H. 圓規 (Compass)

問題敘述

奇異果很擅長和圓形有關的問題，例如圓形交點、圓形聯集面積、扇形聯集面積等等的問題都難不倒他。

今天，奇異果在路邊發現了一個大小為 7001×7001 的棋盤，正中央的格子座標為 $(0, 0)$。一開始，所有格子都是白色的，只有一個格子 $(r, 0)$ 是黑色的。因為這個棋盤實在是太大了，只能站在棋盤外面的奇異果並不知道 r 確切是多少，只知道 $0 < r \leq 3000$。

奇異果想要畫一個以 $(0, 0)$ 為圓心、半徑為 r 的圓。你可能會很疑惑要怎麼在棋盤上畫圓，因此奇異果告訴了你他心目中的圓形的定義：

- 對於 $0 \leq x \leq \sqrt{r^2 - x^2}$，格子 $(x, \lfloor\sqrt{r^2 - x^2}\rfloor)$、$(-x, \lfloor\sqrt{r^2 - x^2}\rfloor)$、$(x, -\lfloor\sqrt{r^2 - x^2}\rfloor)$、$(-x, -\lfloor\sqrt{r^2 - x^2}\rfloor)$ 是黑色的。
- 對於 $0 \leq y \leq \sqrt{r^2 - y^2}$，格子 $(\lfloor\sqrt{r^2 - y^2}\rfloor, y)$、$(\lfloor\sqrt{r^2 - y^2}\rfloor, -y)$、$(-\lfloor\sqrt{r^2 - y^2}\rfloor, y)$、$(-\lfloor\sqrt{r^2 - y^2}\rfloor, -y)$ 是黑色的。
- 其他格子是白色的。

剛好他在旁邊發現一台神奇的機器，能夠幫助他達成目標。這台機器可以做的操作如下：

- `set_col(x, y, len, val)`：
  $val \in \{0, 1\}$，對於 $0 \leq k < len$，如果 $val = 1$，把 $(x, y + k)$ 變成黑色，反之變成白色。
- `set_row(x, y, len, val)`：
  $val \in \{0, 1\}$，對於 $0 \leq k < len$，如果 $val = 1$，把 $(x + k, y)$ 變成黑色，反之變成白色。
- `not_col(x, y, xout, yout, len)`：
  對於 $0 \leq k < len$，如果 $(x, y + k)$ 本來是白色的，把 $(x_{out}, y_{out} + k)$ 變成黑色，反之變成白色。
- `not_row(x, y, xout, yout, len)`：
  對於 $0 \leq k < len$，如果 $(x + k, y)$ 本來是白色的，把 $(x_{out} + k, y_{out})$ 變成黑色，反之變成白色。
- `and_col(x1, y1, x2, y2, xout, yout, len)`：
  對於 $0 \leq k < len$，如果 $(x_1, y_1 + k)$ 和 $(x_2, y_2 + k)$ 本來**都是黑色的**，把 $(x_{out}, y_{out} + k)$ 變成黑色，反之變成白色。
- `and_row(x1, y1, x2, y2, xout, yout, len)`：
  對於 $0 \leq k < len$，如果 $(x_1 + k, y_1)$ 和 $(x_2 + k, y_2)$ 本來**都是黑色的**，把 $(x_{out} + k, y_{out})$ 變成黑色，反之變成白色。
- `or_col(x1, y1, x2, y2, xout, yout, len)`：
  對於 $0 \leq k < len$，如果 $(x_1, y_1 + k)$ 和 $(x_2, y_2 + k)$ 本來**有至少一個是黑色的**，把 $(x_{out}, y_{out} + k)$ 變成黑色，反之變成白色。
- `or_row(x1, y1, x2, y2, xout, yout, len)`：
  對於 $0 \leq k < len$，如果 $(x_1 + k, y_1)$ 和 $(x_2 + k, y_2)$ 本來**有至少一個是黑色的**，把 $(x_{out} + k, y_{out})$ 變成黑色，反之變成白色。
- `xor_col(x1, y1, x2, y2, xout, yout, len)`：
  對於 $0 \leq k < len$，如果 $(x_1, y_1 + k)$ 和 $(x_2, y_2 + k)$ 本來**有恰好一個是黑色的**，把 $(x_{out}, y_{out} + k)$

