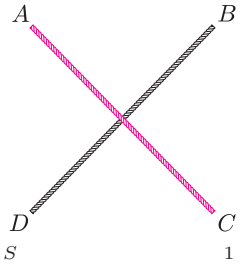
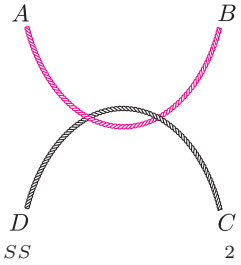
Taniec gordyjski

Piotr CHRZĄSTOWSKI

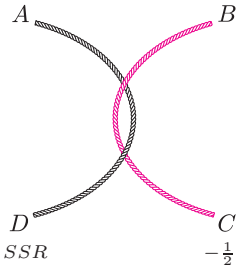
Rys. 1



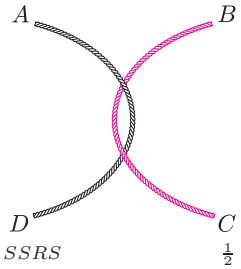
Rys. 2



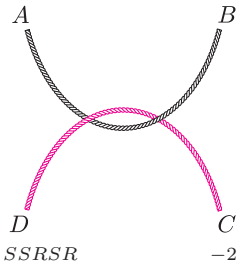
Rys. 3



Rys. 4



Rys. 5



Rys. 6

Instytut Informatyki UW od paru lat organizuje otwarte internetowe konkursy programistyczne. Do 2004 roku nazywały się one „Pogromcy Algorytmów”, w tym roku konkurs ten miał nową nazwę „Potyczki Algorytmiczne”. Zadania, z którymi mierzą się zawodnicy, są zróżnicowanej trudności, ale te z ostatniego dnia każdych zawodów są zazwyczaj dość trudne, na poziomie olimpiad informatycznych. W 2004 roku zaproponowałem zadanie, które Komitet Główny Olimpiady Informatycznej uznał za nieco zbyt trudne na olimpiadę i tak stało się ono jednym z zadań finałowej rundy „Pogromców”.

Taniec gordyjski to tradycyjny bajtocki taniec tańczony przez dwie pary tancerzy. Początkowo tancerze stoją w wierzchołkach kwadratu $ABCD$, w dwóch parach: A–B i C–D. Każda z par rozciąga między sobą sznurek. Tak więc na początku oba sznurki są rozciągnięte poziomo i równolegle do siebie (rys. 1).

Taniec składa się z ciągu ruchów, z których każdy może być ruchem następującego rodzaju:

- (S) Tancerze stojący w punktach B i C zamieniają się miejscami (nie puszczając swoich sznurków) w ten sposób, że tancerz stojący w punkcie B podnosi rękę ze sznurkiem do góry i idąc do punktu C, przepuszcza tancerza idącego z punktu C do B przed sobą, pod swoją ręką (rys. 2).
- (R) Wszyscy tancerze wykonują obrót o 90 stopni w prawo, nie puszczając sznurków, czyli tancerz, który stał w punkcie A, idzie do punktu B, ten, który stał w punkcie B, idzie do punktu C, ten, który stał w punkcie C, idzie do punktu D, a ten, który stał w punkcie D, idzie do punktu A.

W trakcie tańca sznurki plączą się ze sobą, jednak na koniec tańca powinny zostać rozplątane i znowu być rozciągnięte poziomo i równolegle do siebie. Tancerze nie muszą przy tym stać na tych samych miejscach, na których stali na początku. Taniec ten wymaga od tancerzy dużej wprawy, gdyż w trakcie tańca sznurki mogą być bardzo splątane i ciąg ruchów, który prowadziłby do ich rozplątania i rozciągnięcia poziomo i równolegle do siebie, może być trudny do odgadnięcia.

Na podstawie ciągu już wykonanych ruchów Twój program powinien wyznaczyć minimalną liczbę ruchów pozwalających zakończyć taniec.

Przykładowo, po wykonaniu ciągu ruchów SS otrzymujemy konfigurację z rysunku 3.

