

Kiki i Buba

OFF 2025

Kod zadania: **Kik**
Limit pamięci: **64 MiB**



Kiki i Buba są dobrymi przyjaciółmi i gdy czasem nudziło im się na lekcji, grali w zeszyty w kółko i krzyżyk. Jednak dosyć szybko zorientowali się, że gra zawsze kończy się remisem. Postanowili zmodyfikować zasady gry, by ta była ciekawsza. W nowej grze plansza nie jest ograniczona do kratki 3x3, ale można grać na całej (nieskończonej) kartce. Dodatkowo w nowej wersji trzeba połączyć aż 5 symboli z rzędu. Nowa gra okazała się niezłym wyzwaniem, dlatego aby poćwiczyć, Buba poprosił cię o napisanie bota, z którym będzie mógł się uczyć grać.

Komunikacja

Zaimplementuj program, który zagra z Bubą w kółko i krzyżyk. Do komunikacji skorzystaj z dostarczonej biblioteki:
`#include "interface.h"`

Biblioteka udostępnia następujące funkcje:

- `void submitMove(int x, int y)` – Funkcja rysuje odpowiednio X lub O (w zależności od tego, którym graczem w danej grze jesteś) w kratce o koordynatach (x, y) .
- `pair<char, vector<pair<pair<int, int>, char>>> loadInitialState()` – Zwołaj tę funkcję raz, na początku wykonywania programu. Funkcja zwróci symbol gracza którym jesteś, listę dotychczasowych ruchów.
- `std::pair<int, int> readOpponentMove()` – Funkcja zwraca ruch przeciwnika. Ta funkcja musi zostać wywołana dokładnie raz po każdym ruchu.

Twój program nie może otwierać żadnych plików ani używać standardowego wejścia i wyjścia. Może korzystać ze standardowego wyjścia diagnostycznego (`stderr`), jednak pamiętaj, że zużywa to cenny czas.

Rozwiązanie będzie kompilowane wraz z biblioteką następującym poleceniem:

```
g++ -O3 -static -std=c++20 kib.cpp -o kib
```

Uwaga: Podane na górze ograniczenie pamięci dotyczy tylko Twojego rozwiązania, a zatem nie wlicza pamięci wykorzystywanej przez bibliotekę.

Przykłady

Poniżej możesz zobaczyć przykładową komunikację między Twoim botem a przeciwnikiem (Bubą).

Ruch	Wywołania funkcji	Opis
0	<code>loadInitialState(Game) → {'X', {{0,0}, 'X'}, {{1,0}, 'O'}}</code>	Bot wczytuje początkowy stan gry (X na pozycji (0,0) i O na pozycji (1,0))
1	<code>submitMove(0, 1)</code>	Bot zagrywa w (0,1).
2	<code>readOpponentMove() → (1, 1)</code>	Bot otrzymuje kolejny ruch Buby.
3	<code>submitMove(0, 2)</code>	Bot zagrywa w (0,2).
4	<code>readOpponentMove() → (1, 2)</code>	Buba gra dalej w tej samej kolumnie.
5	<code>submitMove(0, 3)</code>	Bot buduje linię w kolumnie 0.
6	<code>readOpponentMove() → (1, 3)</code>	Ruch przeciwnika.
7	<code>submitMove(0, 4)</code>	Bot zagrywa piąty symbol w pionie i wygrywa.



Po tej sekwencji Twój bot ma następujące ruchy: (0, 0), (0, 1), (0, 2), (0, 3), (0, 4) — tworzy więc pionową piątkę i wygrywa grę.

Buba próbował przerywać linię, ale robił to równolegle: (1, 0), (1, 1), (1, 2), (1, 3) – co było nieskuteczne.

Ocenianie

Twój bot pod koniec obozu weźmie udział w turnieju, w którym rozegra parę rozgrywek, zaczynając od wylosowanej pozycji. Punktacja będzie odwrotnie proporcjonalna do miejsca zajętego w turnieju. Dodatkowo w trakcie trwania konkursu przewidujemy turnieje testowe, na których będzie można zmierzyć się z innymi zawodnikami. Na wykonanie jednego ruchu będziesz miał 0.25s. Jeśli masz pytania dotyczące działania biblioteki lub zasad gry, możesz zgłaszać je przez sio2.

PS: Grę możesz znaleźć pod linkiem <https://tic.netlify.app/>