變成黑色，反之變成白色。

- `xor_row(x1, y1, x2, y2, xout, yout, len)`：
  對於 $0 \le k < len$，如果 (x_1+k, y_1) 和 (x_2+k, y_2) 本來**有恰好一個是黑色的**，把 $(x_{out}+k, y_{out})$ 變成黑色，反之變成白色。

每一個操作都會先讀取需要知道的格子顏色，再把要修改的格子一起修改。因此，如果有讀取的格子也是要修改的格子，那麼會被讀取到的是執行這整個操作之前的狀態。

注意如果奇異果所做的操作，會導致這台機器必須要讀取或修改棋盤外的格子，這台機器就會爆炸。要是他做了超過 6×10^6 個操作，或是如果他做的所有操作中，len 的總和超過 10^9，這台機器也會爆炸。雖然這台機器聽起來充滿了危險，但奇異果真的很想要畫圓形。不過，奇異果累了，所以請你告訴他應該要如何操作，才能滿足 r 不管是多少，都能使得棋盤的最終結果是奇異果心中完美的圓形。

實作細節

你需要在首行加入 `#include "Compass.h"`，並完成以下函式：

```
void draw_circle();
```

你的程式可以呼叫以下函式：

```
void set_col(int x, int y, int len, int val);
void set_row(int x, int y, int len, int val);
void not_col(int x, int y, int xout, int yout, int len);
void not_row(int x, int y, int xout, int yout, int len);
void and_col(int x1, int y1, int x2, int y2, int xout, int yout, int len);
void and_row(int x1, int y1, int x2, int y2, int xout, int yout, int len);
void or_col(int x1, int y1, int x2, int y2, int xout, int yout, int len);
void or_row(int x1, int y1, int x2, int y2, int xout, int yout, int len);
void xor_col(int x1, int y1, int x2, int y2, int xout, int yout, int len);
void xor_row(int x1, int y1, int x2, int y2, int xout, int yout, int len);

void debug(int x, int y);
```

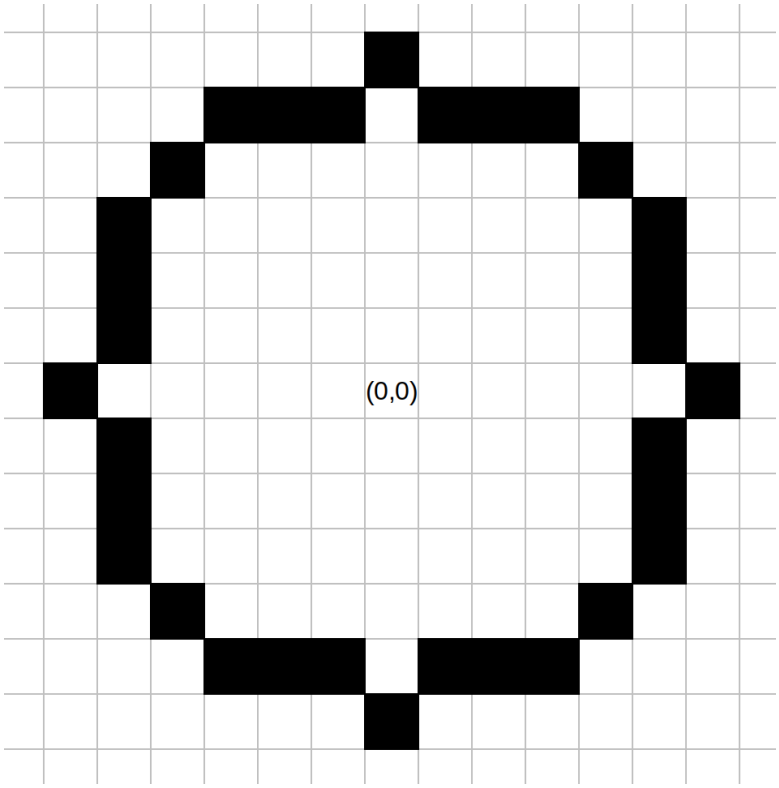
- `debug(x, y)` 會使範例評分程式輸出目前格子 (x, y) 的顏色，1 表示黑色、0 表示白色。在實際的評測過程中，此函式不會有任何效果，亦不計入操作次數。
- 其他函式的詳細功能請見問題敘述。
- `val` 必須是 `0` 或 `1`。
- `x,y,x1,y1,x2,y2,xout,yout` 必須是 $[-3500, 3500]$ 內的整數。
- `len` 必須是正整數。
- 在 `not_col,set_col,and_col,or_col,xor_col` 中，`y,y1,y2,yout` 加上 `len-1` 必須在 $[-3500, 3500]$ 內。
- 在 `not_row,set_row,and_row,or_row,xor_row` 中，`x,x1,x2,xout` 加上 `len-1` 必須在 $[-3500, 3500]$ 內。
- 你總共可以呼叫除了 `debug()` 以外的函式最多 6×10^6 次。
- 呼叫的所有函式的 `len` 總和不能超過 10^9。