Najkrótszy ciąg ruchów, który pozwala zakończyć taniec, ma długość 5 i jest nim $RSRSS$ (rysunki 4–8).

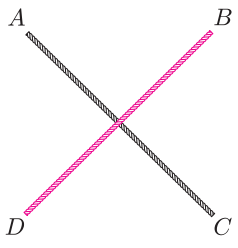
Napisz program, który:

- wczyta ze standardowego wejścia opis ciągu wykonanych ruchów w tańcu,
- wyznaczy minimalną liczbę ruchów potrzebnych do rozplątania sznurków i rozciągnięcia ich poziomo i równolegle do siebie (po wykonaniu tych ruchów tancerze nie muszą znajdować się na swoich początkowych pozycjach),
- wypisze wynik na standardowe wyjście.

W pierwszym wierszu standardowego wejścia zapisana jest jedna dodatnia liczba całkowita n równa liczbie wykonanych ruchów, $1 \leqslant n \leqslant 1\,000\,000$. W drugim wierszu zapisane jest jedno słowo długości n złożone z liter S i/lub R. Kolejne litery tego słowa reprezentują kolejno wykonane ruchy w tańcu.

Twój program powinien wypisać na standardowe wyjście w pierwszym i jedynym wierszu jedną liczbę całkowitą – minimalną liczbę ruchów (S i/lub R)

Piotr Chrząstowski, Instytut Informatyki, Uniwersytet Warszawski



Rys. 7

Rys. 7

koniecznych do rozplątania sznurków i rozciągnięcia ich poziomo i równolegle do siebie.

Postępując w sposób opisany w treści zadania, można nieźle zaplątać sznurki. Miłe z kolei jest to, że wykonując w odpowiedniej kolejności wspomniane operacje, zawsze jesteśmy w stanie dojść do stanu wyjściowego, co wcale nie jest oczywiste i co wkrótce wykażemy. Pamiętajmy tu od razu o zastrzeżeniu czynionym w treści zadania, że przy identyfikowaniu węzłów to, kto trzyma sznurki, jest nieistotne, podobnie jak to, który sznurek jest gdzie i w którą stronę jest skierowany. Mówiąc ściślej, *abstrahujemy* od nieistotnych różnic – interesuje nas jedynie rodzaj węzła. Tak samo wyglądające węzły uznajemy za *równoważne* niezależnie od tego, kto, gdzie i jak trzyma końce sznurków. Za to zwracamy uwagę na to, czy węzeł jest ułożony pionowo, czy poziomo.

O klasyfikację wszystkich takich węzłów, jakie zdefiniowano w zadaniu, jak również wielu innych, pokusił się znakomity matematyk John H. Conway (inicjał drugiego imienia jest o tyle istotny, że żyje w tej chwili dwóch matematyków o tym samym pierwszym imieniu i nazwisku – to dla tych, którzy zechcą w Internecie poszukać więcej o działalności „naszego” Conwaya, a warto!). John H. Conway znany jest szeroko jako twórca gry LIFE, najpopularniejszego chyba automatu komórkowego. Jego twierdzenie o charakteryzacji tangli (tak nazwał węzły występujące w naszym zadaniu) jest niezwykłej wprost urody. A że może być podstawą rozwiązania, przedstawimy je tu z radością.

Do tego potrzebne będzie nam pojęcie *izomorfizmu*, jedno z podstawowych pojęć w całej matematyce. Intuicyjnie oznacza ono nierozróżnialność pewnych obiektów, jeśli ograniczymy się do obserwowania, co zadany zestaw operacji (lub ogólniej relacji), które będą określone na obiektach dwóch zbiorów, z tymi obiektami wyczynia. *Algebrą* $\mathcal{A} = \langle A, o_1, \ldots, o_n\rangle$ nazwiemy zbiór A wraz z zestawem operacji $o_1, \ldots, o_n$. Zbiór A nazwiemy *nośnikiem algebry* $\mathcal{A}$. Dwie algebry mogą być różne, mimo że ich nośniki są identyczne. Na przykład algebry $\langle\mathbb{R}, +\rangle, \langle\mathbb{R}, -\rangle, \langle\mathbb{R}, +, *\rangle$ są wszystkie różne. Dla dwóch zadanych algebr dobrze jest wiedzieć, czy zachowują się tak samo. Oczywiście, aby algebry uznać za tak samo się zachowujące, konieczne jest, aby operacji w każdej z nich było tyle samo i żeby odpowiadające sobie operacje miały po tyle samo argumentów. Będziemy też żądali, żeby nośniki algebr były tak samo liczne, czyli żeby istniała funkcja różnowartościowa odwzorowująca jeden zbiór na drugi.

