Numer jest w całości poświęcony informatyce i napisany przez informatyków.
Reprezentował ich w Kolegium Redakcyjnym
mgr Jarosław Deminet.

SPIS TREŚCI

NUMERU 9 (141)

W następnym numerze:
Pierścienie Saturna

„Delta"
matematyczno-fizyczno-astronomiczny miesięcznik popularny Polskiego Towarzystwa Matematycznego, Polskiego Towarzystwa Fizycznego i Polskiego Towarzystwa Astronomicznego wydawany przy poparciu Ministerstwa Oświaty i Wychowania

Adres Redakcji
ul. Koszykowa 6a
00-564 Warszawa
tel. 21-19-85

Krajowe Wydawnictwo Czasopism
RSW „Prasa—Książka—Ruch"
ul. Noakowskiego 14
00-666 Warszawa
tel. centr. 25-72-91 do 93
Biuro Reklam i Propagandy
tel. 25-56-26
Nakład 30 000 egz. Objętość 2 ark. wyd; 2,50 ark. druk;
papier offsetowy V kl. 70 g.
Wydrukowano w drukarni
im. Rewolucji Październikowej
Warszawa, ul. Mińska 65.
Nr zam. 587/85. N-59

WARUNKI PRENUMERATY

Cena prenumeraty rocznej zł 240,— cena prenumeraty półrocznej zł 120,—
1. dla osób prawnych — instytucji i zakładów pracy:
— instytucje i zakłady pracy zlokalizowane w miastach wojewódzkich i pozostałych miastach, w których znajdują się siedziby oddziałów RSW „Prasa-Książka-Ruch", zamawiają prenumeratę w tych oddziałach,
— instytucje i zakłady pracy zlokalizowane w miejscowościach, gdzie nie ma oddziałów RSW „Prasa-Książka-Ruch" i na terenach wiejskich opłacają prenumeratę w urzędach pocztowych i u doręczycieli.
2. dla osób fizycznych — indywidualnych prenumeratorów:
— osoby fizyczne zamieszkałe na wsi i w miejscowościach, gdzie nie ma oddziałów RSW „Prasa-Książka-Ruch", opłacają prenumeratę w urzędach pocztowych i u doręczycieli,
— osoby fizyczne zamieszkałe w miastach — siedzibach oddziałów RSW „Prasa-Książka-Ruch" opłacają prenumeratę wyłącznie w urzędach pocztowych nadawczo-oddawczych właściwych dla miejsca zamieszkania prenumeratora. Wpłaty dokonują używając „blankietu wpłaty" na rachunek bankowy miejscowego oddziału RSW „Prasa-Książka-Ruch".
3. Prenumeratę ze zleceniem wysyłki za granicę przyjmuje RSW „Prasa-Książka-Ruch", Centrala Kolportażu Prasy i Wydawnictw, ul. Towarowa 28, 00-958 Warszawa, konto NBP XV Oddział w W-wie Nr 1153-201045-139-11. Prenumerata ze zleceniem wysyłki za granicę pocztą zwykłą jest droższa od prenumeraty krajowej o 50% dla zleceniodawców indywidualnych i o 100% dla zlecających instytucji i zakładów pracy.
Terminy przyjmowania prenumeraty na kraj i za granicę:
— do dnia 10 listopada na I kwartał, I półrocze roku następnego oraz cały rok następny,
— do dnia 1-go każdego miesiąca poprzedzającego okres prenumeraty roku bieżącego,

Sprzedaż numerów bieżących i uprzednich

Instytucje państwowe i społeczne, zakłady pracy, szkoły i czytelnicy indywidualni mogą nabywać „DELTĘ":
— w Księgarni Ośrodka Wydawnictw Naukowych PAN, Warszawa — Pałac Kultury,
— w Głównej Księgarni Naukowej, Warszawa — ul. Krakowskie Przedmieście 7,
— w Księgarni Ossolineum, Wrocław — Rynek 8,
— w Księgarni Naukowej, Kraków — Podwale 6.

Orders for this periodical from abroad can be placed with „Ars Polona" Krakowskie Przedmieście 7, 00-068 Warszawa, Poland or with
— Kubon & Sagner, Inhaber Otto Sagner, D8 München 34, Postfach 68, Bundesrepublik Deutschland,
— Earlcourt Publications Ltd., 130 Shepard Bush Centre, London W12, Great Britain,
— Licosa Commisionaria Sansoni, Via Lamarmora 45, 50 121 Firenze. Italia.

Cena 1 egzemplarza zł 20,—

Informatyka bez komputerów?

Prof. dr Władysław M. TURSKI

To nie będzie tekst o nędzy. „Bez komputerów" nie dlatego, że ich nie ma, lecz dlatego, że być może nie są potrzebne. No, nie dla całej informatyki, ale dla kawałka, aspektu, podejścia.

Historycznie rzecz biorąc, nie można zaprzeczyć, że problemy obliczalności, którymi zajmował się Turing przed wojną (II światową), należą do informatyki. A że Turing nie używał wtedy komputerów, to sprawa oczywista. Są więc wielkie problemy informatyczne, które można formułować i badać bez związku z komputerami. W każdym razie bez związku z konkretnymi komputerami, boć przecie Turing skonstruował czysto myślowy model uniwersalnej maszyny liczącej.

Zauważmy, że maszyna Turinga nigdy nie pretendowała do roli wzorca czy prototypu jakiejkolwiek rzeczywistej maszyny liczącej, że nie ma ona żadnego związku z konstrukcją konkretnych komputerów. A jednak ... chcąc wykazać uniwersalność jakiejkolwiek konkretnej maszyny liczącej, demonstrujemy jej równoważność właśnie owej abstrakcyjnej maszynie Turinga. Praktyczne znaczenie mają, naturalnie, konkretne komputery, twory z układów elektronicznych, obudowanych różnymi elektro-mechanicznymi i foto-elektronicznymi urządzeniami peryferyjnymi. Ale znaczenie poznawcze ma maszyna Turinga: to za jej pomocą tworzy się pojęcie obliczalności i bada zakres możliwości obliczeń.

Niektórzy powiedzą: ba, ale to przecież matematyka, albo nawet filozofia, a nie żadna informatyka. Nie warto zapewne sprzeczać się o etykietki; ze scholastycznych podziałów, tu matematyka, tu informatyka, niewiele wynika. Zauważmy jednak, że gdyby nie pojawiły się rzeczywiste komputery, programowane maszyny liczące, niewielu byłoby amatorów badania problemów obliczalności. Gdyby nie komputery, problem ten byłby niezwykle ezoteryczny, rzeczywiście ważny tylko dla wąskiego grona filozofów matematyki. Ale komputery są. (I to jak jeszcze!) A przez to, że są, że stosuje się je tak powszechnie, do tak szalenie praktycznych zadań, problem obliczalności i wiele problemów pokrewnych stało się ważne dla licznego grona.

W historii nauki zjawisko to wcale nie nowe: działalność praktyczna wyzwala zainteresowanie problemami, które w zasadzie mogły być sformułowane i roztrząsane całkiem abstrakcyjnie, w zupełnej izolacji od jakichkolwiek konkretów. Tyle tylko, że bez owych konkretów, bez owej działalności praktycznej, albo nikomu by nie przyszły do głowy, albo stanowiłyby co najwyżej ciekawostkę.

Pojawienie się komputerów i niepohamowany rozwój ich zastosowań pobudza zainteresowanie bardzo głębokimi problemami, które bez istnienia komputerów prawie na pewno nie znalazłyby uznania badaczy.

Wspomnieliśmy już o problemie obliczalności i jego pochodnych. Z grubsza powiedziawszy, chodzi o to, jakie zadania dopuszczają rozwiązania algorytmiczne oraz o to, jak zmienia się koszt rozwiązania zadania w zależności od jego (tj. zadania) rozmiarów charakterystycznych.

Oczywiście nie wszystkie zadania można rozwiązać algorytmicznie. Wiadomo na przykład, że nie można skonstruować ogólnego algorytmu badania, czy zadany algorytm zawsze prowadzi do kończących się obliczeń, nie można też ułożyć ogólnego algorytmu podziału zadanego algorytmu na wzajemnie niezależne części (części, które można wykonywać niezależnie). Ale są zadania, co do których nie wiadomo, czy w ogóle mogą być rozwiązane algorytmicznie, mimo że skądinąd wiadomo, że są one rozwiązywalne. Należy do nich tzw. problem rozpoznawania postaci: nie wiadomo, czy istnieje algorytm pozwalający np. odróżnić A od innych liter. Co więcej, nie ma żadnej pewności, czy uda się ten problem kiedykolwiek rozstrzygnąć, tj. albo ustalić na pewno, że odpowiedni algorytm istnieje, albo udowodnić, że nie można go skonstruować.

