MALOWANIE

Majster Maurycy wracał do domu po dniu ciężkiej pracy. A raczej próbował wracać – korek na autostradzie spowodowany kolizją kombajnu z cysterną usilnie starał się mu w tym przeszkodzić.

Silnik Poloneza wydawał z siebie miarowe, usypiające dźwięki. Krople deszczu pukały rytmicznie w blaszany dach auta. Trochę zbyt żwawy głos prezentera radiowego, zachwalającego zalety nowego środka do czyszczenia rur, przecinany był co chwila szumami i trzaskami.

Maurycy wyłączył odbiornik. Mimo ogólnej szarówki panującej na zewnątrz, ciągle piekły go oczy – no ale nie codziennie zdarza się przez dziesięć godzin malować ściany, sufit i podłogę na jadowicie zielony kolor. Doświadczony majster przyjął już w swoim życiu wiele dziwnych zleceń, ale całkowite przemalowanie hangaru lotniczego na potrzeby hodowli jakichś dziwnych roślin to była dla niego nowość. W takich chwilach poważnie zastanawiał się nad porzuceniem swojego zawodu. A może by tak zająć się swoim hobby trochę bardziej na poważnie?

Tak, to jest myśl. Maurycy od zawsze lubił gry komputerowe – polubił je tak bardzo, że naszła go ochota na zrobienie swojej własnej gry. W wolnych chwilach pomiędzy pracą a graniem w więcej gier zaczął więc uczyć się programowania. Stał się już w tym całkiem niezły. Jego umiejętności zasadniczo pozwalałyby mu na zrealizowanie jakiegoś prostego projektu – kłopot w tym, że nie przychodził mu do głowy żaden dobry pomysł.

Głośne wycie syreny strażackiej wyrwało go z rozmyślań. Popatrzył we wsteczne lusterko. Rzeczywiście, furgonetka straży pożarnej próbowała przebić się przez korek. Szybki rzut oka na drugi pas wyjaśnił wszystko - szczątki kombajnu i cysterny zajęły się ogromnym, jasnym płomieniem. Pewnie zauważyłby go wcześniej, gdyby ciągle nie miał przed oczami tej przeklętej zieleni. Maurycy zamknął oczy. Przed oczami zaczęły mu latać wielkie wałki malarskie, chlapiące pasami farby na wszystkie strony. Natychmiast otworzył oczy z powrotem. Ten dzień był już dla niego zdecydowanie za długi. W ogóle, czy przypadkiem pojazdy wolnobieżne nie mają zakazu wjazdu na autostradę?

Nagle olśnienie! Pomysł tak oczywisty, że aż dziwne, że Maurycy nie wpadł na niego wcześniej! Zamknął oczy raz jeszcze, tym razem jednak widział wszystko dokładnie. Wieloosobowa, zręcznościowa gra sieciowa, która z pewnością podbije serca graczy na całym świecie. Teraz pozostało tylko wydostać się z tego przeklętego korku . . .

Opis rozgrywki

Gracze rywalizują ze sobą w parach. Rozgrywka odbywa się na prostokątnej planszy o **n** wierszach i **m** kolumnach, podzielonej na pola 1x1. Niektóre z tych pól mogą być zablokowane. Pola w pierwszym wierszu, ostatnim wierszu, pierwszej kolumnie i ostatniej kolumnie zawsze są zablokowane. Na początku na planszy znajduje się **p** pionków pierwszego gracza oraz **p** pionków drugiego gracza. Pionki są kwadratami o boku $2 \cdot \mathbf{r} + 1$, gdzie **r** to liczba całkowita ($\mathbf{r} \geqslant 1$). Pionki nie mogą zachodzić na pola zablokowane, na inne pionki oraz muszą się w całości zawierać w planszy. Każdy pionek musi w całości zasłaniać pola planszy, na których stoi.

