בדיקות קורונה

בבית הספר של אדם, התפרץ גל חדש של מגיפת הקורונה. במטרה למנוע הפצה נוספת, בית הספר החליט לבדוק את כל התלמידים באמצעות בדיקות אנטיגן על דגימות רוק של התלמידים.

משום שכל המורים שכחו איך להשתמש בהן לפני זמן רב, אדם נרשם כמתנדב לעזור עם הבדיקות.

הוא קיבל בדיקות רוק מ-N תלמידים (מטעמי פרטיות, הוא מורשה לדעת רק מספרים מזהים מ-0 עד $N-1$) והמשימה שלו היא לקבוע אילו דגימות הן חיוביות. למרבה הצער, כשהבין שלבדוק את כל התלמידים זו משימה ארוכה ומשעממת בטירוף זה היה מאוחר מידי. אף על פי כן, הוא הבין שהוא יכול לבצע את תהליך הבדיקה בדרך חכמה יותר מאשר בדיקה של הדגימות אחת אחרי השניה. אם הוא יערבב תת קבוצה של הדגימות ויבדוק את התערובת הזו, הוא יגלה האם כל הדגימות בתערובת הן שליליות או האם לפחות אחת מהן חיובית. הוא יכול להשתמש בזה על מנת להוריד את מספר הבדיקות שהוא צריך לבצע!

מכיוון שיש מספיק רוק בכל דגימה, הוא יכול לבדוק דגימה כמה פעמים שירצה. יתרה מכך, כל הבדיקות מדויקות לחלוטין, אז מעולם לא יקרה מצב שבו שתי בדיקות שונות יניבו תוצאות שונות עבור אותה הדגימה.

תחת תנאים אלו, הוא ירצה לייעל את התהליך כדי להשתמש במספר הנמוך ביותר האפשרי של בדיקות. אבל, הוא עסוק בביצוע הבדיקות, אז הייעול של התהליך מוטל עליכם.

מהסטטיסטיקה המקומית, אדם הצליח להבין שההסתברות שדגימה נתונה כלשהי חיובית שווה ל-P, ושהחיוביות או השליליות של בדיקה אחת לא מושפעת מהחיוביות או השליליות של בדיקות אחרות. אולי אתם יכולים להשתמש בכך כדי לייעל את בחירת הבדיקות שהוא מבצע?

תקשורת

בעיה זו היא אינטרקטיבית.

התוכנית שלכם תורץ על מספר מקרי בדיקה. כחלק ממקרה בדיקה אחד, ולכן תוך כדי הרצה אחת של התוכנית שלכם, יהיה עליכם לפתור T תרחישים שונים. הערכים של N ושל P יהיו זהים בכל התרחישים, אבל מזהי הדגימות החיוביות של תלמידים ישתנו (בסבירות גבוהה) בכל תרחיש.

אתם יכולים לממש את הפרוטוקול המבוקש בעצמכם, או להשתמש בתבנית הנתונה לכם. אתם יכולים למצוא את התבנית במערכת ה-CMS כקובץ מצורף בשם `template.cpp`.

הפרוטוקול

תחילה, על התוכנית שלכם לקרוא שורה מה-standard input המכילה מספר שלם N, מספר ממשי P ומספר שלם T מופרדים על ידי רווחים — מספר התלמידים, ההסתברות לדגימה חיובית ומספר התרחישים.

לאחר מכן, התוכנית שלכם יכולה להדפיס שאילתות ל-standard output. כל שאילתה יכולה להיות שורה אחת המכילה `Q`, רווח, ומחרוזת s מאורך N, כאשר s_i הוא `1` אם אדם צריך להוסיף את דגימת התלמיד i לבדיקה, ו-`0` אחרת. לאחר הדפסת שורה זו, על התוכנית שלכם לבצע flush ל-standard output ואז לקרוא תו אחד בשורה אחת שיהיה `P` אם לפחות תלמיד אחד בקבוצה חיובי, ו-`N` אחרת.

התוכנית גם יכולה להדפיס את התשובה כשורה יחידה ל-standard output המכילה `A`, רווח, ומחרוזת s מאורך N, כאשר s_i הוא `1` אם דגימת התלמיד ה-i חיובית, ו-`0` אחרת. לאחר הדפסת שורה זו, על התוכנית לעשות flush ל-standard output ולאחר מכן לקרוא תו אחד בשורה אחת.

אם השורה מכילה `C`, התשובה שלכם הייתה נכונה. במקרה זה התוכנית שלכם יכולה להתחיל לבצע שאילתות לגבי התרחיש הבא, או, אם זו הייתה התשובה ה-T שלכם, לסיים את ריצתה.

אם השורה מכילה `W`, אז התשובה שלכם לא הייתה נכונה. במקרה זה על התוכנית לסיים את ריצתה באופן מיידי.

אנא שימו לב שסיום ריצת התוכנית לאחר `W` חשובה לקבלת המשוב הנכון ממערכת ה-CMS. אם התוכנית שלכם ממשיכה לרוץ, היא עלולה לקרוס או לקבל משוב חוסר הצלחה אחר כלשהו.