Jeżeli, na przykład, przyjęlibyśmy za nośnik jednej algebry zbiór liczb rzeczywistych dodatnich, a drugiej ujemnych, a w obu algebrach byłaby określona jedna operacja dodawania, to można by znaleźć funkcję $f : \mathbb{R}_+ \to \mathbb{R}_-$, która tak sparuje elementy tych dwóch zbiorów, że wykonanie dodawania w jednym zbiorze da wynik, który ta funkcja przeniesie na wynik dodawania obrazów argumentów. Innymi słowy

$$f(x_1 +_1 x_2) = f(x_1) +_2 f(x_2).$$

Indeksy przy znakach dodawania podkreślają to, że dodawanie odbywa się w innych dziedzinach, choć z punktu widzenia całego zbioru liczb rzeczywistych jest to to samo, co zwykłe dodawanie. Taka funkcja to np. $f(x) = -x$. Równie dobra byłaby zresztą funkcja $g(x) = -2x$. Zauważmy też, że gdybyśmy dołączyli jeszcze operację odejmowania (nie zawsze określoną, bo czasami wynik ucieka poza dziedzinę), to również nie sposób byłoby odróżnić wspomniane algebry od siebie: operacja odejmowania byłaby określona w zbiorze $\mathbb{R}_+$ dla argumentów x_1, x_2 wtedy i tylko wtedy, gdy byłaby określona w zbiorze $\mathbb{R}_-$ dla argumentów $f(x_1), f(x_2)$ i w takim przypadku, podobnie jak dla dodawania, dawałaby izomorficzne wyniki dla izomorficznych argumentów, czyli $f(x_1 -_1 x_2)$ byłoby równe $f(x_1) -_2 f(x_2)$. Ustalając izomorfizm dwóch algebr, musimy wskazać, które elementy nośnika pierwszej algebry odpowiadają którym elementom drugiej (u nas odpowiedniość tę wyznacza np. funkcja $f(x) = -x$) oraz którym operacjom pierwszej algebry odpowiadają które operacje drugiej (u nas operacji $+_1$ odpowiada operacja $+_2$, a operacji $-_1$ operacja $-_2$. Powiemy zatem, że algebry $\langle\mathbb{R}_+, +_1, -_1\rangle$ oraz $\langle\mathbb{R}_-, +_2, -_2\rangle$ są izomorficzne, gdyż ustaliliśmy stosowne odpowiedniości między nośnikami i operacjami obu algebr, które dają odpowiadające sobie wyniki.

Gdybyśmy dołączyli do zbioru operacji w obu algebrach mnożenie, to takie algebry nie byłyby izomorficzne, gdyż w algebrze drugiej mnożenie byłoby operacją niedającą rezultatu dla żadnych argumentów – wynik wykraczałby poza dziedzinę. Wystarczyłoby wówczas zapytać o wynik mnożenia dla odpowiadających sobie elementów, np. w pierwszej algebrze $1 * 2$ byłoby równe 2, natomiast w drugiej $(-1) * (-2)$ byłoby działaniem nieokreślonym, zatem usłyszawszy obie odpowiedzi, wiedzielibyśmy, o której algebrze rozmawiamy. Te algebry byłyby zatem nieizomorficzne.