Każdy gracz ma swój kolor. Początkowo wolne pola na planszy nie mają koloru, lecz gracze mogą malować je na swój kolor. Pól zablokowanych nie można malować. Celem gry jest zamalowanie jak największej liczby pól swoim kolorem.

Każdy pionek również składa się z kolorowych pól. Początkowo wszystkie $(2 \cdot \mathbf{r} + 1)^2$ pól pomalowanych jest kolorem gracza, do którego należą pionki. Pionki gracza również mogą zostać pomalowane, a w wyniku malowań gracze mogą przejmować oraz tracić kontrolę nad pionkami. Dokładniej, w danej chwili pionek należy do tego gracza, który posiada więcej pól swojego koloru na powierzchni pionka.

Pionki mogą się ruszać oraz mogą strzelać farbą. Gracz zna dokładne pozycje kontrolowanych przez niego pionków, ale nie zna dokładnych pozycji pionków przeciwnika. Widzi on jednak kolory znajdujące się na planszy. Dokładniej – jeżeli pole planszy jest przykryte przez pionek przeciwnika, gracz **widzi kolor pola na pionku**. W przeciwnym przypadku (tzn. jeżeli pole jest wolne bądź zajęte przez własny pionek) widzi on kolor pola z planszy.

Do strzelania farbą potrzebna jest amunicja. Początkowo każdy pionek posiada **f** jednostek farby. Farbę można regenerować poprzez stanie na pomalowanych polach planszy, ilość farby nie może jednak przekroczyć ilości początkowej.

Dodatkowo, jeżeli pionek „ukrywa się” w polach swojego koloru (tj. jeżeli wszystkie pola pod tym pionkiem są koloru gracza kontrolującego pionek), porusza się on dwa razy szybciej.

Opis ruchów

Rozgrywka jest podzielona na tury. W trakcie tury gracz może każdemu pionkowi wydać jedną z dwóch komend:

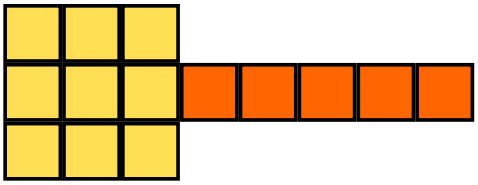
- **przesunięcie** pionka w jednym z czterech kierunków (góra, dół, lewo, prawo),
- **wystrzelenie pasa farby** w jeden z czterech kierunków, o zadanej długości.

Pod koniec tury serwer próbuje wykonać komendy wydane przez graczy. Najpierw wykonywane są wszystkie komendy ruchu – wszystkie pionki przesuwane są o jedno pole w podanym kierunku, jednocześnie. W wyniku wykonania tego ruchu mogły zdarzyć się kolizje. Rozwiązywane są one następująco:

1. wszystkie pionki, które zaczęły kolidować ze ścianą, są cofane (tzn. ich ruchy są anulowane).
2. wszystkie pionki, które zaczęły kolidować z pionkami, które nie ruszyły się w tej turze (tj. albo nie wydano im polecenia ruchu, albo ich ruch został anulowany), są cofane (jednocześnie). Po cofnięciu mogą powstać nowe kolizje tego typu (tj. z pionkami, które się nie ruszyły w tej turze) – procedurę powtarzamy zatem, aż zostaną tylko kolizje pomiędzy pionkami ruszającymi się.
3. cofamy jednocześnie wszystkie pionki, które jeszcze biorą udział w jakiejś kolizji. Jeżeli po tej operacji pozostały jeszcze jakieś kolizje, wracamy do punktu nr 2.

Następnie wykonywane są ruchy wynikające z bonusu za „ukrywanie się”. Jeżeli pionek ruszył się w tej turze (i powyższy algorytm tego nie anulował), a po wykonaniu ruchu wszystkie pola planszy przykrywane przez ten pionek są koloru gracza, wykonywany jest dodatkowy ruch w tym samym kierunku. Wszystkie ruchy „bonusowe” wykonywane są jednocześnie, tym samym algorytmem co poprzednio.