Jeśli wiadomo, że algorytm istnieje, to naturalnie chcielibyśmy go poznać; często dowód istnienia jest konstrukcyjny i wtedy łapiemy dwie sroki za ogon: wiemy, że algorytm istnieje i znamy jakiś, lepszy czy gorszy, konkretny algorytm. Ale nie zawsze życie jest takie piękne ...

Samo pojęcie kosztu wykonania algorytmu wynikło z niesłychanie praktycznych przesłanek: wiadomo, że moc obliczeniowa komputerów ustalonej klasy rośnie z kwadratem ich ceny. Jeśli więc koszt wykonania algorytmu rośnie z rozmiarem zadania nie szybciej niż kwadrat (no, niechby nie szybciej niż n-ta potęga!) tego rozmiaru, mamy do czynienia z dosyć rozsądną sytuacją: koszt rozwiązania na jednostkę rozmiaru zadania jest mniej więcej stały (albo rośnie w sposób umiarkowany). Ale jeśli koszt wykonania algorytmu rośnie wykładniczo, nie ma właściwie żadnej szansy na rozwiązanie dużego „egzemplarza" takiego zadania.

Można powiedzieć, no dobrze, w takim przypadku trzeba zmienić algorytm — dla danego zadania znaleźć tańszy algorytm. Ale czy to będzie możliwe? Podkreślmy, że pytamy o możliwość, to bardziej fundamentalne pytanie niż „czy umiemy taki tańszy algorytm skonstruować?". Dla niektórych zadań znamy dobre dolne oszacowanie kosztu algorytmów (wiemy, że tańszych nie ma i oszacowanie nie jest banalne, w rodzaju „koszt jest dodatni"), ale wszystkie znane algorytmy są sporo droższe od tego oszacowania. Zachęca to do intensywnych poszukiwań, obiecuje znaczną

premię pomysłowemu odkrywcy. A może to tylko złudzenie, miraż, może to dolne oszacowanie jest zbyt optymistyczne (jak ta góralska odpowiedź na pytanie „daleko do Krzeptówek?" — „Ano, mila i oho". Myślisz, że to mila i kawałeczek, a to mila i pięciokroć tyle).

Wiemy też, że są zadania, dla których nie ma najtańszych algorytmów: ze znajomości każdego algorytmu rozwiązującego je wynika możliwość skonstruowania tańszego.

Nic dziwnego, że badanie złożoności algorytmów (bo tak oficjalnie nazywa się studiowanie problematyki wrodzonych kosztów algorytmów) rozwinęło się w rozległą i piękną dziedzinę informatyki, przekraczającą granice podziałów, które jeszcze kilka lat temu uważano za nieprzekraczalne (np. między analizą numeryczną i metodologią programowania).

Weźmy inny przykład. Komputery z łatwością wykonują najbardziej złożone rachunki logiczne. Zasady rachunku logicznego przywykliśmy uważać za rachunkowe wyrażenie reguł myślenia. Wynikałoby z tego, że komputery „myślą". Ale przepaść między tym, co „potrafi" komputer, a tym, co potrafi kilkuletnie dziecko, nie zmniejsza się ani trochę mimo dwudziestopięcioletnich już prób stworzenia „sztucznej inteligencji".

To prawda, że komputery świetnie grają w szachy; w każdym razie dużo lepiej niż autor tego artykułu. Ale podczas gdy kilkuletnie dziecko równie łatwo można nauczyć dowolnej z kilkudziesięciu (czy więcej) gier, dla komputera trzeba pisać specjalny program dla każdej gry oddzielnie. Z biegłości wykonywania działań rachunku logicznego wcale nie wynika biegłość myślenia czy rozumowania. Co więc wyrażają formuły logiczne? Reguły myślenia? Czy może są one po prostu przyjętym (w cywilizacji pochodzącej z basenu Morza Śródziemnego) sposobem przekazywania argumentów?

Powiesz, Czytelniku, że to już naprawdę czysta filozofia? Ależ tak! Tylko że właśnie dzięki komputerom odpowiedź na to pytanie ma znaczenie nie tylko poznawcze, ale także czysto praktyczne. Pojawienie się komputerów to bodajże najważniejsze zdarzenie, prawdziwy przełom w filozofii, nawet jeśli nie wszyscy filozofowie jeszcze to sobie uświadomili!

Wszystkie znane dziś komputery licząc — dyssypują energię. Ma to swoje znaczenie praktyczne: jak odprowadzić ogromne ilości ciepła wytwarzające się w układach liczących (to nieważne, że w układach VLSI chodzi o ułamki wata — wydzielają się one bowiem w mikroskopijnej objętości!). Ale ma to też ogromne znaczenie poznawcze: wszystkie procesy, dzięki którym komputery liczą, prowadzą do wydatków energii, nie na skutek niedoskonałości konstrukcji, ale z zasady (zmiana polaryzacji elementu musi wywołać zmianę pola elektromagnetycznego, a więc emisję energii poza element). Wynika stąd fundamentalna nieodwracalność procesu obliczeniowego (kłania się termodynamika, entropia itd.). A przecież można sobie wyobrazić — teoretyczny — model komputera liczącego bez strat energii: elementami liczącymi byłyby idealnie sprężyste kule bilardowe, toczące się bez tarcia po stole, na którym ustawiano by program z idealnie sprężystych zastawek, od których kule odbijałyby się tak jak kulka we „flipperze". Dane zadawalibyśmy przez wtoczenie kul przez niektóre z oznaczonych bramek (kodem byłoby: bramka z wtoczoną kulą — 1, bramka bez kuli — 0), a wyniki odczytywalibyśmy z tego, przez które bramki kule się wytoczą (kodowanie podobne jak poprzednio). Pomijając różne niedoskonałości modelu, wpływ ciał niebieskich, prądów powietrza itp., mamy komputer idealny — liczący bez strat energii, a więc w sposób odwracalny.

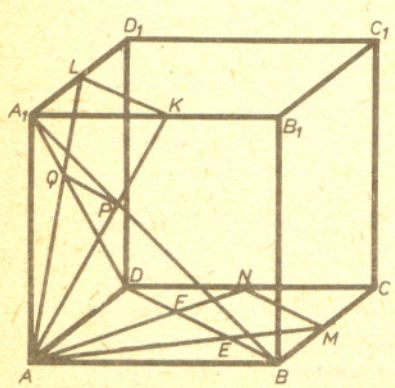
Czy nieodwracalność procesów obliczeniowych w dzisiejszych komputerach wpływa na ich realną użyteczność? A czy będzie na nią wpływać w przyszłości, gdy postępy miniaturyzacji dadzą nam zamiast mikrokomputerów nanokomputery i pikokomputery? Problem informatyczny do rozwiązania — chyba na pewno — bez komputerów!

Ale żart na stronę, gdyby komputery się nie przegrzewały, nikt chyba nie podjąłby tematu, czy liczenie musi być dyssypatywne! A ostatnio całkiem poważni uczeni, w całkiem poważnej uczelni (MIT w amerykańskim Cambridge), podjęli próbę stworzenia nowej fizyki, obejmującej kosmologię, fizykę cząstek elementarnych, mechanikę kwantową i teorię pola, jednym słowem: próbę stworzenia jednolitej teorii świata, traktując świat jako komputer, a dokładniej, jako automat komórkowy. Toż to odwrócenie „normalnego" porządku rzeczy — nie dzięki zdobyczom fizyki mamy komputery, lecz dzięki pojęciom i wynikom informatyki próbuje się uprościć podstawy fizyki!

Informatykę można uprawiać bez komputerów, zwłaszcza gdy w pokoju obok ma się superkomputer!

P.S. Autor niniejszego nie lubi sztuki abstrakcyjnej. Uznaje natomiast, że artyście przysługuje prawo wyboru stylu najbardziej odpowiadającego wizji świata, jaką ma artysta, zwłaszcza temu, który udowodnił (na przykładach!), że potrafi malować obrazy „jak żywe". Wiem wtedy, że prawo wyboru nie zostało użyte do zamaskowania indolencji.

Rozwiązanie zadania M 409. Niech E i F będą punktami przecięcia AM i AN z BD. Rozważmy sześcian $ABCDA_1B_1C_1D_1$. Wybierzmy na krawędziach A_1B_1 i A_1D_1 punkty K i L tak, aby $A_1K = CN$ i $A_1L = CM$. Niech P i Q będą odpowiednio punktami przecięcia prostych AK i BA_1, AL i DA_1. W trójkącie A_1PQ mamy $A_1P = BE$ ($BM = AB - CM = CN = A_1K$) i $A_1Q = DF$ ($A_1L = DN$), a z przystawania trójkątów AKL i AMN i równości $AP = AE$ i $AQ = AF$ wynika równość $PQ = EF$. Kąt PA_1Q jest równy $\frac{\pi}{3}$, ponieważ trójkąt A_1BD jest równoboczny.