互動範例

當 $r = 6$ 時，一個會畫出正確的圓形的例子如下：

```
set_col(5, 1, 3, 1);
set_col(5, -3, 3, 1);
or_row(6, 0, 5, 0, 4, 4, 1);
set_row(-3, 5, 7, 1);
and_col(0, 4, 0, 4, 0, 5, 2);
not_row(-4, 4, -4, 4, 1);
xor_col(-5, -3, 5, -3, -5, -3, 7);
xor_col(-5, 0, 6, 0, -6, 0, 1);
set_row(-3, -5, 7, 1);
not_col(0, -6, 0, -6, 2);
set_col(-4, -4, 1, 1);
set_col(4, -4, 1, 1);
```

操作後的結果如下圖：



評分說明

本題共有 3 組子任務，條件限制如下所示。每一組可有一或多筆測試資料，該組所有測試資料皆需答對才會獲得該組分數。

子任務	分數	額外輸入限制
1	5	$r \leq 1000$
2	10	$r \leq 2500$
3	85	無額外限制

在第 3 個子任務中，假設奇異果做的操作次數為 q，並且答案正確、機器也沒有爆炸，那麼你的得分為：

- 如果 $q \leq 7.5 \times 10^5$，你會得到 85 分。
- 如果 $7.5 \times 10^5 < q \leq 3.5 \times 10^6$，你會得到的分數是 $85 - \frac{55}{2.75 \times 10^6}(q - 7.5 \times 10^5)$。
- 如果 $3.5 \times 10^6 < q \leq 6 \times 10^6$，你會得到的分數是 $30 - \frac{30}{2.5 \times 10^6}(q - 3.5 \times 10^6)$。

範例評分程式

範例評分程式以下列格式讀取輸入：

- 第一列：r

r 表示圓形的半徑。

如果你的程式被評為 Accepted，範例評分程式會輸出 Accepted: q，其中 q 表示除 `debug()` 外呼叫的函式總次數。如果你的程式被評為 Wrong Answer，範例評分程式會輸出 Wrong Answer: MSG，其中 MSG 意義如下：

- invalid operation：呼叫函式的參數不符合要求，例如修改或讀取的格子超出棋盤範圍。
- too many operations：呼叫函式次數過多，或者 len 的總和超出限制。
- wrong result：最後棋盤的長相不是奇異果想要的圓形。

在附件檔案中，有一個名為「Compass.zip」的壓縮檔，下載後解壓縮可以找到三個資料夾 `cpp`、`c` 和 `examples`，資料夾的意義分別為：

- `cpp`：內部包含一個檔案 `Compass.cpp`，你可以參考或修改這份程式碼，並將修改過的程式碼上傳至評測系統。你可以用檔案 `compile_cpp.sh` 或 `compile_cpp.bat` 在自己的電腦上編譯。
- `c`：內部包含一個檔案 `Compass.c`，你可以參考或修改這份程式碼，並將修改過的程式碼上傳至評測系統。你可以用檔案 `compile_c.sh` 或 `compile_c.bat` 在自己的電腦上編譯。
- `examples`：內部包含互動範例的輸入。

請不要嘗試撰寫題目指定需要函式以外的任何東西，例如自行輸入、輸出等。`grader.cpp` 與 `grader.c` 僅供參考用，並與 Judge 上的有所落差。