

# מבני נתונים ואלגוריתמים – 88-280-02

## תרגיל 7 – String matching

---

תאריך הגשה: 13/12/2012

הוראות הגשה:

יש להגיש את התרגיל דרך האתר – submit.cs.biu.ac.il

יש לציין בתחילת הקובץ בהערה שם ות.ז.

לתרגיל זה 2 חלקים – עליכם להגיש 2 קבצים בלבד (אחד עבור כל חלק).

עבור החלק הראשון: יש להגיש קובץ יחיד בשם targil7\_1\_c.c (למי שמגיש ב-C) או targil7\_1\_cpp.cpp (למי שמגיש ב-C++).

עבור החלק השני: יש להגיש קובץ יחיד בשם targil7\_2\_c.c (למי שמגיש ב-C) או targil7\_2\_cpp.cpp (למי שמגיש ב-C++).

**ניקוד:**

הציון המקסימלי בבדיקה האוטומטית הוא 50 נקודות עבור כל חלק, כלומר סה"כ 100.

יש גם בדיקה ידנית שבה נבדוק אם פתרתם לפי הדרישות וההגבלות שצוינו (סיבוכיות, זיכרון, דרך פיתרון וכו'). בנוסף יש להתשמש בהערות על מנת לתעד ולהסביר מה שעשיתם בקוד.

### תיאור המשימה:

עליכם לכתוב תוכנית המקבלת בשורת הפקודה מספר n ואחריו מחרוזת ACII שנקרא לה S. לדוגמא, אם לקבוץ הריצה שלכם קוראים a.out ומריצים את הפקודה:

```
>> a.out 5 aabAgFumy
```

```
S=aabAgFumy-l n=5 אז
```

התוכנית תעבור על כל תתי המחרוזות של S באורך n ותדפיס את כל תתי המחרוזות המופיעות **בדיוק פעמיים** באופן הבא מ סורקים את S משמאל לימין, אז כאשר נתקלים בתת מחרוזת שכבר ראינו פעם שכבר ראינו בעבר, יש להדפיס אותה, את המיקום שלה (האינדקס) ואת המיקום של המופע הקודם שלה ב-S. אם לא קיימת תת מחרוזת כזו, לא תדפיסו כלום.

כל הדפסה כזו תהיה בשורה נפרדת. בין המחרוזות והמספרים יהיה רווח יחיד.

דוגמא של קלט:

```
>> a.out 3 the_rain_in_spain_stays_mainly_on_the_drain
```

הפלט (כמובן ללא מה שכתוב באפור):

```
n_s 16 10 // the_rain_in_spain_s stays_mainly_on_the_drain
```

```

the 34 0 // the_rain_in_spain_stays_mainly_on_the_drain
he_ 35 1 // the_rain_in_spain_stays_mainly_on_the_drain
rai 39 4 // the_rain_in_spain_stays_mainly_on_the_drain

```

**שימו לב** שתת המחרוזות "ain" ו-"in" לא יופיעו בפלט מכיוון שהן מופיעות יותר מפעמיים!

### חלק I – מימוש באמצעות רבין קרפ

- עליכם להיעזר בפונקציית Hash של מחרוזות, שנסמן HS המקיימת שניתן לחשב את  $HS(a_0a_1 \dots a_{n-1})$  מ- $HS(a_1a_2 \dots a_n)$  בזמן  $O(1)$ .
- עליכם לממש טבלת Hash, אליה תמפו ערכי Hash של תתי מחרוזות באורך n של S. אתם רשאים לממש את טבלת ה-Hash בכל דרך שלמדתם, אך מומלץ לממש Cuckoo Hashing שהוא קל ונוח למימוש. כדאי להקצות כ-50% יותר תאים ממספר האיברים שמכניסים לטבלת ה-Hash.
- המלצה נוספת – פונקציית Hash מוצלחת היא  $HS(a_1a_2 \dots a_n) = (\sum a_i p^i \% P)$  כאשר p מספר ראשוני קטן ו-P מספר ראשוני גדול (לדוגמא,  $p = 109, P = 999997$ ). \* כיצד תחשבו את  $hs_{i+1}$  מ- $hs_i$  ב- $O(1)$ ? שימו לב שהפונקציה pow (העלאה בחזקה) לא עובדת ב- $O(1)$ !

**דרישות סיבוכיות:**

זמן (ללא ההדפסות):  $O(len(S))$ .

מקום:  $O(len(S))$ .

**הערה:** מותר להשתמש בספריות string.h, math.h ובחלקת vector ב-STL (למי שמגיש ב-C++).

### חלק II – מימוש באמצעות עצי סיפא

עליכם לבנות עץ סיפא מהמחרוזת שקיבלתם כקלט.

על מנת לחסוך לכם תכנות מיותר ומייגע, אתם יכולים להשתמש בקוד קיים של בניית עצי סיפא מהאינטרנט. אני מצרפת כאן לינק לקוד אפשרי. בכל מקרה עליכם לבדוק שהקוד נכון ובונה את העץ כמו שצריך!

למתכנתים ב-C: [http://mila.cs.technion.ac.il/~yona/suffix\\_tree](http://mila.cs.technion.ac.il/~yona/suffix_tree)

למתכנתים ב-C++:

<http://code.google.com/p/simplesuffixtree/source/browse/trunk/src/?r=15>

על מנת שהפלט של החלק הזה יהיה זהה לפלט של החלק הראשון, עליכם למיין (בסיבוכיות נורמלית...) את הפלט שלכם לפי המספר הראשון, כלומר מיקום המופע השני של תת מחרוזת מסוימת.

לדוגמא- אם אתם עוברים על המחרוזת ובודקים כל תת מחרוזת, ואתם מקבלים את התוצאה הבאה:

```
the 34 0
he_ 35 1
rai 39 4
n_s 16 10
```

אז עליכם להדפיס את תתי המחרוזות בצורה ממויינת בסדר הבא:

```
n_s 16 10
the 34 0
he_ 35 1
rai 39 4
```

### דוגמאות קלט פלט:

```
>> a.out 4 AAAAAaaaaa
AAAA 1 0
aaaa 6 5
>> a.out 5 AAAAAaaaaa
>> a.out 4 SheSellsSeaShellsOnTheSeaShore
ells 13 4
heSe 20 1
SeaS 22 8
eaSh 23 9
>>> a.out 8 123456789101112
>>> a.out 2 123456789101112
11 12 11
12 13 0
>>> a.out 3 aaaAAAaaaAAAAaa
aaA 7 1
aAA 8 2
AAA 9 3
AAa 10 4
Aaa 11 5
```