Tutaj mieliśmy do czynienia z dość prostą funkcją ustalającą izomorfizm. „Światy” tych dwóch algebr były jak gdyby wzajemnym lustrzanym odbiciem względem funkcji $f(x) = -x$. Czasem taki izomorfizm nie jest oczywisty. Zauważmy, że algebry $\langle\mathbb{R}_+, *\rangle$ oraz $\langle\mathbb{R}, +\rangle$

są izomorficzne, a funkcją ustalającą taki izomorfizm jest np. funkcja $f(x) = \log_2 x$. Funkcja logarytm jest bowiem różnowartościowa i „na", a ponadto zachodzi wzór

$$f(x * y) = f(x) + f(y).$$

Zapytajmy jeszcze, po co w ogóle rozważać izomorfizmy. Okazuje się, że nawet gdy dwie algebry są izomorficzne, może się okazać, że wykonywanie operacji w jednej z nich jest technicznie prostsze niż w drugiej. Zatem możemy zyskać możliwość wykonywania operacji w tej z algebr, w której się to robi prościej. Jeśli tylko umielibyśmy w prosty sposób tłumaczyć z nośnika jednej algebry w nośnik drugiej i na odwrót, to mając do wykonania działanie w algebrze „trudniejszej", moglibyśmy przetłumaczyć argumenty za pomocą funkcji f ustalającej odpowiedniość między nośnikami, wykonać prostsze działanie i za pomocą funkcji f^{-1}, odwrotnej do f, uzyskać wynik w oryginalnej algebrze. Pamiętajmy, że funkcja odwrotna zawsze istnieje: wszak f musi być 1-1 i „na".

Wszyscy się chyba zgodzimy, że dodaje się prościej, niż mnoży. Zatem, jeśli umiemy szybko logarytmować i antylogarytmować, to zamiast żmudnego mnożenia moglibyśmy postąpić następująco. Szukając wyniku mnożenia x przez y, znajdujemy ich logarytmy, dodajemy je i szukamy liczby, której logarytmem byłby otrzymany wynik dodawania. Tak zresztą postępowano przez parę wieków, od czasu gdy Napier wynalazł logarytmy i ułożył ich tablice, które pozwalały szybko znajdować zarówno logarytmy, jak i antylogarytmy. Działania te wykonywano w sposób przybliżony, ale do wielu celów wystarczająco dokładny. Na tej zasadzie działał też suwak logarytmiczny – podstawowe narzędzie inżynierów epoki przedkalkulatorowej.

Ten cały wstęp był potrzebny, aby przedstawić rozwiązanie naszego zadania za pomocą tej samej techniki. Zauważmy bowiem, że węzły tworzą algebrę: nośnikiem są wszystkie rodzaje węzłów, a operacjami dwie figury taneczne, które przekształcają węzły w węzły – obie są operacjami jednoargumentowymi. Wykorzystamy tu twierdzenie Conwaya, które skomplikowaną algebrę węzłów „tłumaczy" w prostą algebrę liczb wymiernych z nieskończonością i z dwiema szczególnymi operacjami. Pierwszy krok rozwiązania polega na znalezieniu odpowiedników figur tanecznych w lepiej wyobrażalnej algebrze liczb wymiernych. Rozszerzmy więc zbiór liczb wymiernych o nieskończoność i rozważmy, jako nośnik naszej algebry, zbiór $\mathcal{Q}_\infty = Q \cup \{\infty\}$, gdzie Q jest zbiorem liczb wymiernych. Zbiór $\mathcal{Q}_\infty$ nazwiemy zbiorem liczb superwymiernych. Operacjami superwymiernymi, odpowiadającymi operacjom S i R z algebry węzłów, będą $s(x) = x + 1$ i $r(x) = -\frac{1}{x}$. Zakładamy tu, że $r(0) = \infty$, $r(\infty) = 0$ oraz $s(\infty) = \infty$. Początkowy układ poziomych węzłów odpowiadał będzie liczbie 0. Nasza algebra $\mathcal{Q}_\infty = \langle \mathcal{Q}_\infty, r, s\rangle$ okazuje się być izomorficzna z algebrą węzłów, w której nośnikiem są wszystkie rodzaje węzłów możliwych do otrzymania za pomocą operacji R i S, a te dwie jednoargumentowe operacje są jedynymi rozważanymi. To jest właśnie treścią twierdzenia Conwaya, którego dowodu nie będziemy tu przytaczali. Wykonując w odpowiedniej kolejności operacje s i r, możemy z zera uzyskać dowolną liczbę superwymierną. Wykonując analogiczne ciągi operacji S i R, możemy uzyskać wszystkie węzły zgodnie z treścią zadania. Każdy z tych węzłów może więc uzyskać unikalny numer zwany numerem Conwaya i będący odpowiadającą mu liczbą superwymierną. Różnym węzłom odpowiadają różne numery Conwaya i na odwrót: różnym liczbom superwymiernym odpowiadają różne węzły. Zauważmy przy okazji, że wykonanie dwóch kolejnych operacji r po sobie na liczbach superwymiernych jest identycznością, podobnie jak wykonanie dwóch operacji R na węzłach: dwukrotny obrót o 90 stopni jest symetrią środkową, która nie zmienia węzła, a jedynie zamienia miejscami położenia końców oraz tych, którzy te końce trzymają.

