.1

א.לא כי כמות זכרון העזר שהאלגוריתם בפקודה משתמש בה תלויה בגודל הליסט(הקלט) והיא לא קבועה.(נוצר ליסט חדש שתופס מקום בזכרון בדומה לליסט הקודם)

ב.הפונקציה תחזיר None. היא פועלת על הליסט אבל לא תחזיר אותו כי זה העתק של הקלט וההעתק תלוי בזיכרון של הקלט...

ההבדל בין מחרוזת לליסט שלתווים בליסט אפשר לגשת דרך אינדקסים ולשנות אותם באותו הליסט בלי ליצור ליסט חדש- (שכמובן לא יהיה במקום). במחרוזת לעומת זאת צריך לרוץ על כל המחרוזת ולקחת תווים מהסוף עד ההתחלה וליצור מחרוזת חדשה עם התווים האלה, וכך זה לא יהיה אלגוריתם במקום כי הוא יוצר סטרינג בגודל של הקלט ומשתמש בכמות זכרון תלויה בסטרינג.

.2

א.מס' קטן ביותר של פעולות כפל = m+1 זה קורה כאשר b הוא חזקה של 2. היצוג הבינארי שלו במקרה זה הוא 1 ולאחריו (m-1) אפסים, כלומר mביטים) הלולאה תבצע m הכפלות a*a + הכפלה אחת של a *result (כש b=1)

מס' גדול ביותר של פעולות כפל = 2m זה קורה כאשר b הוא מספר שמיוצג על ידי רק "1-ים" בייצוג בינארי, לכל ביט מבוצעות 2 פעולות כפל (b אינו חזקה של 2 כמובן)

.3

א.

.1

```
n=int(math.log((3**2017),2))+1
print(n)
```

.2

```
a= str(bin(3**2017))
b0 בגלל#, Print(len(a) -2)
```

תשובה:3197

ג.

!.

14 מינימום

17 מקס'

!!.

הייצוג הבינארי של המס' הוא 1 ו 19 אפסים מימין ל 1 שזה ייצוג בעל 20 ביטים

אם המספר קטן מ 524288 צריך להוריד לפחות 1 מהיצוג הבינארי לעיל ואז המספר הבינארי ירד ל 19 ביטים אבל הדרישה גדול או שווה 524288 אז המינימום = 20 ביטים

נניח בשלילה שהמס' בעל 21 ביטים אז הכי קטן ב 21 ביטים הוא 1 ו 20 אפסים מימין שווה ל 100000 בהקסדצימלי שזה לא ייתכן לפי הדרישה ב שאלה

לכן מינימום = מקסימום = 20 ביטים

ד.

!.

בגלל k הזזות המספר N יהפוך ל N*2**K ונוסיף לו 2**0+...+2**k-1 שווה ל מה שאמיר הוסיף

שזה שווה ל 2**k -1 לפי סכום סדרה הנדסית

סך הכל יהיה שווה N*2**k +2**k -1

!!.

N+2**[logN+1] הערות:לוג בבסיס 2, והסוגריים מרובעים זה ערך רצפה.

הסבר:המספר N לא השתנה כי לא הזיזו את הספרות הבינאריות שלו שמאלה.

צריך רק להוסיף את היצוג העשרוני של הספרה 1 במיקום שמצד שמאל לספרות הבינאריות של N

הייצוג הוא 2 בחזקת(המיקום בייצוג הבינארי -1) שזה בעצם בחזקת מס' הביטים של N

שכמו שראינו בשאלה שווה ל log בסיס 2 של N + 1 , ערך רצפה.

4.

א.

קיימים אינסוף מספרים ראשוניים, ונניח שסדרה מתחילה אחד מהם, אז המספר הבא חייב להיות 1 כי זה המספר היחיד שקטן מ המס הראשוני שמחלק אותו וזה כמובן סכום המספרים הקטנים ממנו שמחלקים אותו. אחרי 1 מופיע 0 וככה יש אינסוף סדרות.

אם יש מס' שסכום המחלקים ממש שלו שווים למס' עצמו אז הסדרה תהיה אינסופית מצורה של x-x-x-x-....

ב. 268

ג.אם נוסיף את מקרה 3 כתנאי ל false אי אפשר יהיה לעצור את הלולאה שבודקת את התנאי הזה כי המעגל במקרה 3 אינו מחזורי ואינסופי ואין תנאי עצירה כמו במקרה 2 ש"היה את האיבר הזה כבר בסדרה".

5.

ג.

Lcs length 1 פועל בלולאה שעוצרת כאשר אין מחרוזת משותפת שגדולה מאורך K כלשהו שגודל מ1 עד אורך המחרוזת המשותפת הגדולה ביותר. כלומר זמן הריצה של פונקציה זו תלוי באורך הכי גדול של המחרוזת המשותפת. במדידה הראשונה האורך הכי גדול 5 ובשנייה 4000 בגלל זה ההבדל הענקי בזמן ריצה שלו. לעומת lcs length 1, lcs 2 , הוא די דומה בזמני ריצה אפילו למחרוזות משותפות גדולות ביותר שונות מאוד בגודלן כי האלגוריתם שלו מעתיק לתוך טבלה דו מימדית בגודל (אורך s1 על אורך s2) (שזהה בגודלה כי בשני המקרים המחרוזות בגודל 4000)ואז עובר על הטבלה כולה ומוצא את הערך המקסימלי.בגלל זה הזמן דומה מאוד במקרה שלו כי הטבלה זהה בגודלה.זאת אומרת זמן הריצה שלו תלוי בגודל הטבלה הדו מימדית כלומר באורך המחרוזות ולא באורך המחרוזת המשותפת הגדולה ביותר כמו ב lcs1 .