Przykład praktyczny

Jedno z polskich biur podróży spotkało się w swej działalności z następującym problemem: wykorzystując własną bazę noclegową należało zorganizować pobyt grup wycieczkowych podróżujących po Polsce. Istniało kilkadziesiąt wariantów programu realizowanego z daną grupą różniących się między sobą czasem trwania, trasą przejazdu, liczbą dni spędzanych przez uczestników w poszczególnych miastach na trasie oraz szczegółami pobytu w każdym z miast. Turyści podróżujący w takich grupach stanowili około 70% wszystkich klientów obsługiwanych przez biuro. Dlatego też niezwykle istotne okazało się takie zaprojektowanie terminów przyjazdów grup w poszczególnych wariantach, aby można było obsłużyć możliwie dużo klientów korzystając wyłącznie z własnych hoteli. Próby „ręcznego" rozwiązania problemu przez kilka lat nie doprowadziły do pożądanego rezultatu. Powstał więc pomysł projektowania przy użyciu komputera. Badania podjęte przez zespół informatyków doprowadziły do powstania rozwiązania podstawowego, udoskonalanego następnie w kilku krokach. Rozwiązanie podstawowe polegało na określeniu kryterium oceny wyprodukowanego projektu. Stwierdzono, że projekt jest tym lepszy, im bardziej równomierne jest obciążenie hoteli w trakcie sezonu. Jeżeli teraz w kolejnych krokach iteracji dodawać po jednej grupie do wygenerowanego uprzednio projektu tak, by ocena nowego była jak najlepsza, to można się spodziewać, że otrzymamy po zakończeniu programu zupełnie dobre rozwiązanie. Tak też było w istocie, lecz okazało się, że możliwe (a zatem konieczne) są pewne modyfikacje algorytmu oraz całego rozwiązania zwiększające jego przydatność dla biura. W pierwszym kroku podzielono zbiór hoteli na klasy o różnych priorytetach wyrażających ich ważność dla biura. Teraz algorytm dążył do osiągnięcia równomiernego obciążenia przede wszystkim w najważniejszych hotelach. Okazało się, że taka modyfikacja znacznie poprawiła ekonomiczne efekty poprzedniego rozwiązania. Kolejnym ulepszeniem była zmiana kryterium oceny. Obecnie żądano, aby obciążenie było równomierne w trakcie sezonu, ale ze zmniejszonymi przyjęciami na początku i końcu sezonu. Było to związane z trudnościami towarzyszącymi początkowej fazie działalności hotelu i spiętrzeniem prac pod koniec sezonu. Przedstawione powyżej rozwiązanie nie dopuszczało możliwości ingerencji pracowników biura w proces przygotowywania projektu sezonu. Okazało się jednak, że taka ingerencja jest bardzo pożądana, a z punktu widzenia biura czasami wręcz konieczna. Typowy przykład stanowi projektowanie obciążenia hotelu w Gdańsku. W dniach festiwalu sopockiego występuje tam duże zapotrzebowanie na wolne miejsca w hotelach. Można zatem założyć przyjęcie w tych dniach mniejszej liczby grup, jako że miejsca i tak będą wykorzystane. Problemy tego rodzaju spowodowały konieczność wykonania programów dodających, usuwających i zmieniających projektowane terminy przyjazdów grup. Zostały one wykonane w ten sposób, aby po każdej operacji można było zobaczyć, jakie zmiany w obciążeniu hoteli ona wywołała. Istniał również mechanizm pozwalający odwołać dowolne przeprowadzone zmiany.

Opisany powyżej system przeznaczony był do wykorzystywania w następujących etapach:

1. projektowanie priorytetów hoteli i tych terminów przyjazdów grup, które nie mogą ulec zmianie,
2. automatyczne projektowanie pozostałych terminów,
3. ewentualne modyfikacje projektu,
4. podpisanie umów z kontrahentami,
5. uwzględnienie zmian w stosunku do projektu wynikłych w trakcie podpisywania umów.

Jego zastosowanie pozwoliło znacznie skrócić czas projektowania sezonu i poprawić jakość projektu. Projekt został wykonany na zlecenie „Almaturu" przez Koło Naukowe Informatyków Uniwersytetu Warszawskiego w 1980 roku. Jego efekty były równoważne uruchomieniu kilku dodatkowych hoteli.

mgr Jacek PACHOLCZYK

Rozwiązanie zadania F 180. Kulka znajduje się w równowadze względem układu odniesienia wirującego z prędkością kątową Ω, muszą więc być spełnione następujące warunki:
1. Suma sił działających na kulkę musi być równa zeru.
2. Żadnemu z dopuszczanych przez więzy przesunięć względem położenia równowagi nie może towarzyszyć pojawienie się siły zwiększającej to przesunięcie.
Oznaczając odległość kulki od osi obrotu przez r, a przez $F(r)$ zależność wypadkowej działających sił od r możemy poprzednie warunki zapisać w postaci

$$F(r_0) = 0 \quad \text{i} \quad \left(\frac{dF}{dr}\right)_{r=r_0} \leq 0.$$

W naszym przypadku istotne są siły: sprężystości $F_s = -kr$ i siła odśrodkowa $F_{od} = M\Omega^2 r$, zatem $F(r) = -kr + M\Omega^2 r = (M\Omega^2 - k)r$. Dla $r \neq 0$ możliwa jest więc jedynie równowaga obojętna $\left(\frac{dF}{dr} = 0\right)$, gdy

$$\Omega = \sqrt{\frac{k}{M}}.$$

Poprawność algorytmu — odpowiedź

Błąd bezwzględny obliczonych wartości x_1 i x_2 jest tego samego rzędu, natomiast błędy względne (tzn. ilorazy błędu i wartości $\frac{|\varepsilon_1|}{|x_1|}$ i $\frac{|\varepsilon_2|}{|x_2|}$) są istotnie różne.
Aby tego uniknąć, należy użyć szkolnego algorytmu do obliczenia tego pierwiastka, którego moduł jest większy $\left(\text{tzn. } \frac{-b-\sqrt{\Delta}}{2a}\right.$, jeśli $b > 0$ lub $\left.\frac{-b+\sqrt{\Delta}}{2a}\right.$ — jeśli $\left.b < 0\right)$, natomiast drugi pierwiastek obliczyć z równości $x_1 \cdot x_2 = \frac{c}{a}$ $\left(\text{czyli } x_2 = \frac{c}{a \cdot x_1}\right)$.
W naszym przypadku $b < 0$, $x_2 \approx 2 \cdot 10^5$, $x_1 = \frac{c}{x_2} \approx 10^{-5}$

O złożoności algorytmów

Mgr Teresa PRZYTYCKA

Rozważając algorytm, czyli przepis automatycznego postępowania prowadzący do rozwiązania określonego zadania, chcielibyśmy zazwyczaj wiedzieć, jak szybko uzyskamy rozwiązanie stosując właśnie ten algorytm. Pozwoli nam to wybrać spośród wielu algorytmów możliwie najlepszy i, co nie mniej ważne, oszacować jego realizowalność na dostępnej nam maszynie. Załóżmy np., że maszyna wykonuje 10^5 kroków na sekundę, a liczba kroków naszego algorytmu jest funkcją danej n i wyraża się przez n^n. Wówczas dla $n = 5$ maszyna będzie pracowała ułamek sekundy, dla $n = 10$ około doby, dla $n = 11$ ponad 33 dni, podczas gdy dla $n = 14$ powyżej 3000 lat! Jeśli więc liczba kroków algorytmu rośnie tak szybko wraz ze wzrostem n, to już dla całkiem małych wartości n algorytm praktycznie nie jest realizowalny.

Oceny czasochłonności algorytmu dokonujemy na podstawie jego funkcji złożoności czasowej. Jest to funkcja przyporządkowująca rozmiarowi danych liczbę kroków algorytmu.

Jeżeli dopuszczamy możliwość wystąpienia różnych danych o tym samym rozmiarze (porównaj uwagę na marginesie), to przyporządkowując rozmiarowi danych maksymalną liczbę kroków wykonywanych dla danych tego rozmiaru otrzymamy funkcję pesymistycznej złożoności czasowej (T_{pes}).

W przypadku, gdy rozmiarowi przyporządkowujemy średnią liczbę kroków po zbiorze danych tego rozmiaru (dokładniej — wartość oczekiwaną, porównaj margines), otrzymujemy funkcję oczekiwanej złożoności czasowej (T_{ave}).

