Malachoviacus Informaticus

Etap I, 22 marca 2014 r.

Instrukcje dla zawodników

Arkusze otwieramy na wyraźne polecenie komisji. Wszystkie poniższe instrukcje zostaną odczytane i dokładnie wyjaśnione.

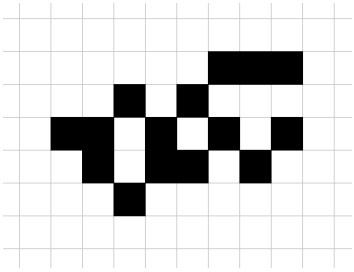
1. Arkusz składa się z *3 zadań*.
2. Każde zadanie składa się z *wprowadzenia* oraz *trzech* pytań.
3. Pytania w każdym zadaniu warte są kolejno 3, 6 i 9 punktów. Za każde zadanie można więc uzyskać maksymalnie 18 punktów.
4. Przed udzieleniem odpowiedzi na pytania przeczytaj dokładnie wprowadzenie oraz treści poleceń.
5. Swoje odpowiedzi na pytania zapisz *czytelnie* na przeznaczonych do tego arkuszach.
6. Do zapisu odpowiedzi używaj wyłącznie *długopisu lub pióra* z *czarnym lub niebieskim* tuszem. Do wykonywania rysunków możesz użyć ołówka.
7. Każdy arkusz powinien zawierać odpowiedź (lub jej część) na *tylko jedno* zadanie.
8. Przed rozpoczęciem rozwiązywania zadań na kartce z kodem zapisz wyraźnie swoje imię i nazwisko.
9. W prawym górnym rogu każdego arkusza odpowiedzi zapisz czytelnie *swój kod* oraz *numer zadania*.
10. Użycie kalkulatora jest dozwolone, jednak żadne z zadań tego nie wymaga.
11. Czas na rozwiązanie zadań wynosi *180 minut*.

Zadanie 1

Gra w życie

W 1970 roku brytyjski matematyk John Conway stworzył jeden z pierwszych przykładów *automatu komórkowego*, który nazwał „grą w życie".

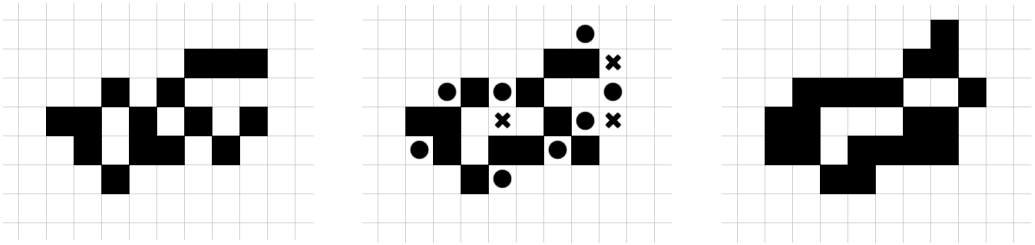
Plansza gry w życie składa się z nieskończenie dużej, dwuwymiarowej planszy, podzielonej na nieskończenie wiele kwadratowych *komórek*. W danym momencie dana komórka może być *żywa* lub *martwa*.



Rysunek 1: Przykładowa plansza gry w życie. Żywe komórki zaznaczono na czarno, a martwe na biało.

Cała gra dzieli się również na *pokolenia*, czyli kolejne układy żywych komórek na planszy. Pierwsze pokolenie możemy ustalić dowolnie, każde kolejne jest obliczane na podstawie poprzedniego według następujących zasad:

- martwa komórka, która ma dokładnie 3 żywych sąsiadów (wliczając komórki po przekątnej), w następnym pokoleniu staje się żywa, *rodzi się*,
- żywa komórka, która ma mniej niż 2 żywych sąsiadów umiera z osamotnienia,
- żywa komórka, która ma więcej niż 3 żywych sąsiadów umiera z zatłoczenia,
- w każdym innym przypadku komórka zostaje w takim samym stanie w jakim się znajdowała.

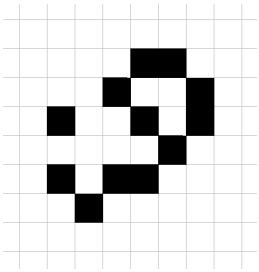


Rysunek 2: Przykład generowania kolejnego pokolenia. Po lewo pierwsze pokolenie, po prawo drugie. Rysunek pośrodku pokazuje, które komórki w następnym pokoleniu umrą (krzyżyki), a które się urodzą (kropki).