Zobaczmy na kilku przykładach, jak to wszystko działa. Układ początkowy to 0. Wykonanie operacji R prowadzi nas do węzła „nieskończoność", jako że $r(0) = \infty$. Powtórne wykonanie R oznacza powrót do zera, bo $(r(\infty) = 0)$. Zatem węzły całkowicie rozplątane poziome i pionowe odpowiadają liczbom 0 i ∞. Przy okazji: dla układu pionowego wykonanie operacji S nie zmienia go, gdyż $s(\infty) = \infty$.

Jak wygląda sytuacja po wykonaniu pojedynczego S w stanie wyjściowym na węzłach? Ano węzeł, który uzyskujemy, ma numer 1, bo $s(0) = 1$. Zgodnie z twierdzeniem Conwaya tylko jeden węzeł ma numer jeden i aby go rozplątać, należy za pomocą operacji s i r przekształcić jedynkę w zero. Zauważmy, że w tym celu wystarczy wykonać ciąg $r \cdot s$, bo $s(r(1)) = s(-1) = 0$. Łatwo sprawdzić, że faktycznie wykonanie kolejno operacji R, a potem S doprowadzi nas do węzła wyjściowego.

Gdybyśmy wykonali dwa razy S, tak jak w przykładzie z treści zadania, otrzymalibyśmy węzeł o numerze 2. Najprostsza metoda rozwikłania go będzie taka, która za pomocą minimalnej liczby operacji r i s przekształci dwójkę w zero. Zaczynamy więc od

$$2 \xrightarrow{r} -\frac{1}{2} \xrightarrow{s} \frac{1}{2} \xrightarrow{r} -2 \xrightarrow{s} -1 \xrightarrow{s} 0.$$

Ponieważ ciąg operacji $rsrss$ w algebrze liczb superwymiernych przekształca liczbę $2 = s(s(0))$ w 0, więc ciąg operacji $RSRSS$ w algebrze węzłów doprowadzi nas od węzła $S(S(0))$ do punktu wyjściowego.

Teraz rozplątać możemy każdy węzeł, korzystając z izomorfizmu naszych algebr. Mając ciąg zaplątań złożony z operacji S i R, obliczamy numer otrzymanego węzła przez wykonanie analogicznego ciągu operacji s i r na liczbach superwymiernych, a następnie przekształcamy otrzymany numer możliwie krótką

sekwencją operacji s i r doprowadzającą do zera. Rozwiązaniem będzie długość tej sekwencji.