W praktyce nie interesuje nas dokładna postać funkcji złożoności czasowej, ale jej rząd. Chcemy bowiem wiedzieć, jak szybko rośnie liczba kroków wraz ze wzrostem rozmiaru danych. Pytamy więc, czy rośnie ona tak szybko jak funkcja n^2, czy jak $n!$, czy też jak $\log n$, a nie czy jest to dokładnie $\frac{1}{2}n^2+7$, czy $n!-8$, czy też $120 \log n$. Zamiast wszystkich kroków algorytmu możemy więc liczyć liczbę operacji dominujących, przy czym za operacje dominujące przyjmujemy takie operacje, że funkcja przyporządkowująca rozmiarowi danych liczbę tych operacji jest rzędu funkcji złożoności czasowej.

Rozważmy dla przykładu różne algorytmy dla zadania sortowania. Zadanie to można sformułować w następujący sposób: Dany jest ciąg n liczb $a_1, a_2, \ldots, a_n$; dokonać takiej permutacji tego ciągu, aby $a_1 \leqslant a_2 \ldots \leqslant a_n$. W zadaniu tym rozmiarem danych będzie długość ciągu, we wszystkich zaś prezentowanych algorytmach operacją dominującą będzie porównanie dwu elementów ciągu. Przy analizie funkcji złożoności będziemy zakładać, że $a_1, \ldots, a_n$ jest permutacją n różnych liczb oraz że dla ustalonego n każda permutacja jest jednakowo prawdopodobna.

Pierwszy z proponowanych algorytmów polega na znalezieniu największego elementu w danym ciągu, potem drugiego co do wielkości itd. Metoda ta nosi nazwę sortowania przez wybór.

Dla każdego problemu algorytmicznego możemy określić funkcję ω, która danym dla tego problemu przyporządkowuje ich wymiar. Niech D będzie zbiorem danych, W — zbiorem wymiarów. Zatem $\omega: D \to W$. Np. dla zadania znajdowania minimalnego elementu w ciągu n elementów D będzie zbiorem ciągów skończonych, a ω będzie przyporządkowywać ciągowi jego długość ($W = N$).

Niech $f: D \to N$ i $f(d)$ będzie liczbą kroków dla danych d, zaś p_{dn} będzie prawdopodobieństwem wystąpienia danej d w zbiorze danych o rozmiarze n ($n \in W$). Wówczas

$$T_{pes}(n) = \sup\{f(d) : \omega(d) = n\},$$

$$T_{ave}(n) = \sum_{\omega(d)=n} f(d)p_{dn}.$$

Algorytm 1: Sortowanie przez wybór

$i := n$

dopóki $i > 1$ powtarzaj:

- {wybieramy największy element z podciągu $a_1, \ldots, a_i$}
- $max := 1$;
- {max jest indeksem największego z rozważonych dotychczas elementów}
- $k := 2$;
- dopóki $k \leqslant i$ powtarzaj:
  - jeśli $a_k > a_{max}$ to $max := k$;
  - $k := k+1$;
- {zamieniamy miejscami a_i z a_{max}}
- $x := a_{max}$; $a_{max} := a_i$; $a_i := x$;
- $i := i-1$;
- {teraz mamy następującą sytuację $a_1, \ldots, a_{i-1}, a_i \leqslant a_{i+1} \leqslant \ldots \leqslant a_n$}

$i=n=5$	1 5 4 2 3	max przyjmuje kolejno wartości 1,2
$i=4$	1 3 4 2 5	max przyjmuje kolejno wartości 1,2,3
$i=3$	1 3 2 4 5	max przyjmuje kolejno wartości 1,2
$i=2$	1 2 3 4 5	max przyjmuje kolejno wartości 1,2
$i=1$	koniec działania	

Obok pokazujemy, jak działa ten algorytm dla ciągu 1, 5, 4, 2, 3.

Poszukajmy funkcji złożoności tego algorytmu. Zauważmy, że dla ustalonej długości ciągu liczba porównań będzie zawsze taka sama. Zatem

$$T_{pes} = T_{ave} = \sum_{i=2}^{n}\sum_{k=2}^{i} 1 = \sum_{i=2}^{n}(i-1) = \sum_{i=1}^{n-1} i = \frac{1+n-1}{2}\cdot(n-1) = \frac{n^2}{2} - \frac{n}{2}.$$

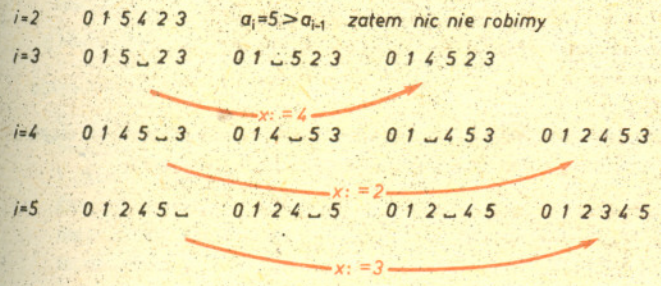
Zatem funkcja złożoności czasowej (zarówno średniej, jak i pesymistycznej) jest rzędu n^2.

Aby poprawić średnią złożoność czasową, postarajmy się znaleźć taki algorytm, który w przypadku, gdy dany ciąg jest już częściowo posortowany, będzie wykonywał mniej kroków. Zastosujmy metodę podobną jak przy układaniu kart do gry. Każdą kolejną kartę wkładamy między karty już wcześniej ułożone.

Załóżmy, że ciąg $a_1, a_2, \ldots, a_{i-1}$ jest posortowany i chcemy wstawić i-ty element we właściwe miejsce. W tym celu przeglądamy ciąg $a_1, \ldots, a_{i-1}$ od końca aż do napotkania pierwszego elementu mniejszego od elementu, który chcemy wstawić. Aby taki element zawsze się znalazł, dokładamy na początku ciągu element $a_0 = 0$. Za tak znaleziony element wstawiamy element i-ty. Otrzymany ciąg $a_1, a_2, \ldots, a_{i-1}, a_i$ jest posortowany i o ile $i < n$ możemy czynność powtórzyć.

Algorytm 2: Sortowanie przez wstawianie

```
i := 2;
dopóki i ⩽ n powtarzaj:
  {a0 ⩽ a1 ⩽ ... ⩽ a(i-1)}
  jeśli [a(i-1) > ai] to
    {a0 ⩽ a1 ⩽ ... ⩽ a(i-1) i a(i-1) > ai, wstawiamy i-ty element}
    x := ai;
    j := i-1;
    dopóki [aj ⩾ x] powtarzaj:
      a(j+1) := aj;
      {przesuwamy element aj o jedno miejsce w prawo}
      j := j-1;
    {aj < x i wszystkie elementy większe od x zostały przesunięte
     o jeden w prawo}
    a(j+1) := x;
  i := i+1;
```



Obok pokazujemy, jak będzie wyglądało takie sortowanie ciągu 1, 5, 4, 2, 3.

Zostały przyjęte następujące reguły zapisu algorytmów:

1. Instrukcję „x otrzymuje wartość y", zwaną w informatyce instrukcją podstawienia, zapisujemy symbolicznie $x := y$.
2. Poszczególne instrukcje oddzielamy średnikami.
3. Klamrą [obejmujemy ciąg instrukcji, które chcemy traktować jako jedną całość. Klamra ta będzie najczęściej obejmować ciąg instrukcji, który chcemy powtórzyć pewną liczbę razy lub też wykonać, jeśli spełniony jest określony warunek.
4. W nawiasach klamrowych { } umieszczamy komentarze wyjaśniające poszczególne fragmenty algorytmu.
5. Operacje dominujące będziemy obwodzić ramką.

Łatwo zauważyć, że najwięcej porównań trzeba będzie wykonać w przypadku, gdy ciąg jest posortowany w odwrotnej kolejności.

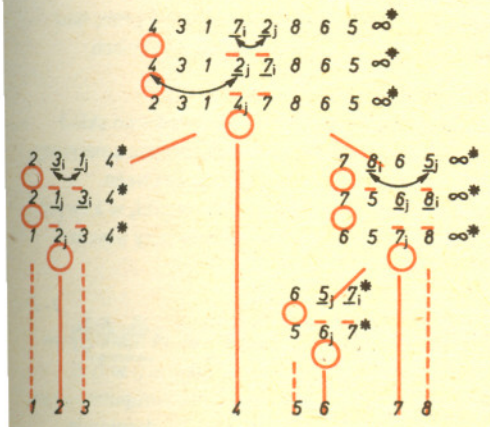
$$T_{pes} = \sum_{i=2}^{n}\left(1+\sum_{j=0}^{i-2}1\right) = \sum_{i=2}^{n}(1+i-1) = \frac{2+n}{2}(n-1) = \frac{1}{2}n^2+\frac{1}{2}n-1.$$

T_{pes} jest więc analogicznie jak w przypadku pierwszego algorytmu rzędu n^2.

