תכנון צעצועים

שם השאלה	ToyDesign
קובץ הקלט	בעיה אינטרקטיבית
קובץ הפלט	בעיה אינטרקטיבית
מגבלת הזמן	1 שניות
מגבלת הזכרון	256 מגהבייט

את עובדת בחברה שמתכננת צעצועים. צעצוע חדש שנוצר עובד באופן הבא: יש n סיכות, הממוספרות מ-1 עד n, הבולטות החוצה מקופסה. כמה זוגות של סיכות מחוברים עם חוטים בתוך הקופסה. (במילים אחרות, הסיכות והחוטים יוצרים גרף לא מכוון, בו הסיכות הן הצמתים והחוטים הם הקשתות.) החוטים לא חשופים מחוץ לקופסה, והדרך היחידה ללמוד משהו עליהם היא להשתמש ב**טסטר** על הסיכות: את יכולה לבחור שתי סיכות i ו-j כאשר $i \neq j$, והטסטר יגיד האם הסיכות מחוברות בתוך הקופסה, באופן ישיר או עקיף. (כלומר, הטסטר אומר האם יש מסלול בין הסיכות האלו בגרף).

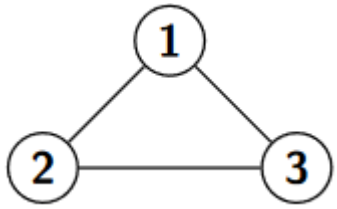
נקרא לקבוצת החיבורים בתוך הקופסה ה**תכנון** של הצעצוע.

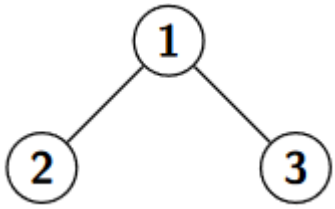
את משתמשת בתוכנה מיוחדת כדי לדגום וליצור את התכנונים הללו. התוכנה הזו עובדת בצורה הבאה: היא מתחילה עם תכנון כלשהו של הצעצוע שנסמן "תכנון 0". היא לא מראה לך את החיבורים של תכנון זה בתוך הקופסה. במקום זאת, את יכולה לבצע שוב ושוב את הפעולה הבאה בת שלושת השלבים:

1. בחרי מספר תכנון a ושני מספרי סיכות i ו-j כך ש-$i \neq j$.
2. התוכנה תספר לך מה יקרה אם נשתמש בטסטר על שתי הסיכות הללו. במילים אחרות, היא תגיד לך האם הסיכות i ו-j מחוברות (באופן ישיר או עקיף) בתכנון a.
3. בנוסף, אם הסיכות לא מחוברות ישירות או בעקיפין בתכנון a, אז התוכנה יוצרת תכנון חדש המכיל את כל החיבורים מתכנון a וחיבור ישיר חדש בין i ו-j. התכנון הזה מקבל את מספר התכנון הזמין הבא. (אז התכנון הראשון בדרך זו יקבל את המספר 1, התכנון אחר כך את המספר 2 וכו'). שימי לב לכך שזה לא משנה את תכנון a, אלא רק מוסיף תכנון חדש המכיל את החיבור הנוסף.

מטרתך היא ללמוד כמה שניתן על תכנון 0 באמצעות פעולה זו.

שימי לב שלא תמיד ניתן לקבוע את אוסף החיבורים המדויק עבור תכנון 0, משום שאין שום דרך להבדיל בין חיבורים ישירים ועקיפים. למשל, בשני התכנונים הבאים עם $n = 3$:





הטסטר ידווח שכל שתי סיכות מחוברות בשני התכנונים, אזי לא נוכל להבדיל בין התכנונים באמצעות התוכנה שמתוארת לעיל.

המטרה שלך היא למצוא תכנון כלשהו ששקול לתכנון 0. מבחינתנו שני תכנונים הם **שקולים** אם הטסטר מדווח את אותה התוצאה בשני התכנונים לכל זוג סיכות.

מימוש

זו בעיה אינטרקטיבית. עליכן לממש פונקציה

```
void ToyDesign(int n, int max_ops);
```

המוצאת תכנון *שקול* לתכנון 0. על המימוש שלך להשיג מטרה זו ע"י קריאה לשתי פונקציות כמתואר מטה. הפונקציה הראשונה שאת יכולה לקרוא לה היא:

```
int Connected(int a, int i, int j);
```

כאשר $1 \leq i, j \leq n$, $i \neq j$, $a \geq 0$, וגם אסור ש-a יעבור את מספר התכנונים שנוצרו עד כה. אם הסיכות i ו-j מחוברות (באופן ישיר או עקיף) בתכנון a, אז הפונקציה תחזיר את a בחזרה. אחרת, הפונקציה תחזיר את מספר התכנונים שנוצרו עד עתה ועוד אחד, שהופך למספר שמוקצה לתכנון החדש המכיל את החיבורים מתכנון a ובנוסף החיבור בין i ו-j. הפונקציה `Connected` יכולה להיקרא לכל היותר `max_ops` פעמים.