Sprowadziliśmy zatem nasz problem do następującego: jak mając daną liczbę superwymierną, możliwie szybko uzyskać zero za pomocą operacji s i r? Zachłanny algorytm, który liczby dodatnie będzie natychmiast zamieniał na ujemne za pomocą operacji r, a na liczbach ujemnych będzie wykonywał operację s tak długo, aż otrzymamy liczbę nieujemną, okazuje się być tu optymalny. Wiadomo, że z wyjątkiem sytuacji, gdy zaczynamy od węzła ∞, czyli pionowych sznurków, ostatnim ruchem rozwiązania musi być s. Zatem będziemy atakowali zero od dołu: naszym celem powinno być osiągnięcie w miarę szybko ujemnej liczby całkowitej, a następnie wykonanie odpowiedniej liczby s-ów. Zrobimy to w taki sposób, że mianowniki kolejnych liczb ujemnych, powstałych po wykonaniu operacji r, będą malały do jedności. Ostatnia liczba ułamkowa dodatnia w rozwiązaniu powinna być postaci $\frac{1}{k}$ dla pewnego k i końcówka algorytmu będzie wyglądała tak, że po uzyskaniu $-k$ wykonamy k-krotnie operację s. Z kolei liczbę $\frac{1}{k}$ można uzyskać tylko za pomocą operacji s (inaczej bezsensownie wyskoczylibyśmy z „dobrej" ujemnej liczby całkowitej). Niżej pokażemy, że podana metoda daje zawsze minimalną liczbę ruchów rozplątującą węzeł.

Możemy więc zastosować następujący algorytm. Niech l/m będzie liczbą wymierną lub minus nieskończonością reprezentującą zaplątanie sznurków.

1. `if` $l = 0$ `then exit`

2. `else if` $(l/m > 0) \vee (m = 0)$ `then` $(l, m) := (-m, l)$ // (operacja R)

3. `else` $l := l \bmod m$ //(ciąg operacji S)

Udowodnimy, że ten algorytm prowadzi do rozplątania sznurków za pomocą minimalnej liczby ruchów.

Wzór 1. Dla liczb naturalnych $l > 0$ i $m > 0$

$$l = (-m) \bmod (l + m) \tag{1}$$

Dowód wzoru 1

$$l = (-m) \bmod (l+m) = -m - \left\lfloor \frac{-m}{l+m} \right\rfloor (l+m)$$

$$l + m = -\left\lfloor \frac{-m}{l+m} \right\rfloor (l+m)$$

$$1 = -\left\lfloor \frac{-m}{l+m} \right\rfloor$$

$$1 = \left\lceil \frac{m}{l+m} \right\rceil$$

Lemat 1. Algorytm się zatrzymuje.

Dowód lematu 1. Wykażemy, że dla dowolnej liczby wymiernej ujemnej postaci $-\frac{l}{m}$, dla $l, m > 0$, wykonywanie kolejnych kroków algorytmu doprowadzi nas w końcu do wartości $x = 0$. Jeśli $m = 1$, to po l-krotnym wykonaniu s dostaniemy zero. Jeśli zaś $m > 1$, to wykonując kroki algorytmu, dojdziemy do innej liczby wymiernej $-l'/m'$, takiej że $m' < m$. Niech bowiem $x = -l/m$ oraz $l > 0$ i $m > 1$. Dla $x < 0$ wykonujemy ciąg operacji s i po jego wykonaniu dojdziemy do sytuacji, gdy x po raz pierwszy stanie się większe od zera. Niech wtedy $x = l_1/m$, gdzie $m > l_1 > 0$. Teraz wykonujemy R i mamy $x = -m/l_1$. Mianownik zmalał, a ułamek jest ujemny. Zatem zawsze po wykonaniu ciągu S, aż do uzyskania liczby dodatniej, zakończonego jednym R, dostaniemy ułamek ujemny o mniejszym mianowniku. Nie można w nieskończoność zmniejszać dodatniego mianownika. Zatem po skończonej liczbie kroków mianownik stanie się równy 1, a to, jak wiemy, doprowadza do zatrzymania całego algorytmu.

Lemat 2. Algorytm wykonuje minimalną liczbę kroków.

Dowód lematu 2. Możliwe są następujące ruchy dla $l \neq 0, m \geqslant 0$.

- $m = 0$ i S – zły oczywiście, bo nie zmienia węzła,
- $m = 0$ i R – dobry, bo w jednym ruchu doprowadza nas do zera. Szybciej nie można.
- $l, m > 0$ i S – zły.