Analiza kosztu oczekiwanego (wymagająca trochę dłuższych rachunków) prowadzi do wyniku $T_{ave} = \frac{n^2}{4} - \frac{n}{4}$, co też jest niestety funkcją rzędu n^2. Zastosowane usprawnienie nie okazało się więc wystarczające. Poprawę oczekiwanej złożoności czasowej (przy złożoności pesymistycznej takiej jak poprzednio) uzyskujemy stosując algorytm zwany sortowaniem szybkim. Algorytm ten opiera się na następującym pomyśle. Poprzestawiajmy elementy ciągu tak, aby po lewej stronie znalazły się wszystkie elementy mniejsze od losowo wybranego elementu v, po prawej stronie wszystkie od niego większe, i niech element v oddziela obie te grupy. Otrzymamy następujący ciąg $a_1, \ldots, a_{j-1}, v, a_{j+1}, \ldots, a_n$. Opisaną wyżej operację powtarzamy teraz dla każdego z ciągów $a_1, \ldots, a_{j-1}$ oraz $a_{j+1}, \ldots, a_n$, który ma długość większą od jedności.

Algorytm szybkiego sortowania jest kolejnym przykładem algorytmu rekurencyjnego. Tego typu algorytm wraz z szerszym omówieniem zamieściliśmy w *Delcie* 12/1984.

W podanym niżej algorytmie wybieramy jako v pierwszy element podciągu. Do ciągu dodajemy dodatkowy element a_{n+1} większy od wszystkich elementów ciągu ($a_{n+1} = \infty$). Pełni on rolę „strażnika". Sprawdźcie sami, co by się stało, gdyby pierwszy element był największy w ciągu i nie byłoby takiego strażnika.



Algorytm 3: Sortowanie szybkie

```
parametry l: indeks początku sortowanego podciągu
          p: indeks końca sortowanego podciągu
v := ai; i := l; j := p+1; {ak < a(p+1) dla k = l, ..., p}
dopóki i < j powtarzaj {podział na podciągi}
  dopóki ai ⩽ v powtarzaj i := i+1; {ai jest po dobrej stronie}
  dopóki aj > v powtarzaj j := j-1; {aj jest po dobrej stronie}
  jeśli i < j to x := ai; ai := aj; aj := x; {zatrzymaliśmy się
    na takich ai oraz aj, które nie były po dobrych stronach,
    zamieniamy je miejscami}
ai := aj; aj := v {wstawiamy v między podciągi}
jeśli l < j-1 to wykonaj sortowanie szybkie startując od l = l
    i p = j;
jeśli p > j+1 to wykonaj sortowanie szybkie startując od l = j+1
    i p = p;
```

Aby posortować ciąg, należy wystartować od $l = 1$ i $p = n$.

Obok przykład działania algorytmu dla ciągu 4, 3, 1, 7, 2, 8, 6, 5. Element oznaczony * pełni rolę strażnika w przypadku, gdy a_l jest największym elementem podciągu.

W pesymistycznym przypadku (tu będzie to np. ciąg posortowany) liczba porównań będzie znowu rzędu n^2. Zauważmy, że suma wszystkich porównań wykonywanych na jednym „poziomie" jest nie większa niż n. Gdyby więc przy każdym podziale na część „lewą" i „prawą" otrzymane części były równe, to liczba porównań byłaby rzędu $n\log_2 n$. Okazuje się jednak, że w przypadku tego algorytmu oczekiwana złożoność czasowa nie odbiega wiele od tego korzystnego przypadku i jest rzędu $n\log_2 n$. Jest to więc niewątpliwy postęp w stosunku do poprzednich algorytmów.

Istnieją inne algorytmy umożliwiające posortowanie ciągu ze średnią złożonością czasową tego samego rzędu co algorytm sortowania szybkiego. Interesujące wydaje się być pytanie, czy da się to zrobić jeszcze szybciej. Okazuje się, że nie. Jeśli kolejność elementów wyznaczamy na podstawie porównywania tych elementów, nie da się poprawić rzędu funkcji oczekiwanej złożoności czasowej.

Zadanie. Sortujemy ciąg $a_1, \ldots, a_n$ w następujący sposób: generujemy wszystkie możliwe permutacje ciągu $a_1, \ldots, a_n$ i dla każdej z nich sprawdzamy, czy tworzy ona ciąg posortowany.

Uzasadnić, iż średnia liczba porównań jest co najmniej $\frac{n!}{2}$. Zakładając, że maszyna wykonuje 10^5 operacji na sekundę obliczyć, ile co najmniej czasu średnio potrzeba na posortowanie ciągu długości 15 opisaną wyżej metodą, a ile za pomocą algorytmu szybkiego sortowania.

Komputer zamiast zecera

Zawody rodzą się i umierają wraz z rozwojem techniki. Nie ma już dorożkarzy, zapalaczy latarni gazowych i wielu innych zawodów. Także rozwój komputerów eliminuje niektóre grupy zawodowe. Nie każdy jednak wie, że największą taką grupą (przynajmniej w krajach wysoko rozwiniętych) są ... zecerzy.

Od czasów Gutenberga do połowy XX wieku technika druku bardzo się rozwinęła. Ręczne składanie kolejnych stron z drewnianych czcionek zastąpiono odlewaniem całych wierszy w specjalnych urządzeniach, zwanych linotypami. Ich obsługiwanie było właśnie zajęciem zecerów. Praca zecera odbywała się w bardzo trudnych warunkach. Linotypy były bardzo hałaśliwe, a pary gorącego stopu ołowiowo-antymonowego zatruwały pracujących. Nic więc dziwnego, że — gdy tylko było to możliwe — wprowadzono tzw. skład komputerowy.

Cały tekst artykułu gazetowego albo książki jest przygotowywany na komputerze. Na ekranie widać dokładny obraz tego tekstu, z uwzględnieniem różnych krojów czcionki, zmiennej szerokości znaków (porównajcie długość słów *iiiii* oraz *mmmmm*), wytłuszczeń, kroju pochyłego (kursywy) itd.

Komputer automatycznie dzieli tekst na wiersze i kolumny, a następnie zapisuje na nośniku magnetycznym (np. na dysku elastycznym — od 100 do 500 stron maszynopisu na krążku o średnicy do 20 cm i grubości dwóch okładek *Delty*). Dalsze etapy przygotowania matryc do drukowania przebiegają już bez bezpośredniego udziału człowieka.

Poza likwidacją miejsc pracy szkodliwych dla zdrowia skład komputerowy przyniósł także inne korzyści. Przy przygotowywaniu wydania gazety problemem było zawsze tzw. łamanie numeru, tzn. podział posiadanego materiału na szpalty i kolumny, rozmieszczenie, wielkość i krój tytułów, zdjęć itp. Przy składzie ręcznym wymagało to fizycznego manipulowania matrycami, składania pojedynczych wierszy w szpalty, układania kolumn ze szpalt, ręcznego układania tytułów z dużych czcionek i wielu jeszcze innych czasochłonnych zajęć. W praktyce redaktor wydania mógł wypróbować maksymalnie 3—4 różne wersje rozmieszczenia materiału na stronach. Poza tym częste były przypadki przemieszania wierszy (albo zamiany wierszy w sąsiednich szpaltach).

Przy składzie komputerowym redaktor może zażądać, aby komputer przesunął artykuł na wskazaną kolumnę, aby zmienił krój czcionki tytułu i odpowiednio przemieścił pozostałe artykuły. Wynik jest natychmiast widoczny na ekranie. Pozwala to wypróbować nawet kilkadziesiąt wariantów złamania numeru przed wyborem najlepszego.

Jeszcze większe korzyści daje skład komputerowy przy druku książek. W dotychczasowym systemie autor pisał rękopis, na który nanosił wielokrotnie poprawki, maszynistka przepisywała rękopis, a maszynopis był przekazywany do redakcji, po recenzjach na maszynopis były nanoszone poprawki i uwagi techniczne (krój czcionki itp.), wreszcie zecer składał matryce książki. Tak długa droga powodowała powstawanie licznych błędów. Wprowadzanie poprawek do już złożonego tekstu wymagało ponownego złożenia obszernego fragmentu tekstu (często pojawiały się nowe błędy), a wznowienie lub wydanie w innym formacie — na ogół złożenia całości od nowa.

Przy składzie komputerowym cały tekst książki jest przygotowywany od razu za pomocą komputera. Tekst, zapamiętany na nośnikach magnetycznych, może być łatwo i szybko modyfikowany zgodnie z sugestiami recenzentów i redakcji. Nikt poza autorem nie pisze ani nie przepisuje tekstu (o ile tekst został zapisany na komputerze), a więc nie ma obawy o powstanie błędów wynikających z niezrozumienia zawartości książki. Wprowadzanie zmian jest możliwe i łatwe jeszcze na moment przed oddaniem książki do druku. Automatyczna produkcja matryc przyspiesza także pracę nad książką w samej drukarni. A przy ponownych wydaniach wystarczy wyjąć z szuflady dysk elastyczny ...