התבנית

אם אתם משתמשים במימוש של הפרוטוקול שנמצא ב-`template.cpp`, עליכם לממש את הפונקציה `std::vector<bool> find_positive()`. פונקציה זו תיקרא פעם אחת עבור כל תרחיש. עליה להחזיר וקטור של בוליאנים באורך N, כאשר האיבר ה-i הוא `true` אם ורק אם דגימת התלמיד ה-i חיובית.

כדי לבצע זאת, השתמשו בפונקציה `bool test_students(std::vector<bool> mask)`. פונקציה זו מבצעת בדיקה על תת קבוצה של דגימות. הארגומנט היחיד שלה הוא וקטור של בוליאניים באורך N, כאשר האיבר ה-i הוא `true` אם הדגימה ה-i צריכה להתווסף לתערובת. היא תחזיר `true` אם (ורק אם) לפחות אחת מהדגימות בתערובת חיובית.

אתם יכולים גם להשתמש במשתנים הגלובליים `N` ו-`P` המכילים את N ואת P מהסטייטמנט. אתם יכולים לבצע אתחולים כרצונכם בפונקציה `main` לאחר הקריאה הראשונה ל- `scanf`.

קלטים

קוד הבדיקה למשימה זו לא אדפטיבי, כלומר החיוביות של כל אחת מהבדיקות נקבעת לפני הרצת התוכנית שלכם. יתרה מכך, נקבע באופן בלתי תלוי עבור כל בדיקה נתונה האם היא חיובית בהסתברות של P באמצעות מחולל מספרים אקראיים הוגן.

תתי משימות וניקוד

ישנן שתי תתי משימות.

תת משימה ראשונה (10 נקודות)

- $N = 1\,000$
- $T = 1$

- $0 \leq P \leq 1$

פתרון יתקבל אם הוא עונה נכונה ומשתמש לכל היותר ב-$2 \cdot N$ שאילתות בכל מקרה בדיקה.

תת משימה שניה (90 נקודות)

- $N = 1\,000$
- $T = 300$
- $0.001 \leq P \leq 0.2$

תת משימה זו משתמשת בניקוד חלקי.

אם התשובה שלכם בתרחיש כלשהו שגויה, תקבלו אפס נקודות. אחרת, מספר הנקודות שתקבלו במקרה בדיקה נתון יקבע בהתבסס על מספר השאילתות הממוצע בכל תרחיש. באופן כללי, מספר קטן יותר של שאילתות יניב מספר גדול יותר של נקודות. נסמן ב-Q את המספר הממוצע של שאילתות שהתוכנית שלכם משתמשת בו על פני כל התרחישים, מעוגל מטה לספרה אחת אחרי הנקודה העשרונית. עבור כל מקרה בדיקה, חישבנו ערך F (ראו מטה). הניקוד במקרה בדיקה נתון יחושב לפי החוקים הבאים:

- אם $Q > 10 \cdot F$ תקבלו 0 נקודות (`wrong answer`).
- אם $F < Q \leq 10 \cdot F$, מספר הנקודות יקבע על ידי הנוסחה הבאה:

$$90 \cdot \frac{F}{F + 4 \cdot (Q - F)}$$

- אם $Q \leq F$, תקבלו את כל 90 הנקודות.

הפתרון שלכם ינוקד על מספר מקרי בדיקה עם ערכי P שונים. מספר הנקודות הכולל שתקבלו יהיה מספר הנקודות הקטן ביותר על פני כל מקרי הבדיקה (כלומר, על פני כל ההסתברויות P).

ישנם מקרי הבדיקה הבאים:

F	P
15.1	0.001
51.1	0.005256
94.9	0.011546
191.5	0.028545
246.3	0.039856
366.2	0.068648
490.3	0.104571
639.1	0.158765
731.4	0.2

מערכת הבדיקה תספק פידבק עבור כל מקרה בדיקה. הפידבק יכלול את הערך של Q של הפתרון שלכם בכל מקרה בדיקה שבו קיבלתם ניקוד שונה מאפס.

אינטרקציה לדוגמה

לפניכם אינטרקציה לדוגמה עם הגריידר. אנא שימו לב שערכי N ו-T הללו לא יכולים להופיע באף תת משימה. אל תשכחו לבצע flush לפלט לאחר כל שורה.

הקלט שלכם	הפלט שלכם
`10 0.4 2`	
	`Q 1000000000`
`P`	
	`Q 0000001000`
`P`	
	`Q 0000000001`
`P`	
	`Q 0111110110`
`N`	
	`A 1000001001`
`C`	
	`A 0000000000`
`W`	

התוכנית פתרה נכונה את התרחיש הראשון, אך לא את השני, כי הפתרון הנכון היה `1100010010` (שהתוכנית לא הייתה יכולה לדעת, כי היא לא ביצעה אף שאילתה). אפילו אם היה תרחיש אחר, התוכנית הייתה צריכה לסיים את ריצתה באופן מיידי.