Wykonajmy S, a potem tak, jak każe algorytm, i pokażmy, że ta sekwencja nie jest najkrótsza. Mamy bowiem

$$\begin{aligned} l/m &\to (l+m)/m \to \\ &\to -m/(l+m) \to \\ &\to ((-m) \bmod (l+m))/(l+m) \overset{(\text{ze wzoru (1)})}{=} \\ &= l/(l+m) \to \\ &\to -(l+m)/l \to \\ &\to ((-(l+m)) \bmod l)/l = \\ &= ((-m) \bmod l)/l, \end{aligned}$$

czyli 5 kroków. Lepszą sekwencją, po której dochodzimy do tej samej liczby, jest

$$l/m \to -m/l \to ((-m) \bmod l),$$

co wymaga 2 kroków.

- $l, m > 0$ i R – dobry,
- $m > 0,\ l < 0$ i S – dobry,
- $m > 0,\ l < 0$ i R – zły.

Niech bowiem $l_1 = -l > 0$, po wykonaniu R mamy $x = m/l_1$. Teraz wykonanie R nie ma sensu, gdyż wrócimy do stanu poprzedniego, wykonanie S jest złe $(m/l_1 > 0)$.

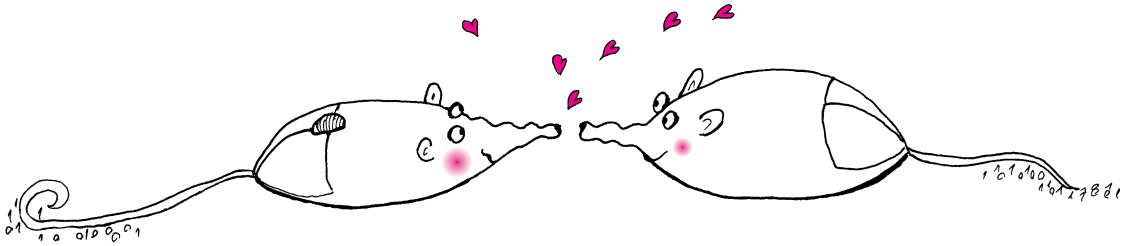
Zatem wystarczy jeszcze wykazać, że jakikolwiek zły krok nie prowadzi do optymalnego rozwiązania. Jeśli będziemy wykonywać tylko S dla liczby dodatniej, to nigdy nie dojdziemy do rozplątania, bo będziemy zwiększać liczbę dodatnią i nie osiągniemy zera. Załóżmy więc, że po złym kroku S (dla liczby dodatniej) kiedyś wykonamy R, czyli tak jak nakazuje algorytm. Wykonanie dobrego kroku po złym to już

przypadek rozpatrywany wcześniej – zawsze można lepiej wyjść na niewykonywaniu tego ostatniego S, tylko pójściu na skróty. Jeśli więc S było wykonane dla liczby dodatniej, to tylko zwiększyło niepotrzebnie liczbę kroków konieczną do rozplątania. Dla operacji R jest jeszcze łatwiej, gdyż R^{2k} nic nie daje, a $R^{2k+1} = R$. Sensowne jest zatem wykonywanie jedynie pojedynczych R przetykanych S-ami. Podsumowując: R-y należy wykonywać tylko jednokrotnie, a S-y tylko dla liczb ujemnych. I tak właśnie działa nasz algorytm, więc dochodzi najszybciej do stanu końcowego.

Wiemy już, jak dojść do rozwiązania końcowego, pozostaje więc zliczyć, ile potrzebujemy ruchów. Jeśli algorytm wykonuje operację R, to zwiększamy licznik o 1, gdy zaś wykonuje S (właściwie ciąg operacji S), dodajemy do licznika $||\frac{l}{m}]|$ (ułamek jest ujemny). Rzecz jasna, zakresy danych wymuszały implementację własnej arytmetyki umożliwiającej wykonanie tych działań.

Przy wykorzystywaniu dowodu lematu 2 widać, iż podczas zaplątywania operacja S zwiększa liczbę kroków algorytmu o co najwyżej 3. Liczba operacji S w ciągu wejściowym jest oczywiście nie większa niż jego długość. Złożoność czasowa algorytmu wynosi więc $O(n)$, gdzie n jest długością ciągu podanego na wejściu.