A co z zecerami tam, gdzie powszechnie wprowadzono skład komputerowy? Część przeszła na renty i wcześniejsze emerytury, pozostali obsługują maszyny do składu komputerowego. Kilkudziesięciu pracuje w różnych muzeach techniki, w których składa się drobne ulotki i programy zwiedzania przy użyciu muzealnych już dziś linotypów.

J. D.

Od redakcji: *Delta* jest składana przy użyciu lino i monotypów oraz (w trudniejszych miejscach) ręcznie. Drukarnie, w tym i nasza, są wyposażane w urządzenia do składu komputerowego, jednak redakcje (nawet redakcja *Informatyki*) nie dysponują możliwością zapisywania tekstu na komputerze. Nie mówiąc już o autorach, którzy rzadko mają dostęp choćby do mikrokomputera, zupełnie zresztą w tej sprawie nieprzydatnego.

Komputer zamiast krawca

„Murarz domy buduje, krawiec szyje ubrania ..."
Tak zaczyna się wiersz dla dzieci napisany przez Juliana Tuwima w połowie dwudziestego wieku. Teraz, pod koniec tego samego stulecia okazuje się, że aktualną jeszcze trzydzieści lat temu wyliczankę należy uzupełnić o jeszcze jeden zawód — zawód informatyka.
W przedsiębiorstwie zajmującym się produkcją odzieży nowy model spodni, marynarki czy kurtki powstaje najpierw w postaci rysunku w pracowni projektanta. Potem rysunek taki trafia w ręce konstruktorów. To oni zgodnie ze ściśle skodyfikowanymi zasadami, popartymi własnym doświadczeniem, rozrysują projekt na wykroje. W pracowniach konstruktorów kończy się twórczy wkład człowieka w nowy model zdobiący potem manekiny domów towarowych. Stąd szablony wykrojów trafią na stoły krojcze, a potem pocięty materiał, zszyty w szwalni pojedzie do magazynów wysyłkowych.
I gdzie tutaj komputer?
Oto jak współcześnie pracuje konstruktor odzieży. Gdy trafia do niego projekt plastyka, rysuje on wykroje na graficznym monitorze ekranowym komputera. Maszyna pomaga mu przy tym nie tylko jako „inteligentny kreślarz" — rysując żądane figury geometryczne, mierząc odległości i kąty — ale podpowiada także gotowe rozwiązania. W typowych przypadkach prowadzi wręcz konstruktora za rękę. Klasyczne dżinsy konstruuje się przecież od kilkudziesięciu lat tak samo i nie warto za każdym razem od nowa odkrywać tajemnic karczka czy „wędrujących" zaszewek. Podstawę nowego modelu stanowi więc z reguły znany algorytm. Tak jest w najprostszym przypadku, kiedy konstruktor korzysta z gotowych algorytmów i bazy danych standardowych wykrojów. Gdy jednak projekt jest rzeczywiście oryginalny, trzeba tworzyć kształty wykrojów od nowa z pojedynczych linii rysowanych na ekranie piórem świetlnym. I w tym przypadku komputer służy konstruktorowi pomocą. Wygładza rysowane ręcznie kreski, sprawdza ich wzajemne położenie, a nawet weryfikuje poprawność konstrukcji, ostrzegając na przykład, że tak zaprojektowany rękaw w żaden sposób nie da się wszyć w przewidziane dla niego miejsce.
Komputer przeskalowuje też wykroje do wszystkich wymaganych wymiarów. Konstruktorzy projektują ubrania dla wymiaru wzorcowego. W trakcie rysowania zaznaczają na rysunkach punkty charakterystyczne. Posłużą one potem do powielania wykrojów. Jedne z nich odpowiadają wymiarom charakterystycznym dla różnych sylwetek. Inne niezbędne są do tego, by po przeskalowaniu wykroje dawały się zszyć. Istnieje wiele sposobów doboru punktów charakterystycznych i wiele algorytmów skalowania. To one właśnie decydują o dopasowaniu odzieży do różnych sylwetek, często więc stanowią pilnie strzeżoną tajemnicę firmową. By przekonać się, że problem przeskalowania odzieży wcale nie jest prosty, wystarczy uświadomić sobie, że przyroda powiela sylwetki ludzkie bynajmniej nie przez jednokładność, a stosowane przez nią reguły nie są wcale trywialne.
Przygotowane przez konstruktorów wykroje trafiają potem do działu przygotowania produkcji. Tutaj dba się o to, by jak najekonomiczniej wykorzystać materiał. Komputer wykona to zadanie najsprawniej. Z zaproponowanego przez niego rozkładu wykrojów na materiale pozostanie najmniej ścinków. I nie dość na tym, w gotowym ubraniu krata dobrze zbiegnie się w szwach, a części, z których zrobiony będzie aksamitny żakiet, ułożone będą w tę samą stronę.

K. B.

Zadania

Redaguje mgr Witold MARCISZEWSKI

M 409. Na bokach BC i CD kwadratu $ABCD$ wybrano punkty M i N tak, że $CM+CN = AB$. Proste AM i AN dzielą przekątną BD na trzy odcinki. Wykazać, że jeden z kątów trójkąta zbudowanego z tych odcinków jest równy $\frac{\pi}{3}$.

Rozwiązanie na str. 2

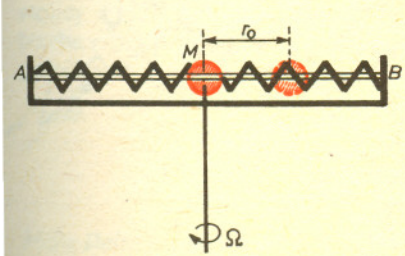
M 410. Tablica $n \times n$ liczb całkowitych nieujemnych jest zbudowana w ten sposób, że jeśli na przecięciu i-tego wiersza i j-tej kolumny znajduje się 0, to suma liczb w tym wierszu i tej kolumnie jest nie mniejsza niż n. Udowodnić, że suma wszystkich liczb w tablicy jest nie mniejsza niż $\frac{n^2}{2}$.

Rozwiązanie na str. 12

M 411. Na płaszczyźnie dany jest skończony zbiór wielokątów $W_1, W_2, \ldots, W_n$, z którego każde dwa mają punkt wspólny. Udowodnić, że istnieje prosta, która przecina wszystkie wielokąty.
Rozwiązanie na str. 12

Redagują mgr Tomasz TRATKIEWICZ i mgr Włodzimierz ZIELICZ

F 180. W przedstawionym na rysunku układzie kulka o masie M zamocowana jest między jednakowymi nieważkimi sprężynami o stałych sprężystości $k/2$ i może ślizgać się bez tarcia wzdłuż poziomego pręta AB. Układ obraca się ze stałą prędkością kątową Ω wokół pionowej osi biegnącej przez środek pręta. Ile wynosi prędkość kątowa Ω, jeśli kulka w trakcie wirowania utrzymuje się w stałej, niezerowej odległości r_0 od osi obrotu?



Rozwiązanie na str. 3

F 181. Kulkę z urządzenia opisanego w poprzednim zadaniu przymocowano w odległości R od osi obrotu, układ rozpędzono do prędkości kątowej ω_0 i odłączono napęd. Jaki będzie ruch kulki po jej uwolnieniu (może poruszać się wzdłuż pręta)? Opór powietrza i tarcie w łożyskach należy zaniedbać.
Rozwiązanie na str. 11

Którędy do granicy?

Potężny władca dalekiego kraju postanowił dowiedzieć się, ile co najmniej dni potrzeba żołnierzowi, aby dotrzeć ze stolicy do granicy państwa, poruszając się wzdłuż istniejących gościńców. Gościńców tych nie było zresztą wiele — łączyły one miasta i nie krzyżowały się poza nimi. Udzielenie odpowiedzi nie było łatwe. Wiadomo było, ile czasu wymaga dojście do rozmaitych punktów granicy różnymi drogami. Nikt jednak nie wiedział na pewno, czy nie istnieją inne, krótsze drogi. Dopiero Naczelny Dowódca Armii zaproponował następujące rozwiązanie:

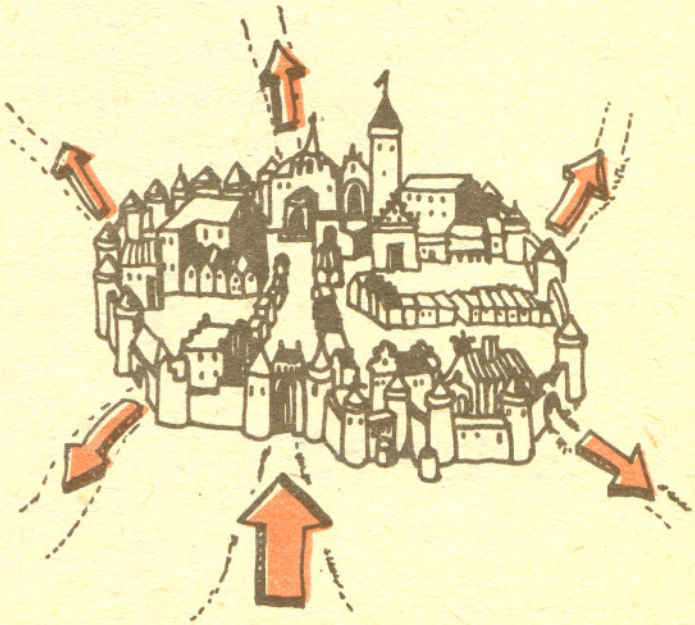
1. Należy zmobilizować wielką armię — na tyle dużą, aby można ją w nieskończoność dzielić na coraz mniejsze oddziały.

