מבני נתונים ואלגוריתמים – תרגיל 8

זה תרגיל יחסית קל, אז כדאי לכם להשקיע בו.
יש להגיש את התרגיל עד 23.1.12 ב-11:55 בלילה.
הגשה תתאפשר החל מה-12.1.12.

כללי

- מי שמתכנת בשפת C יגיש את התרגיל לשם targil8c (בקובץ אחד בשם targil8.c).
- מי שמתכנת בשפת C++ יגיש את התרגיל לשם targil8cpp (בקובץ אחד בשם targil8cpp.cpp).
- מותר להשתמש בספריות string.h, math.h ובמחלקת vector שב-STL (למתכנתים ב-C++).
- אין להשתמש באף מימוש מוכן של טבלת האש.
- חובה לשרר כל הקצאת זיכרון דינאמית.
- אין להניח הנחות כלשהן על גודל הקלט.

תיאור המשימה

עליכם לכתוב תוכנית המקבלת כקלט בשורת הפקודה מספר n ואחריו מחרוזת ASCII, שנקרא לה S.
לדוגמא, אם לקובץ הריצה של התוכנית שלכם קוראים a.out ומריצים את הפקודה:

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

אז $n = 5$, S=aabAgFumy.

התוכנית תעבור על כל תתי המחרוזות של S באורך n ותדפיס את כל תתי המחרוזות המופיעות פעמיים באופן הבא: אם סורקים את S משמאל לימין, אז כאשר נתקלים בתת מחרוזת שכבר ראינו בעבר, יש להדפיס אותה, את המיקום שלה (האינדקס בו היא מופיעה) ואת המיקום של המופע הקודם שלה ב-S. כל הדפסה כזו תהייה בשורה נפרדת. בין המחרוזות והמספרים יהיה רווח אחד בדיוק.

לדוגמא, אם לקובץ הריצה שלכם קוראים a.out, שורת הפקודה הבאה:

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

תדפיס את הפלט הבא (ללא החלק האפור והשורה הריקה):

```
                0 2 4 6  10   15   20   25   30   35   40
in_ 9 6     // the_rain_in_spain_stays_mainly_on_the_drain
ain 14 5    // the_rain_in_spain_stays_mainly_on_the_drain
in_ 16 9    // the_rain_in_spain_stays_mainly_on_the_drain
n_s 17 10   // the_rain_in_spain_stays_mainly_on_the_drain
ain 25 14   // the_rain_in_spain_stays_mainly_on_the_drain
the 0 34    // the_rain_in_spain_stays_mainly_on_the_drain
he_ 1 35    // the_rain_in_spain_stays_mainly_on_the_drain
rai 39 4    // the_rain_in_spain_stays_mainly_on_the_drain
ain 40 25   // the_rain_in_spain_stays_mainly_on_the_drain
```

דוגמאות קלט פלט נוספות נמצאות בהמשך.

עליכם לממש את התוכנית בעזרת רעיונות דומים לאלו של אלגוריתם רבין-קרפ. כלומר:

- עליכם להיעזר בפונקציית האש של מחרוזות, שנסמן ב-HS המקיימת שניתן לחשב את $HS(a_1a_2 \dots a_n)$ מ-$HS(a_0a_1 \dots a_{n-1})$ בזמן $O(1)$.
- עליכם לממש טבלת האש, אליה תכניסו האשים של תתי מחרוזות באורך n של S. אתם רשאים לממש את טבלת ההאש בכל דרך שלמדתם, אך אנו ממליצים על האש הקוקיה (Cuckoo Hash) כי הוא קל ונוח למימוש. כדאי להקצות כ-50% יותר תאים ממספר האיברים שמכניסים לטבלת ההאש.

עליכם לעמוד בדרישות הסיבוכיות הבאות:

- סיבוכיות הזמן של האלגוריתם, ללא ההדפסות, תהייה $O(Length(S))$.
- סיבוכיות הזיכרון של האלגוריתם תהייה $O(Length(S))$.

טיפ: פונקציה HS מוצלחת היא (לדוגמא) $HS(a_1a_2 \dots a_n) = (\sum a_i p^i)\%P$ באשר p ראשוני קטן ו-P ראשוני גדול[1] (לדוגמא, $p = 109, P = 999997$). [כיצד תחשבו את hs_i מ-hs_{i+1} ב-$O(1)$? זהירות: הפונקציה pow (העלאה בחזקה) לא עובדת ב-$O(1)$.]

תזכורת: אלגוריתם רבין-קרפ

בהינתן מחרוזת לחיפוש T ותבניות $P_1, \dots, P_k$ בגודל n תווים, אלגוריתם רבין-קרפ מוצא את כל ההופעות של התבניות ב-T בזמן $O(Length(T) + k)$. האלגוריתם משתמש בפונקציית האש מיוחדת HS למחרוזות המקיימת שניתן לחשב את $HS(a_1a_2 \dots a_n)$ מ-$HS(a_0a_1 \dots a_{n-1})$ ב-$O(1)$ פעולות.

האלגוריתם בקצרה:
אם $n > Length(T)$ אין פיתרון וסיימנו. אם לא:
בונים טבלת האש H (בגודל k) ולכל $1 \le i \le k$ מבצעים $H[HS(P_i)] = i$.
נסמן $hs_i = HS(T[i]T[i+1] \dots T[i+n-1])$.
נחשב את hs_0. אם $H[hs_0]$ מוגדר אז מצאנו התאמה ולאחר בדיקה ש-$P_{H[hs_0]}$ אכן מופיעה ב-T (במקום המצופה), נדפיס את $H[hs_0]$.
כעת, עבור $0 < i < Length(T) - n$ נבצע: חשב את hs_i מ-hs_{i-1}. אם $H[hs_i]$ מוגדר מצאנו התאמה ולאחר בדיקה ש-$P_{H[hs_i]}$ אכן מופיעה ב-T, נדפיס את $H[hs_i]$.

דוגמאות קלט ופלט

אם לתוכנית שלכם קוראים a.out היא תחזיר את הפלטים הבאים עבור הקלטים הנתונים:

```
>>> a.out 4 AAAAAaaaaa

AAAA 1 0
aaaa 6 5

>>> a.out 5 AAAAAaaaaa

>>> a.out 4 SheSellsSeaShellsOnTheSeaShore
```

[1] אבל עדיף שיתקיים $p^n < P$.

```
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 6 0
AAA 9 3
aaa 12 6
```

תפורסמנה דוגמאות נוספות מהבדיקה האוטומטית בימים הקרובים.