Niektóre początkowe ustawienia gry w życie mają bardzo ciekawe właściwości. Istnieją ustawienia, które po kilku pokoleniach powracają do początkowego ustawienia, inne po jakimś czasie zatrzymują się, a wiele z nich po prostu wygląda ciekawie.

PYTANIA

PYTANIE 1 Spójrz na poniższy schemat jednego pokolenia gry w życie. Narysuj kolejne pokolenie.



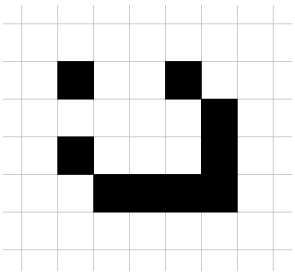
Podpowiedź: w kolejnym pokoleniu liczba żywych komórek wyniesie 14.

PYTANIE 2 Jeżeli narysujemy kilka dodatkowych pokoleń gry w życie z poprzedniego pytania, w pewnym momencie otrzymamy ten sam, wyjściowy układ. Ile potrzeba do tego pokoleń? Narysuj wszystkie pozostałe stadia pośrednie.

Podpowiedź: możesz założyć, że

- w żadnym momencie wzór nie wykroczy poza granice kwadratu 6x6, w którym znajduje się w pierwszym pokoleniu,
- wzór początkowy otrzymamy ponownie po mniej niż 10 pokoleniach,
- liczba żywych komórek zawsze będzie znajdować się między 10 a 15.

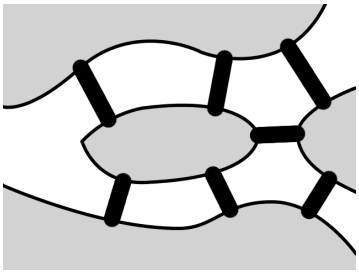
PYTANIE 3 Spójrz na poniższą planszę, która przedstawia pojedyncze pokolenie gry w życie. Narysuj 5 kolejnych pokoleń. Jakie ciekawe właściwości posiada ten wzór?



ZADANIE 2

Problem mostów Królewieckich

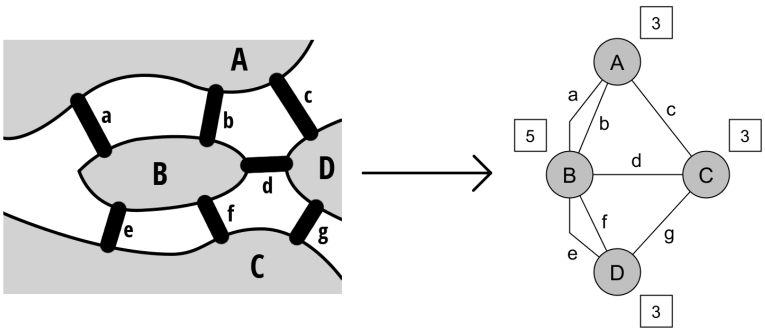
Przez Królewiec (obecnie Kaliningrad, miasto na terenie dzisiejszej Rosji) przepływa rzeka, w której rozwidleniu znajdują się dwie wyspy. Kilkaset lat temu obie strony lądu i obie wyspy połączone były siedmioma mostami, jak na rysunku poniżej.



Przez lata mieszkańcy głowili się, czy istnieje sposób przejścia po wszystkich mostach taki, żeby po każdym z nich przejść dokładnie raz. Każdy próbował, ale nikomu się nie udawało.

W końcu do miasta zawitał matematyk Leohard Euler i rozwiązał ten problem raz na zawsze, tworząc przy okazji zupełnie nowy dział matematyki, obecnie niesamowicie istotny także w informatyce – *teorię grafów*.

Graf jest to matematyczna konstrukcja składająca się z *wierzchołków* oraz łączących je *krawędzi*. Możemy na przykład przedstawić układ mostów Królewieckich w postaci następującego grafu:



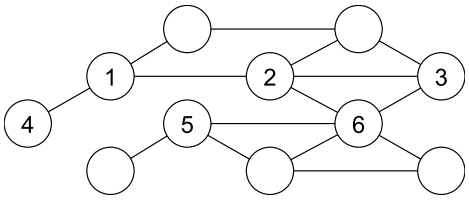
Rysunek 3: Przekształcenie mapy Królewca (po lewo) w graf (po prawo). Wierzchołki (okręgi) symbolizują brzegi rzeki oraz wyspy, natomiast krawędzie (linie) przedstawiają mosty. Oznaczenia A-D i a-g zostały dodane dla ułatwienia i nie stanowią części grafu. Numery w kwadratowych polach to stopnie wierzchołków (wyjaśnione poniżej).