2. Należy równocześnie rozesłać oddziały wszystkimi gościńcami wychodzącymi ze stolicy.

3. Każdy oddział po dojściu do miasta, które jeszcze nie spłonęło, powinien je natychmiast spalić, a następnie podzielić się na mniejsze oddziały, wyruszające wszystkimi gościńcami (z wyjątkiem gościńca, po którym oddział przybył do miasta).

4. Po dojściu do spalonego miasta cały oddział ma popełnić zbiorowe samobójstwo.

5. Po dojściu do granicy oddział powinien wrócić do stolicy tą samą drogą, którą z niej szedł (oczywiście ignorując dopalające się miasta).



Trasa oddziału, który najszybciej powróci do stolicy, będzie najkrótszą możliwą drogą łączącą stolicę z granicą. Czas marszu oddziału będzie dwukrotnie dłuższy od czasu, o który pytał Król.

Naczelny Dowódca sformułował w ten sposób pewien algorytm równoległy, tzn. taki, w którym różne kroki są wykonywane równocześnie (w każdej chwili różne oddziały podróżują po różnych gościńcach, a niektóre, być może, palą miasta). „Czas pracy" podanego algorytmu jest równy podwojonemu czasowi marszu ze stolicy do granicy.

Algorytm Dowódcy Armii ma kilka wad, m.in. kończy się on całkowitym zniszczeniem kraju (czy potraficie powiedzieć, pod jakim warunkiem ocaleje stolica?). Załóżmy, że Naczelny Geograf Królestwa ma mapę, na której są zaznaczone wszystkie gościńce i czasy ich przebycia. Spróbujcie opisać, jak — posługując się taką mapą — Geograf może odpowiedzieć na pytanie Króla.

Wskazówka
Przypuśćmy, że wiemy, iż do miasta A można dojść ze stolicy w ciągu co najmniej a dni. Ile co najwyżej dni trzeba iść ze stolicy do miasta B, połączonego z A gościńcem, którego przejście trwa x dni?

Małą Deltę przygotował Jarosław DEMINET

ELIZA—jak człowiek może rozmawiać z komputerem

Mgr Małgorzata NALBACH, mgr Krzysztof STUDZIŃSKI

Dzień dobry. Co cię do mnie sprowadza?
Moja mama ciągle na mnie krzyczy —
Opowiedz mi więcej o swojej rodzinie.
Mój tata każe mi się ciągle uczyć —
Kto jeszcze w twojej rodzinie każe ci się ciągle uczyć?
Wszyscy. Mówią, że **jeśli** się nie będę uczyć, to do niczego nie dojdę —
Uzasadnij to.
To oni **tak** mówią —
Dlaczego tak uważasz?
Zadajesz niemądre pytania. Przecież słyszałam —
Nie jestem pewna, czy cię całkowicie rozumiem.
Jesteś taka jak oni —
Jakie związki tu widzisz?
Ty mnie też **nie** rozumiesz. Nie można życia spędzić nad książkami —
Uzasadnij to.
Powtarzasz się **jak** głupia maszyna —
Czy istotnie jest tu jakieś podobieństwo?
Wydaje się, że tak —
Czy jesteś pewna?
Przepraszam za to porównanie —
Usprawiedliwienia nie są potrzebne.
Oni **zawsze** każą przepraszać —
Czy możesz podać konkretny przykład?

Tak rozmawiał pewien człowiek z komputerem. Spróbuj dokładnie prześledzić tok rozmowy; być może zauważysz bardzo ciekawe rzeczy.

Od początku historii komputerów ludzi denerwowały trudności, które musieli pokonywać, żeby maszyna cyfrowa wykonywała ich polecenia. Komputery bowiem są w stanie zrozumieć tylko bardzo precyzyjnie sformułowane polecenia wyrażone w złożonym i sformalizowanym języku programowania. Ograniczało to liczbę ludzi korzystających z komputerów do tylko tych, którzy opanowali pewien język programowania, a więc spełnili wymagania stawiane przez maszynę. Czyli coś, co ma służyć człowiekowi, stawia mu duże wymagania.

Ideałem byłoby, gdyby z komputerem można było porozumieć się tak naturalnie jak podczas wymiany informacji między ludźmi. Tego typu idee ludzie mieli już od dawna. Na pewno znane są powieści fantastyczno-naukowe, gdzie pojawiają się motywy rozmowy z komputerem czy robotem.

Już w roku 1966 Weisenbaum zaprojektował i uruchomił system ELIZA. Nazwa programu pochodzi od imienia głównej bohaterki „Pigmaliona" G. B. Shawa. Założeniem programu ELIZA jest podtrzymywanie dialogu, to znaczy zachowywanie się tak, jak zachowuje się dobrze wychowana osoba z uwagą słuchająca rozmówcy i zachęcająca go do dalszych zwierzeń. Program ten został zaprojektowany w oparciu o metodę słów kluczowych. Maszyna zaprogramowana jest w taki sposób, że z wprowadzanych do niej słów i zwrotów wyszukuje te, które potrafi zidentyfikować i na ich podstawie tworzy odpowiedź. O rodzaju odpowiedzi i temacie rozmowy decyduje zasób słów kluczowych obrany przy projektowaniu programu. Oryginalny program ELIZA prowadził rozmowę w języku angielskim.

W roku 1981 w Instytucie Informatyki Uniwersytetu Warszawskiego powstał analogiczny program pozwalający na rozmowy w języku polskim. Poniżej przedstawimy zasady, w oparciu o które działa taki program.

Podstawową częścią programu ELIZA jest automat złożony z pewnej liczby stanów. Z każdym stanem związane są następujące elementy:
1. nazwa stanu, która pozwala odróżnić go od innych,
2. akcja podejmowana przy wejściu do tego stanu,
3. tekst wypisywany w tym stanie,
4. liczba odpowiedzi w tym stanie. Jeśli liczba odpowiedzi jest równa 0, to stan taki jest stanem

Rozwiązanie zadania o sortowaniu.

Oszacowanie $\frac{n!}{2}$ otrzymujemy zakładając, że do sprawdzenia, czy ciąg jest posortowany, musimy wykonać co najmniej jedno porównanie.
Czas sortowania opisanym algorytmem — więcej niż 75 dób.
Czas sortowania algorytmem 3 — około $6 \cdot 10^{-4}$s.

W 1939 roku A. M. Turing sformułował tezę, że można uważać maszynę za „myślącą", kiedy człowiek, przebywający w innym pokoju niż maszyna i prowadzący z nią rozmowę, nie będzie mógł poznać, czy rozmawia z człowiekiem czy maszyną. ELIZA miała dowieść, że teza ta jest fałszywa: nawet prosty i na pewno bezmyślny program może udawać człowieka wcale sugestywnie. Podobno jeden z prawdziwych („żywych") psychiatrów rozmawiał przez kilkanaście minut z ELIZĄ (za pośrednictwem dalekopisu) przekonany, że rozmawia ze swoją koleżanką po fachu. Powiedział nawet „na przykład żadna maszyna nie potrafiłaby rozmawiać tak, jak Pani". A problem, co to znaczy „myśląca maszyna" nadal jest otwarty.