Następnie wykonywane są komendy malowania. Pas farby jest prostokątem o wysokości 1 i szerokości co najwyżej **k** (lub odwrotnie, w zależności od kierunku wystrzelenia farby). Wystrzelenie pasa o zasięgu **z** kosztuje **z** jednostek farby. Po wystrzeleniu, pas przylega do boku kwadratu, krótszym bokiem do środkowego pola. Przykładowo, na poniższym rysunku pionek o wymiarach 3x3 wystrzeliwuje pas farby o zasięgu 5 w prawo. (Pas farby jest koloru gracza – na rysunku jest on inny dla przejrzystości).



Wystrzeliwany pas jest zatrzymywany przez ścianę, tj. ostatecznie może on się stać krótszy niż **z**. (W szczególności, może on zupełnie zaniknąć, jeżeli pionek przylega do ściany i strzeli w jej kierunku). Nie zmienia to jednak kosztu wystrzelenia farby. Pas farby zmienia kolor planszy i pionków pod tym pasem. Dokładne zasady wyglądają następująco:

- jeżeli jakieś pole na planszy lub na pionku jest pokryte przez pasy obydwu kolorów, pole to nie zmienia swojego koloru.
- w przeciwnym wypadku, jeżeli jeden lub więcej pasów tego samego koloru pokrywa wolne pole na planszy (tj. niezajęte przez żaden pionek), pole to przyjmuje kolor pasa.
- jeżeli jeden lub więcej pasów tego samego koloru pokrywa pole, które jest zajęte przez pionek, to odpowiednie pole na pionku jest malowane kolorem pasa (nie ulega zmianie kolor pola planszy).

Po wykonaniu komend malowania, niektóre pionki mogły zostać przejęte (właścicielem pionka jest ten gracz, który posiada więcej pól swojego koloru na powierzchni pionka). Na samym końcu tury (już po przejęciach), wszystkim pionkom regenerowana jest amunicja. Pionek otrzymuje tyle jednostek farby, ile przykrywa pól na planszy, które są pomalowane na kolor gracza go kontrolującego. Liczba jednostek farby nie może jednak przekroczyć początkowej ilości amunicji **f**.

Opis serwera gry

Gra jest podzielona na turnieje, które z kolei dzielą się na tury. W turnieju bierze udział **d** drużyn (w tym wasza). Każdy turniej składa się z rozgrywek pomiędzy każdą parą graczy – każdy gracz gra z każdym innym graczem dokładnie raz. Wszystkie rozgrywki odbywają się jednocześnie, i we wszystkich początkowy stan planszy jest taki sam (z dokładnością do tego, który gracz kontroluje pionki).

Każdy turniej jest poprzedzony turą przerwy, w trakcie której nie odbywa się rozgrywka. Tura przerwy jest kilkukrotnie dłuższa od tury rozgrywki.

Przez następne **t** tur odbywa się rozgrywka. W trakcie tury gracz może zapytać o stan planszy – odpowiedzią są kolory pól. Kolory odpowiadają temu, co „widzi" gracz (wg zasad podanych wyżej). Ponadto, przed końcem tury może on wydać po jednym poleceniu dla każdego pionka w każdej grze. Na końcu tury, ruchy są wykonywane zgodnie z zasadami. Po upływie **t** tur turniej się kończy, graczom są przyznawane punkty, po czym rozpoczyna się następny turniej (poprzedzony turą przerwy). Liczba punktów jest proporcjonalna do liczby zamalowanych pól we wszystkich grach turnieju. Dokładniej, dla jednej rozgrywki liczba ta wynosi (zaokrąglona do liczby całkowitej):

$$\frac{\text{liczba zamalowanych pól}}{\text{liczba wolnych pól na planszy}} \cdot 1000$$