Każdy wierzchołek w grafie posiada *stopień*, to jest liczbę krawędzi, które mają w nim jeden ze swoich końców. Euler udowodnił, że aby istniała w grafie ścieżka przechodząca przez każdą krawędź dokładnie raz, liczba jego wierzchołków posiadająca nieparzysty stopień musi być mniejsza lub równa 2. Na rysunku powyżej

takie wierzchołki są cztery. W związku z tym nie istnieje, tak długo poszukiwany przez mieszkańców Królewca, sposób na przejście po każdym moście dokładnie raz.

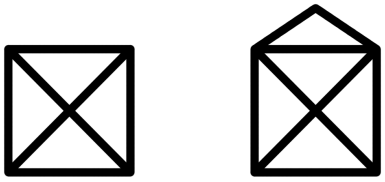
PYTANIA

PYTANIE 1 Spójrz na graf na rysunku poniżej. Jakie są stopnie ponumerowanych wierzchołków?



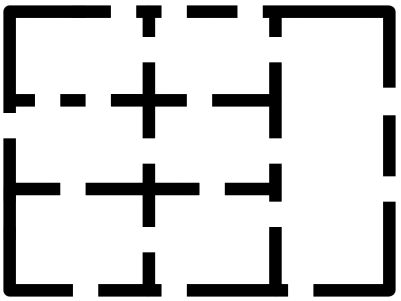
PYTANIE 2 Spróbuj narysować zamkniętą kopertę (po lewo na rysunku poniżej) bez odrywania długopisu od kartki papieru i bez rysowania tej samej linii dwukrotnie. Nie próbuj jednak zbyt długo, ponieważ nie jest to możliwe. Korzystając z wniosków wyciągniętych przez Eulera wyjaśnij dlaczego.

Zwróć uwagę, że narysowanie otwartej koperty (po prawo na rysunku poniżej) w ten sposób nie sprawia już problemu. Dlaczego?



PYTANIE 3 Na rysunku poniżej znajduje się plan pewnego budynku. Między niektórymi pokojami znajdują się drzwi. Czy da się przejść przez wszystkie drzwi tak, żeby przez każde przejść dokładnie raz? Rozpatrując odpowiedni graf uzasadnij swoją odpowiedź. Nie zapomnij o rysunku!

Zaproponuj minimalną (tj. taką, która wymaga najmniejszej liczby modyfikacji) zmianę w rozkładzie drzwi, która zmieni odpowiedź na powyższe pytanie.



Zadanie 3

Problemy przetwarzania równoległego

Obecnie komputer posiadający wiele procesorów nie jest niczym niezwykłym. Nowoczesne karty graficzne posiadają setki, czasem tysiące rdzeni. Nawet niektóre nowe modele smartfonów posiadają już wielordzeniowe procesory.

Mnożenie jednostek obliczeniowych może przyspieszyć działanie komputera, ale także sprawiać problemy, związane na przykład ze współdzieleniem zasobów.

Żeby ułatwić zrozumienie tego problemu, posłużmy się analogią. W jednym mieszkaniu żyje kilka osób. Dla upewnienia się, że w lodówce nigdy nie zabraknie mleka, przyjęto prostą strategię – za każdym razem, kiedy ktoś zauważy, że skończyło się mleko, powinien natychmiast iść do sklepu i je dokupić.

Teraz pomyślmy, co się stanie, kiedy ktoś z nich wyjdzie do sklepu po mleko, a w tym samym czasie ktoś inny zajrzy do lodówki, zauważy brak mleka i również wyruszy do sklepu. W ten sposób w lodówce znajdzie się dwa razy więcej mleka niż potrzeba, co może doprowadzić do jego zmarnowania.

Więcej niż jeden *procesor* (osoba) próbował jednocześnie wykonać pewną *procedurę* (przyjęta strategia), która wykorzystuje pewien *współdzielony zasób* (mleko). Wykonywane przez nich po kolei czynności skrzyżowały się i doprowadziły do nieporządanej sytuacji (nadmiar mleka).