כאשר התוכנית שלך מסיימת עם פעולות ה-`Connected`, היא צריכה לתאר תכנון שקול לתכנון 0. כדי לתאר תכנון, התוכנית צריכה לבצע את הקריאה:

```
void DescribeDesign(std::vector<std::pair<int,int>> result);
```

הפרמטר `result` הינו וקטור של זוגות של מספרים שלמים המתארים את החיבורים הישירים בין הסיכות. כל זוג תואם לחיבור יחיד וצריך להכיל את שני המספרים של הסיכות בחיבור. חייב להיות לכל היותר חיבור ישיר אחד בין כל זוג (לא סדור) של סיכות, ואף חיבור ישיר של סיכה לעצמה. קריאה לפונקציה זו מסיימת את ריצת התוכנית שלך.

מגבלות

- $2 \leq n \leq 200$

ניקוד

- תת משימה 1 (10 נקודות): $n \leq 200$, *max_ops* = 20 000

- תת משימה 2 (20 נקודות): $n \leq 8$, *max_ops* = 20
- תת משימה 3 (35 נקודות): $n \leq 200$, *max_ops* = 2 000
- תת משימה 4 (35 נקודות): $n \leq 200$, *max_ops* = 1 350

אינטרקציה לדוגמה

פעולת המתחרה	פעולת הגריידר	הסבר
	`ToyDesign(4, 20)`	יש 4 סיכות בצעצוע. את צריכה למצוא תכנון כלשהו ששקול לתכנון 0 באמצעות לכל היותר 20 קריאות ל-`Connected`
`Connect(0, 1, 2)`	`Returns 1.`	סיכות 1, 2 לא מחוברות באופן ישיר או עקיף בתכנון 0. התכנון החדש 1 נוצר
`Connected(1, 3, 2)`	`Returns 2.`	סיכות 3 ו-2 לא מחוברות באופן ישיר או עקיף בתכנון 1, התכנון החדש 2 נוצר
`Connected(0, 3, 4)`	`Returns 0.`	סיכות 3 ו-4 מחוברות באופן ישיר או עקיף בתכנון 0, לא נוסף תכנון חדש
`DescribeDesign({{3, 4}})`	-	אנחנו מתארים תכנון המכיל חיבור יחיד, סיכות 3 ו-4

גריידר לדוגמה

הגריידר לדוגמה המסופק, `grader.cpp`, הנמצא בקובץ המצורף למשימה `ToyDesign.zip`, קורא את הקלט מ-standard input בפורמט הבא:

- השורה הראשונה מכילה את מספר הסיכות, n, מספר החיבורים הישירים, m, ואת *max_ops*.
- m השורות הבאות מכילות חיבורים ישירים בתור זוג של סיכות.

הגריידר לדוגמה קורא את הקלט וקורא לפונקציה `ToyDesign` בפתרון שלך. הגריידר ידפיס כפלט את אחת מההודעות הבאות, בהתבסס על ההתנהגות של הפתרון שלך:

- "Wrong answer: Number of operations exceeds the limit.", אם מספר הקריאות ל-`Connected` עובר את *max_ops*
- "Wrong answer: Wrong design id.", אם הפרמטר a לקריאה ל-`Connected` הוא מספר של תכנון שלא קיים ברגע הקריאה שנעשתה.
- "Wrong answer: Incorrect design.", אם התכנון שתואר דרך `DescribeDesign` לא שקול לתכנון 0.
- "OK!" אם התכנון שתואר דרך `DescribeDesign` שקול לתכנון 0.

כדי לקמפל את הגריידר לדוגמה עם הפתרון שלך, את יכולה להשתמש בפקודה הבאה בחלון הטרמינל:

```
g++ -std=gnu++11 -O2 -o solution grader.cpp solution.cpp
```

כאשר `solution.cpp` הינו הפתרון שלך שמיועד להגשה ב-CMS. כדי להריץ את התוכנית עם הגריידר לדוגמה המסופק, הקישו את הפקודה הבאה בחלון הטרמינל:

```
./solution < input.txt
```