Opis planszy jest podawany w skompresowanej postaci – podane są jedynie pola, które mają inny kolor niż w poprzedniej turze. Jest też możliwość zapytania o pełny stan wszystkich pól – jest to jednak ograniczone. Można (i pewnie warto) to zrobić raz podczas tury przerwy. W przypadku zapytania w trakcie turnieju otrzymamy odpowiedź tylko wtedy, gdy od ostatniego zapytania o pełny stan planszy dla danego turnieju i danej rozgrywki oraz **od początku turnieju** minęło 30 tur. (Oznacza to w szczególności, że przez pierwsze 30 tur nie można tego zrobić). Zapytanie w trakcie tury przerwy może odpowiedzieć z pewnym opóźnieniem.

Dokładny opis komend

GET_CONSTANTS

Zwraca parametry **d**, **r**, **k** i **f** serwera (czyli kolejno: liczbę drużyn, „promień" pionka, maksymalny zasięg wystrzeliwanego pasa farby oraz maksymalna ilość amunicji posiadanej przez pionka). Parametry te nie zmieniają się pomiędzy turniejami. Przykład:

```
> GET_CONSTANTS
< OK
< 35 3 10 100
```

WAIT

Natychmiastowo zwraca OK (jak każda inna komenda), po czym zwraca dodatkowe OK na początku nowej tury (tj. tuż po przetworzeniu poprzedniej tury). Przykład:

```
> WAIT
< OK
(po pewnym czasie...)
< OK
```

GET_STATUS

Zwraca cztery liczby – 1 lub 0 (czy trwa – odpowiednio – tura rozgrywki, czy tura przerwy), liczbę tur pozostałych do końca turnieju (lub do początku turnieju, jeśli trwa przerwa – jest to wtedy zawsze 1), liczbę punktów w aktualnym turnieju (suma ze wszystkich rozgrywek) oraz przybliżony czas do końca aktualnej tury (w milisekundach). Przykład:

```
> GET_STATUS
< OK
< 1 50 15000 779
```

GET_ARENAS

Zwraca pełny opis planszy dla rozgrywek prowadzonych przez gracza (innymi słowy to, co „widzi" gracz według zasad podanych w opisie rozgrywki). W trakcie turnieju każdy gracz prowadzi dokładnie **d**-1 rozgrywek, które oznaczamy numerami od 1 do **d**-1. Polecenie przyjmuje numery rozgrywek, dla których chcemy pobrać stan planszy. Polecenie można też wywołać bez żadnych parametrów – zwróci ono wtedy opis dla wszystkich rozgrywek. W pierwszej linii zwraca liczbę zwróconych plansz – w następnych liniach znajduje się opis tych plansz. Opis jednej planszy to najpierw linia zawierające liczby **n** i **m** (wysokość i szerokość planszy), a następnie **n** linii po **m** znaków, opisujące planszę. `X` oznacza pole zablokowane, `.` oznacza wolne niepomalowane pole, `1` oznacza pole zamalowane kolorem gracza, `2` oznacza pole zamalowane kolorem przeciwnika. Polecenie można wykonać w trakcie tury przerwy – opisuje wtedy ono planszę nowego (nadchodzącego) turnieju. Polecenia nie można wykonywać za często dla jednej rozgrywki (patrz opis serwera gry). Przykład:

```
> GET_ARENAS 2 4
< OK
< 2
< 8 8
< XXXXXXXX
< X......X
< X...111X
< X......X
< X...222X
< X...222X
< X...222X
< XXXXXXXX
< 20 20
(opis planszy w drugiej rozgrywce...)
```

