת ה – את רגיסטר 9 ומזיזים אריתמטית את הערך שברגיסטר 7 מקום אחד ימינה – ביט הנשא משתנה שהייתה ברגיסטר 7. מוסיפים את ביט הנשא לרגיסטר 9 וכך הוא מכיל 0 או LSBבהתאם לספרת ה – שהייתה ברגיסטר 1LSB.7 בהתאם לספרת ה –
משווים בין רגיסטר 8 לרגיסטר 9, ובהתאם לכך מגדילים את סכום הביטים הזהים שמתבצע ברגיסטר 10, או שממשיכים בלולאה הפנימית (או שעוברים ללולאה החיצונית הבאה בהתאם למונה האיטרציות שברגיסטר 11).

לפני כל מעבר ללולאה החיצונית הבאה מעדכנים את המצביעים ומאחסנים את הערך שנסכם ברגיסטר 10 בכתובת המתאימה במערך התוצאה. מחסרים 1 ממונה האיטרציות החיצוניות (רגיסטר 12) ובהתאם לכך מסיימים את התוכנית או שמתקדמים ללולאה החיצונית הבאה.

בכתובת 2100 ומסיימת בכתובת FLASH 2146גודל התוכנית: התוכנית מתחילה בזכרון ה – (אקסדצימלי) – גודל התוכנית הוא 46 בתים באקסדצימלי – כלומר, **70 בתים** בעשרוני.

זמן הריצה של התוכנית: התוכניות רצה 2013 מחזורים – 2013 מחזורים כפול 0.954 מיקרו שניות (זמן מחזור) שווה ל – **1920.402 מיקרו שניות**.

הסבר לקוד פייתון:

כדי לפתור את המטלה הגדרנו שני מערכי מקור המכילים את המערכים הנתונים ומשתנה שמכיל את גודלם:

```
input_arr1 = [21,80,3,49,18,81,77,13]
input_arr2 = [31,13,6,65,25,10,52,40]
SIZE = len(input_arr1)
```

הגדרנו שני מערכים ריקים שיכילו את הספרות בצורתם הבינארית ואת המערך הסופי שיכיל את מספר האיברים הזהים של כל שני איברים בהתאם לאינדקס:

```
# Initialize the arrays in binary
binary_arr1 = []
binary_arr2 = []

# Initialize the result array
Identical_indices_amount = []
```

המרנו את המערכים לצורה בינארית וחתכנו את ההתחלה שמכילה אות באנגלית ו0:

```
for i in range(SIZE):
  binary_arr1.append(bin(input_arr1[i])[2:])
  binary_arr2.append(bin(input_arr2[i])[2:])
```

הוספנו אפסים לסטרינג שיצרנו כדי שתהיה באורך 16 תווים:

```
# Extend every binary number to contain 16 bits
for i in range(SIZE):
  while len(binary_arr1[i]) < 16:
    binary_arr1[i] = '0' + binary_arr1[i]

  while len(binary_arr2[i]) < 16:
    binary_arr2[i] = '0' + binary_arr2[i]
```

הגדרנו משתנה שיסכום את כל האיברים הזהים, ובדקנו איבר איבר לפי אינדקס כדי לבדוק אם הם שווים, ובהתאם לכך סכמנו. איחסנו את התוצאה במערך.

```
sum_identical = 0

# Iterate over each number in
# the arrays and find the amount o
# f the identical bits
for i in range(SIZE):
  for j in range(16):
    if binary_arr1[i][j] == binary_arr2[i][j]:
      sum_identical += 1
  Identical_indices_amount.append(sum_identical)
  sum_identical = 0
```

הדפסנו את המערך:

```
print(Identical_indices_amount)
```