

בממלכה השטוחה הייתה חומה מפוארת שבמשך מאות שנים הגנה על שטחה מפני פלישות. אבל, לפני שנה אחת בדיוק כנופיה של שודדים החליטה שהחומה הייתה מורכבת מדי ושרפה אותה עד היסוד. כעת, הנסיכה מריה תכננה לשכור בנאים מוכשרים כדי לשחזר את החומה לתפארתה המקורית. כדי לשלם לבנאים, היא הלכה לאוצר המלכותי ומצאה $N = 2^K$ מטבעות, לכל אחד מהם יש מספר ייחודי מ-0 ועד $N - 1$. רואה החשבון של חצר המלוכה יידע את המלכה שישנם שני סוגים של מטבעות – קלים וכבדים, כך שכל המטבעות הכבדים – כבדים באותה המידה, וכך גם הקלים. בנוסף, לפי רואה החשבון, יש לפחות מטבע קל אחד ולפחות מטבע כבד אחד באוצר.

לפי התנאים שסוכמו עם הבנאים של החומה החדשה, הנסיכה מריה מוכרחה לשלם להם רק במטבעות כבדים, ולפני שהעבודה יכולה להתחיל, היא חייבת (לבנאים) במקדמה של מטבע כבד אחד. הבעיה היא שרואה החשבון לא יודע כמה מהמטבעות ואילו מהמטבעות הם כבדים. למרבה המזל, יש לו מאזניים שאיתם הוא יכול להשוות את המשקל הכולל של שתי קבוצות שמכילות את אותו המספר של מטבעות. רואה החשבון מוכרח כעת לוודא שמספר המטבעות הכבדים באוצר המלכותי מספיק בשביל התשלום והוא חייב למצוא מטבע אחד כזה, אשר עמו אפשר יהיה לשלם את המקדמה. אף על פי כן, זה ייקח לו יותר מדי זמן, והנסיכה מריה רוצה שהבנייה של החומה החדשה תתחיל בהקדם האפשרי, לכן היא מבקשת את עזרתכם.

משימה

כתבו תכנית `coins`, אשר מממשת את הפונקציה `count_heavy`, שתקומפל עם תוכנית הבדיקה במערכת ובאמצעות תקשורת איתה, מוצאת את מספר המטבעות הכבדים באוצר המלכותי וגם את המספר של מטבע אחד כזה.

פרטי מימוש

הפונקציה שלכם צריכה להיות בעלת הפורמט הבא:

```
pair<int, int> count_heavy(int k)
```

היא תקבל כפרמטר את המספר K (תזכורת: מספר המטבעות הוא 2^K) ועליה להחזיר זוג של שני מספרים שלמים – מספר המטבעות הכבדים והמספר הייחודי של מטבע אחד מסוג זה.

כדי לתקשר עם תוכנית הבדיקה במערכת, נתונה לכם הפונקציה הבאה:

```
int weigh(const vector<int> &a, const vector<int> &b)
```

היא מקבלת כפרמטרים שני וקטורים המכילים את המספרים של המטבעות לשקילה. הוקטורים חייבים להכיל מספר איברים זהה ואסור להם להכיל איברים שחוזרים על עצמם. הפונקציה תחזיר ערך:

- 0, אם לשתי קבוצות המטבעות יש משקל כולל זהה;
- -1, אם המשקל הכולל של המטבעות אשר מספרם נמצא בוקטור a קטן מהמשקל הכולל של המטבעות אשר מספרם נמצא ב- b ;
- +1, אם המשקל הכולל של המטבעות אשר מספרם נמצא בוקטור a גדול מהמשקל הכולל של המטבעות אשר מספרם נמצא ב- b ;

שימו לב שהסיבוכיות של הפונקציה `weigh` היא פרופורציונלית למספר האיברים בוקטורים שסופקו כפרמטרים. אתם חייבים להגיש את הקובץ `coins.cpp` למערכת, שמכיל את הפונקציה `count_heavy`. הקובץ יכול להכיל קוד ופונקציות נוספות שהכרחיות לעבודה שלכם, אך אסור שיכיל את פונקציית ה-`main`. כמו כן, אין לקרוא מה-`standard input` או להדפיס ל-`standard output`. התכנית שלכם חייבת לעשות `include` לקובץ ה-`coins.h header` על ידי ההוראה לקדם-המעבד:

```
#include "coins.h"
```

מגבלות

$$1 \leq K \leq 20$$

ניקוד

אם, במהלך הריצה שלו, הפתרון שלכם נותן תשובה נכונה לטסט עם לא יותר מ- 10 000 קריאות לפונקציה *weigh*, יחושב מקדם לטסט זה לפי הנוסחה $coef = \min\left(1, \frac{11}{9} * \left(\frac{author+1}{you}\right)\right)$, כאשר *author* שווה למספר הקריאות לפונקציה *weigh* אשר בוצעו על ידי אחד מהפתרונות של המחברים, בעוד *yours* שווה למספר הקריאות לפונקציה *weigh* אשר בוצעו על ידי הפתרון שלכם. אחרת, *coef* יקבל את הערך 0 עבור טסט זה. **מובטח שבכל הטסטים הערך של *author* יהיה קטן ממש מ- 1 000**. כל תת-משימה מכילה מספר מסוים של קבוצות, בכל קבוצה שלושה טסטים בדיוק. כל קבוצה נשפטת **בנפרד**, ומספר הנקודות שתקבלו עבורה, הוא המכפלה של מספר הנקודות שסופקו לה והערך המינימלי של המקדם *coef* עבור טסט בקבוצה זו.

תתי משימות

מגבלות נוספות	נקודות	תת משימה
$K \leq 5$	5	1
יש רק מטבע כבד אחד או מטבע קל אחד.	10	2
יש לא יותר מ- K מטבעות כבדים או לא יותר מ- K מטבעות קלים.	20	3
$K \leq 10$	25	4
ללא מגבלות נוספות.	40	5

תקשורת לדוגמה

פונקציית המתחרה	תוכנית הבדיקה במערכת
	מבצעת את הקריאה <code>count_heavy(2)</code>
מבצעת את הקריאה <code>weigh({0, 1}, {2, 3})</code>	מחזירה את הערך -1
מבצעת את הקריאה <code>weigh({1, 2}, {0, 3})</code>	מחזירה את הערך +1
מבצעת את הקריאה <code>weigh({1}, {2})</code>	מחזירה את הערך 0
מחזירה <code>{3, 2}</code>	

בדיקה לוקאלית

לצורך בדיקה לוקאלית מסופקים לכם הקבצים `coins.h` ו-`Lgrader.cpp`. מקמו את הקובץ `coins.cpp` ואת שני הקבצים המסופקים באותה התיקייה. לאחר מכן קמפלו את שלושתם ביחד. כך, תקבלו תכנית שתבדוק את הנכונות של הפונקציה שלכם. התכנית תדרוש את רצף הנתונים הבא:

- בשורה הראשונה: מספר שלם חיובי אחד המייצג את הערך של K .
- בשורה השנייה: 2^K מספרים עם הערכים 1 או 0 – המסמנים האם המטבע המתאים הוא כבד או קל.

התכנית תספק כפלט מידע על הקריאות לפונקציה *weigh*, התשובה שהפונקציה *count_heavy* שלכם החזירה ומסר המציין אם התשובה שלכם נכונה או לא.