טעויות נפוצות בחישוב האש מתגלגל

בתרגיל בית 8 המלצנו לכם על פונקציית ההאש המתגלגל הבאה: $HS(a_1a_2 \dots a_n) = \left(\sum a_i p^i\right)\%P$ באשר p ראשוני קטן ו-P ראשוני גדול (לדוגמא, $p = 109, P = 999997$).

נסביר כמה טעויות נפוצות במימוש הפונקציה הזו. הטעויות נובעות מ-int overflow.

Int overflow

משתנה מטיפוס int מוגבל למספרים בגודל 2^31-1. ברגע שחורגים ממגבלה זו, מתקבלות תוצאות מוזרות[1]. לדוגמא, הפלט של התוכנית הבאה (המחשבת חזקות של 10):

```
int i;
int p=1;
     for (i=0;i<25;++i){
          printf("10^%2d = %d\n", i, p);
          p = p*10;
     }
```

יהיה:

```
10^ 0 = 1
10^ 1 = 10
10^ 2 = 100
10^ 3 = 1000
10^ 4 = 10000
10^ 5 = 100000
10^ 6 = 1000000
10^ 7 = 10000000
10^ 8 = 100000000
10^ 9 = 1000000000
10^10 = 1410065408
10^11 = 1215752192
10^12 = -727379968
10^13 = 1316134912
10^14 = 276447232
10^15 = -1530494976
10^16 = 1874919424
10^17 = 1569325056
10^18 = -1486618624
10^19 = -1981284352
10^20 = 1661992960
10^21 = -559939584
10^22 = -1304428544
10^23 = -159383552
10^24 = -1593835520
```

שגיאות מתחילות החל מ-10^10.

אז מה עושים? צריך להיזהר לא לגרום ל-int overlow, כלומר שהמספרים שמתעסקים איתם לא יחרגו מ-2^31.

[1] אפשר להסביר אותן, אך זה חורג ממסמך זה.

ואיך נחשב את $p^n \% P$ (לדוגמא)? מבצעים mod P אחרי כל הכפלה. לדוגמא:

```
int i;
int res = 1;
for(i=0;i<n;++i)
      res = (res * p) % P;
```

הקוד הזה לא יגרום ל-int overflow כל עוד $p \cdot P < 2^{31}$.

זהירות: אם תחשבו את $HS(a_1 a_2 \dots a_n) = \left(\sum a_i p^i\right) \% P$ ע"י חישוב $p^i \% P$ בנפרד עבור כל $1 \le i \le n$ באופן שתואר מקודם, זה יקח $O(n^2)$ עבודה. צריך לעשות משהו קצת חכם יותר (שלא יגרום ל- int overflow).

טיפוסי נתונים אחרים

קיימים מספר סוגי נתונים אחרים, ביניהם:

- long
- long long
- unsigned int
- unsigned long
- unsigned long long

עבור כל אחד מהסוגים האלה יקרה int overflow החל מגודל מסויים (שלעיתים משתנה ממערכת הפעלה אחת לאחרת).

עם זאת, כדאי לציין לטובה את הטיפוס unsigned int שמחזיק מספרים אי שליליים שלמים בטווח $[0, 2^{32} - 1]$. הטיפוס הזה מתנהג בדיוק כמו איבר בחוג $\mathbb{Z}_{2^{32}}$ במובן שפעולות הכפל והחיבור עליו מבוצעות מודולו 2^{32}.

לכן, אם תבחרו $P = 2^{32}$ וכל המשתנים שלכם יהיו מטיפוס unsigned int לא תצטרכו לעשות mod P אף פעם! אחרי כל חיבור חיסור או כפל שתבצעו יתבצע אוטומטית $\bmod 2^{32}$.

הערה: במחשבים ישנים unsigned int הוא משתנה של 16 ביטים ואז הנ"ל נכון עם 2^{16} במקום 2^{32}. לא סביר שיש לכם מחשב כזה.