GET_PAWNS

Przyjmuje numery rozgrywek, dla których chcemy pobrać pozycje kontrolowanych przez nas pionków, jak w GET_ARENAS (podobnie można je wywołać bez parametrów). W pierwszej linii zwraca liczbę rozgrywek. W kolejnych liniach znajdują się opisy dla kolejnych rozgrywek. W pierwszej linii opisu znajduje się liczba pionków kontrolowanych przez nas w danej rozgrywce. Potem następują opisy pionków. W pierwszej linii opisu pionka znajdują się cztery liczby – ID pionka na planszy (unikalne dla danej rozgrywki), liczba posiadanych jednostek farby oraz numer wiersza i numer kolumny środkowego pola pionka. Potem następuje opis pokolorowania pionka – $2 \cdot \mathbf{r} + 1$ linii po $2 \cdot \mathbf{r} + 1$ znaków, gdzie 1 to kolor gracza, a 2 to kolor przeciwnika. Polecenie można wykonać co najwyżej raz na turę. Przykład (w pierwszej rozgrywce posiadamy 1 pionek, a w drugiej 3):

```
> GET_PAWNS 2 4
< OK
< 2
< 1
< 1 95 3 3
< 111
< 112
< 112
< 3
< 1 100 6 6
< 112
< 112
< 112
< 2 89 10 11
< 111
< 111
< 111
< 3 0 15 15
< 111
< 112
< 111
```

GET_DIFFS

Przyjmuje parametry tak samo, jak GET_PAWNS i GET_ARENAS. Dla każdej rozgrywki zwraca, które pola zmieniły swój kolor od poprzedniej tury (z perspektywy tego, co „widzi" gracz). W pierwszej linii zwraca liczbę rozgrywek. W kolejnych liniach znajdują się opisy dla kolejnych rozgrywek. W pierwszej linii opisu zwraca, ile pól się zmieniło. Potem dla każdego zmienionego pola zwraca dwie liczby i znak – numer wiersza pola, numer kolumny pola oraz nowy kolor pola. Polecenie można wykonać co najwyżej raz na turę. Przykład:

```
  > GET_DIFFS 2 4
< OK
< 2
< 6
< 4 5 .
< 4 6 .
< 4 7 .
< 7 5 2
< 7 6 2
< 7 7 2
< 0
```

MOVE

Wydaje pionkowi polecenie ruchu. Przyjmuje trzy liczby: numer rozgrywki, numer pionka oraz kierunek (1-4). Kierunki to kolejno – w górę, w prawo, w dół, w lewo. Jednym poleceniem można wydawać komendy wielu pionkom na raz – wystarczy w linii dopisać kolejne trójki liczb. Przykład, który porusza trzema pionkami na raz:

```
> MOVE 5 3 1 5 4 2 5 5 4
< OK
```

SHOOT

Wydaje pionkowi polecenie wystrzelenia pasa farby. Przyjmuje cztery liczby: numer rozgrywki, numer pionka, kierunek (1-4) oraz zasięg wystrzału (1-**k**). Podobnie jak powyżej, można jednym poleceniem wydawać komendy wielu pionkom. Przykład, który nakazuje wystrzał dwóm pionkom:

```
> SHOOT 5 3 1 4 5 4 2 4
< OK
```

Możliwe błędy

- 201 Tournament is not active – wydano polecenie w trakcie tury przerwy
- 202 Arena numbers must be unique – podano powtarzające się numery rozgrywek
- 203 Invalid arena number – podano numer rozgrywki spoza dobrego przedziału
- 204 Invalid direction – podano zły numer kierunku
- 205 Pawn already has an order – wydano więcej niż jedno polecenie jednemu pionkowi
- 206 Invalid shooting range – podano nieprawidłowy zasięg wystrzału farby
- 207 Not enough paint – liczba jednostek farby pionka jest mniejsza niż zasięg wystrzału
- 208 Invalid pawn id – nieprawidłowy numer pionka
- 209 Pawn not under your control – wydano polecenie pionkowi, który nie jest pod kontrolą gracza
- 210 Arena queried too recently – zapytano o stan w planszy w niedozwolonym momencie
- 211 Diff queried too recently – zapytano o różnice stanu planszy więcej niż raz w jednej turze

- 212 Pawns queried too recently – zapytano o pionki więcej niż raz w jednej turze