משימה 1. רמז

דוק בראון הצליח להפוך את הדלוריאן שלו למכונת זמן. הוא כעת מציב לעצמו כמטרה בעיה גדולה אפילו יותר: תת-רצף משותף ארוך ביותר. בהינתן שני רצפים A ו-B באורכים N ו-M, הוא רוצה למצוא את הרצף הארוך ביותר (או אחד מהארוכים ביותר) C, כך שכל האיברים ב-C מופיעים גם ב-A וגם ב-B באותו הסדר (אבל לא בהכרח אחד אחרי השני) של C. הוא הצליח לכתוב תכנית די איטית שרצה לכמה ימים עד שהיא מוצאת פתרון. עם זאת, הוא צריך תשובה מהר ככל הניתן. התכנית המקורית שלו הייתה לתת לתכנית לרוץ ואז אחרי כמה ימים לשלוח את הפלט של התכנית חזרה בזמן לעצמו בהווה. הבעיה היא שמסע בזמן צורך כמויות אדירות של אנרגיה, ולכן לשלוח את הפתרון המלא חזרה בזמן יהיה יקר מאוד.

לדוק יש תכנית חדשה, אבל הוא צריך את עזרתכם לממש אותה. הוא רוצה לשלוח רמז קצר על הפתרון מהעתיד להווה, ואז להשתמש ברמז הזה כדי לבנות מחדש פתרון אופטימלי מהרמז. שימו לב שהוא לא חייב לבנות את אותו פתרון אופטימלי בדיוק כמו מהעתיד.

כתבו תכנית `hint.cpp` המממשת שתי פונקציות: `genHint` ו-`solve`, שמשיגות את תכניתו של דוק.

פרטי מימוש

הפונקציה `genHint` צריכה להיות עם החתימה הבאה:

```
std::vector<bool> genHint(const std::vector<int>& a, const std::vector<int>& b,
    const std::vector<int>& sol);
```

פונקציה זו תיקרא רק פעם אחת ותקבל כקלט את שני הרצפים ופתרון אופטימלי. הפונקציה צריכה להחזיר רמז, שישלח לפונקציה השנייה שלכם.

הפונקציה `solve` צריכה להיות עם החתימה הבאה:

```
std::vector<int> solve(const std::vector<int>& a, const std::vector<int>& b,
    const std::vector<bool>& hint);
```

פונקציה זו תיקרא רק פעם אחת ותקבל כקלט את שני הרצפים ואת הרמז שהפונקציה האחרת החזירה. פונקציה זו צריכה להחזיר את הפתרון האופטימלי.

חוץ משתי הפונקציות האלו, התכנית שלכם יכולה להכיל כל סוג של פונקציות עזר, מחלקות, משתנים, וכו'.
עם זאת, **אסור** לתכנית להכיל פונקציית `main` והתכנית **חייבת** לעשות include לקובץ `hint.h`. **שימו לב שבמערכת הבדיקה שתי הפונקציות הללו ייקראו בהרצות שונות של התכנית שלכם, אשר רצות בתהליכים נפרדים.** זה אומר שאתם לא תוכלו לשתף איזשהו מצב גלובלי בין שתי הריצות של הפונקציות האלו.

הרצה מקומית

הורידו את הקובץ `Lgrader.cpp`, שאותו תוכלו להריץ ביחד עם התכנית שלכם ולבדוק אותה. הגריידר יקרא N ו-M, ולאחר מכן את A ו-B. אחרי זה הגריידר יקרא את אורך הפתרון האופטימלי K ואת הפתרון האופטימלי C. הגריידר ידפיס את אורך הרמז מפונקציית `hint` שלכם, ואת הפתרון ש`solve` שלכם החזיר. שימו לב, בניגוד למערכת הבדיקה הרשמית, הבודק המקומי לא מריץ את הפונקציות שלכם בתהליכים שונים.

מגבלות

$1 \le N, M \le 10^5$
$0 \le A_i, B_j < \min(N, M)$

תת משימות

תת משימה	ניקוד	N, M
1	10	$\leq 10^4$
2	90	$\leq 10^5$

הפתרון שלכם יקבל נקודות על תת משימה רק אם הוא עבר את כל הטסטים בתת המשימה.

ניקוד

הניקוד שתקבלו עבור תת משימה תלוי באורך הרמז המקסימלי L שהתכנית שלכם יצרה עבור טסט בתת המשימה הזו.
החלק של הניקוד שתקבלו עבור תת המשימה יהיה:

$$\min\left(\left(\frac{640}{L+1}\right)^{0.3}, 1\right)$$