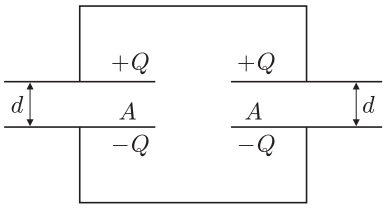
Zadanie to pokazuje, jak przywykli do myślenia abstrakcyjnego (liczby są wszak abstrakcyjne), a nie przywykli do plątania realnych węzłów, lepiej poruszamy się w świecie abstrakcji, choć przecież mogłoby być zupełnie odwrotnie: jakieś plemię biegłe w tańcach gordyjskich mogłoby tłumaczyć dzieciom liczby wymierne w drugą stronę. Patrz synku! Liczba $\frac{1}{2}$ to tak, jakbyś zrobił $SSRS\ldots$



Zadania

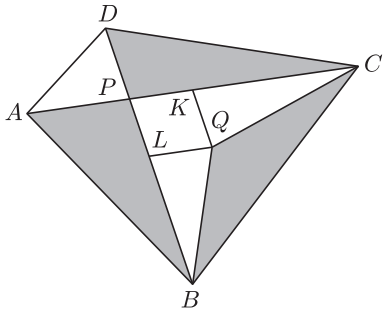
Redaguje Mikołaj KORZYŃSKI

F 649. Zasada zachowania energii oraz pędu zabrania zamiany fotonu w parę elektron – pozyton. Taka przemiana jest jednak możliwa, jeśli w reakcji uczestniczy inna, ciężka cząstka o masie M. Jaka minimalna energia fotonu jest potrzebna, by reakcja zaszła? Masę elektronu m_e traktować jako daną.
Rozwiązanie na str. 22



Rys. 1

F 650. Dwa identyczne kondensatory płaskie o powierzchni płytek A i ładunku Q oraz odległości między płytkami d połączone są jak na rysunku 1. Zmniejszamy odległość między płytkami jednego z kondensatorów. Jak zmienia się energia całkowita układu? Porównać z sytuacją pojedynczego naładowanego kondensatora. Ile wynosi siła przyciągania okładek kondensatora?
Rozwiązanie na str. 22



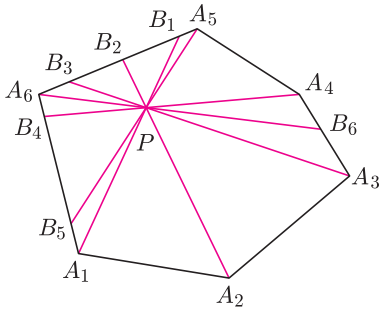
Rys. 2

Redaguje Waldemar POMPE

M 1105. Przekątne AC i BD czworokąta wypukłego $ABCD$ przecinają się w punkcie P. Punkty K i L są odpowiednio środkami przekątnych AC i BD (rys. 2). Punkt Q jest takim punktem, że czworokąt $KPLQ$ jest równoległobokiem. Udowodnić, że

$$[ABP] + [CDP] = 2\,[BCQ]\,,$$

gdzie $[XYZ]$ oznacza pole trójkąta XYZ.
Rozwiązanie na str. 18



Rys. 3

M 1106. Danych jest dziesięć liczb naturalnych dwucyfrowych. Dowieść, że spośród tych liczb można wyłonić dwa takie różne podzbiory, że sumy liczb w obu podzbiorach są jednakowe.
Rozwiązanie na str. 22

M 1107. Punkt P leży wewnątrz $(2n)$-kąta wypukłego $A_1A_2\ldots A_{2n}$ i nie należy do żadnej z jego przekątnych. Proste $A_1P, A_2P, \ldots, A_{2n}P$ przecinają obwód danego wielokąta po raz drugi w punktach $B_1, B_2, \ldots, B_{2n}$ (zob. rys. 3 dla $n = 3$). Wykazać, że na pewnym boku danego $(2n)$-kąta nie leży żaden z punktów B_i.
Rozwiązanie na str. 24