Prostym rozwiązaniem jest umieszczenie, przed wyjściem do sklepu, notatki na drzwiach lodówki, mówiącej że ktoś już poszedł dokupić mleko i zdjęcie jej zaraz po powrocie do mieszkania. Osoba, która widzi brak mleka i notatkę, nie idzie do sklepu, ponieważ wie, że ktoś wyszedł już, żeby je dokupić.

Taką notatkę nazywamy *blokadą*. Informuje ona inne procesory, że dany zasób znajduje się w użytku i nikt inny nie powinien z niego korzystać. Blokadę na dany zasób w danym czasie może posiadać tylko jeden procesor. Jeżeli jakaś inna jednostka chce go użyć, musi poczekać na zwolnienie blokady i dopiero wtedy uzyskać własną. To prowadzić może jednak do zakleszczeń.

Zakleszczenie może nastąpić, kiedy dwa procesory próbują uzyskać dwie blokady na te same dwa zasoby, powiedzmy A i B. Pierwszy procesor uzyskuje blokadę zasobu A, następnie drugi procesor uzyskuje blokadę na zasób B. Teraz pierwszy czeka na zwolnienie się zasobu B, a drugi na zwolnienie się zasobu A. Żaden z nich nie zwolni blokady, więc żaden z nich nie doczeka się drugiego zasobu. Zakleszczenia mogą być trudne do wykrycia, a ich skutki bardzo nieprzyjemne.

Pytania

Pytanie 1 W pewnej restauracji, na zapleczu, znajduje się zamrażarka przeznaczona wyłącznie do przechowywania ryb. Dostęp do niej ma kilku pracowników. Każdy z nich, w momencie, gdy zauważy, że ilość pozostałej ryby spadła poniżej pewnego ustalonego limitu, ma za zadanie złożyć zamówienie na dostawę świeżej porcji ryb. Ilość zamawianej ryby jest stała i wyliczona

tak, żeby jak najbardziej wypełnić zamrażarkę. Jeżeli zamówione zostanie zbyt dużo ryby, nadmiar zmarnuje się.

Zakładając, że każdy pracownik postępuje dokładnie według zaleceń, w powyższym scenariuszu możliwe jest, że pewna ilość ryby zmarnuje się. Opisz sytuację, która może do tego doprowadzić. Zaproponuj zmianę zaleceń, która zapobiegnie tej sytuacji.

PYTANIE 2 W systemie komputerowym pewnego banku wykorzystywana jest poniższa procedura przelewu pieniędzy z jednego konta na drugie.

Przelew kwoty k z konta A na konto B

1. Sprawdź, czy stan konta A jest większy niż k. Jeżeli nie, wyświetl błąd i przerwij procedurę. W przeciwnym razie kontynuuj.
2. Zwiększ stan konta B o k.
3. Zmniejsz stan konta A o k.

Bank chce uniknąć pojawienia się ujemnych stanów konta, jednak stosowanie powyższej procedury może do tego doprowadzić. Opisz jak może do tego dojść.

Podpowiedź: rozpatrz przypadek, w którym próbujemy jednoczesnie wykonać dwa przelewy z jednego konta, na łączną sumę wyższą niż początkowy stan tego konta.

PYTANIE 3 Procedura z poprzedniego pytania została zmodyfikowana.

Przelew kwoty k z konta A na konto B

1. Uzyskaj blokadę na konto A. Zaczekaj, jeżeli jest potrzeba.
2. Uzyskaj blokadę na konto B. Zaczekaj, jeżeli jest potrzeba.
3. Sprawdź, czy stan konta A jest większy niż k. Jeżeli nie, wyświetl błąd, zwolnij blokady i przerwij procedurę. W przeciwnym razie kontynuuj.
4. Zwiększ stan konta B o k.
5. Zmniejsz stan konta A o k.
6. Zwolnij obie blokady.

Rozpatrując przypadek, w którym jeden procesor wykonuje przelew z konta X na konto Y, a drugi w tym samym czasie próbuje przelać pieniądze w drugą stronę, znajdź i opisz problem, jaki może spowodować użycie powyższej procedury.

Zaproponuj rozwiązanie tego problemu. Możesz założyć, że każde konto posiada unikalny numer, a numery te można porównywać (t.j. sprawdzić który z nich jest mniejszy). Krótko uzasadnij, dlaczego to rozwiązanie działa.