Rozwiązanie zadania F 181. Zasada zachowania momentu pędu pozwala wyznaczyć prędkość kątową obrotu ω (po odłączeniu napędu) jako funkcję odległości kulki od osi obrotu, mamy bowiem

$$L = (J+Mr^2)\omega = (J+Mr_0^2)\,\omega_0,$$

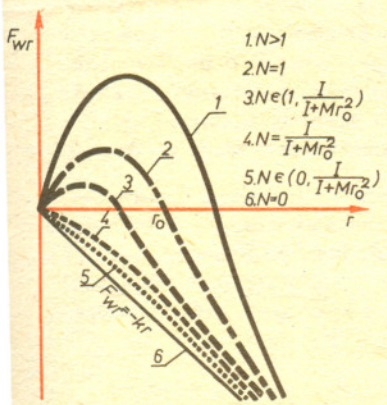
gdzie L — momentu pędu, a J — moment bezwładności wirującej części układu. Działające siły mają postać analogiczną jak w poprzednim zadaniu, składowa siły wypadkowej równoległa do pręta wynosi więc

$$F = (M\omega^2 - k)r = \left[M\left(\frac{L}{J+Mr^2}\right)^2 - k\right]r.$$

Oznaczając $\Omega = \sqrt{\frac{k}{M}}$ i $\omega_0 = N\Omega$ mamy

$$F = M\Omega^2\left[N^2\left(\frac{J+Mr_0^2}{J+Mr^2}\right)^2 - 1\right]r.$$

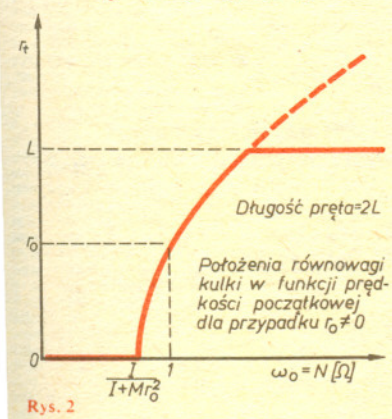
Szkicowe wykresy zależności F od r dla różnych wartości parametru N przedstawia rysunek 1. Od wartości tego parametru zależy położenie miejsc zerowych funkcji (położenia równowagi) i znaki pochodnych w tych punktach (charakter równowagi).



Rys. 1

Po uzyskaniu swobody ruchu kulka wykonuje drgania wokół położenia równowagi trwałej wyznaczonego przez początkową prędkość kątową układu. Położenia równowagi dla $r_0 \neq 0$ przedstawia rysunek 2. Przy małych prędkościach początkowych położenie równowagi r_e znajduje się na osi obrotu, przy większych ω mamy

$$r_e = \sqrt{J\cdot(N-1)/M + Nr_0^2}.$$



Rys. 2

Oczywiście r_e może być mniejsze, równe lub większe niż r_0 i od zachodzącej relacji zależy, w którą stronę zacznie swój ruch kulka. Gdy $r_e = r_0$, kulka nie zmieni swego położenia, choć odłączenie napędu zmieni równowagę z obojętnej na trwałą.

końcowym — na nim kończy się rozmowa. Jeśli liczba odpowiedzi jest równa 1, to z danego stanu następuje bezpośrednie przejście do stanu wskazanego jako następny. Przy liczbie odpowiedzi większej od 1 program próbuje dopasować wyraz wejściowy do jednej z przewidzianych w tym stanie odpowiedzi użytkownika. Kiedy mu się to udaje, następuje przejście do wskazanego dla tej odpowiedzi stanu.

Ponieważ każdy stan jest opisywany w ten sam sposób, cały automat można przedstawić w postaci tabelki, gdzie jeden wiersz opisuje stan. Wiersz taki zawiera następujące informacje:
— nazwa stanu,
— akcje,
— tekst,
— liczba odpowiedzi,
— nazwy stanów docelowych.

Oto fragment automatu, który działa po rozpoznaniu w zdaniu wejściowym jednego ze słów rozumianych przez ELIZĘ jako porównanie, a więc: podobnie, jak, podobny, jednakowy, tacy, taki. Wyrazy odmienne podawane są we wszystkich formach odmiany, w jakich mogą wystąpić.

stan	akcje	tekst	l odp.	odp.	do
sjak	0	—	4	nazw./nazywa./ —/.	nazywa/nazywa/ enjak/sjak
enjak	2	—	1	—	s
nazywa	0	—	2	—/.	ennazywa/nazywa
ennazywa	1	—	1	—	s

W kolumnie AKCJE występują tu numery akcji, jakie należy wykonać po wejściu do danego stanu. Liczba 0 oznacza, że żadna akcja nie będzie podejmowana. W automacie realizującym ELIZĘ akcje służą do generowania i wypisywania odpowiedzi ELIZY. W innym miejscu pamięci komputera jest przechowywana lista możliwych odpowiedzi. Na każde słowo kluczowe ELIZA może odpowiedzieć na kilka możliwych sposobów. I tak na słowo z podanej już grupy porównań ELIZA podaje, realizując w ten sposób akcję numer 2, jedną z następujących odpowiedzi:
Na czym polega to podobieństwo?
Jakie związki tu widzisz?
Czy istotnie jest tu jakieś podobieństwo?
Na co wskazuje to podobieństwo?

Odpowiedzi te podawane są kolejno. Pozwala to na urozmaicenie dialogu, co powoduje, że ELIZA zbyt często się nie powtarza. Myślnik w kolumnie TEKST oznacza, że żaden tekst nie będzie wypisywany. W tym bowiem przypadku teksty, a więc komentarze lub odpowiedzi programu są generowane, a nie wypisywane bezpośrednio. Gdyby zostały podane w tym miejscu, w każdej sytuacji możliwa by była tylko jedna odpowiedź programu.

Kolumna LODP zawiera liczbę odpowiedzi, a jej znaczenie zostało wcześniej wyjaśnione. W stanach ENJAK oraz ENNAZYWA ELIZA udziela odpowiedzi, po czym przechodzimy do stanu S, w którym jest wczytywane następne zdanie rozmówcy i rozpoczyna się proces rozpoznawania kolejnych wyrazów tego zdania.

W kolumnie ODP umieszczone są oczekiwane w tym stanie wyrazy lub zwroty. Kolejne wyrazy oddzielone są znakiem /. Podawane są tylko tematy wyrazów, końcówki zastępuje kropka. Kropka pasuje do wszystkiego, czyli jeśli jakiś wyraz jest zgodny z podanym tu, na wszystkich literach aż do kropki, przyjmuje się, że jest zgodny do końca. W szczególności sama kropka oznacza dowolny wyraz. I tak, na przykład do wzorca **nazw.** mogą zostać dopasowane następujące wyrazy (o ile wystąpią w zdaniu wejściowym):

nazwisko, nazwiska, nazwa, nazwę itd.

Pozwala to na pominięcie problemów związanych z odmianą wyrazu. Wystąpienie tematu wyrazu jest równoznaczne z wystąpieniem całego wyrazu. Jednocześnie widać, że tak rozumiane tematy wyrazów należy wyodrębniać ze szczególną ostrożnością. Źle wybrany temat może prowadzić do błędnych odpowiedzi.

Każde zdanie rozmówcy ELIZY kończy myślnik. Dopiero po rozpoznaniu końca zdania automat udziela odpowiedzi.

W każdym zdaniu może wystąpić 0, 1 lub więcej słów kluczowych. Jeśli nie ma żadnego słowa kluczowego, zdanie jest dla ELIZY niezrozumiałe i podawana jest jedna ze standardowych dla tej sytuacji odpowiedzi:

Nie jestem pewna, czy cię całkowicie rozumiem.
Proszę mów dalej.
Co twoim zdaniem z tego wynika.
Czy cię denerwuje mówienie o tych sprawach?

W przypadku, gdy w zdaniu wystąpi kilka słów kluczowych, jest wybierane słowo uznawane za najistotniejsze, czyli to, które ma najwyższy priorytet. W każdym stanie słowa kluczowe w kolumnie ODP uporządkowane są według priorytetów, od największego do najmniejszego. Ponadto w stanie, który został osiągnięty po rozpoznaniu jednego ze słów kluczowych, sprawdzane jest, czy dalej w zdaniu nie wystąpiło przypadkiem słowo kluczowe o wyższym od tego priorytecie i jeśli tak, następuje przejście do stanu odpowiadającego wystąpieniu tego słowa. W naszym przykładzie widać, że słowa kluczowe oznaczające porównanie mają bardzo wysoki priorytet. Wyższy od nich mają tylko słowa oznaczające nazwy lub nazwiska. W podanym na początku przykładzie dialogu zostały wyróżnione słowa kluczowe, na które ELIZA reagowała udzielając stosownych odpowiedzi.

Można również zaobserwować jeszcze jedną cechę ELIZY. Otóż wszystkie wyrazy bliskoznaczne traktuje ona w ten sam sposób. Bez względu na to czy w zdaniu wystąpi wyraz jak, jednakowy, czy taki, odpowiedź będzie taka sama.

Kolumna DO zawiera nazwy stanów, do których następuje przejście po rozpoznaniu kolejnego słowa ze zdania użytkownika. W kolumnie DO musi być tyle samo elementów co w kolumnie ODP. Jeśli zostanie rozpoznana pierwsza z listy odpowiedzi w kolumnie ODP, następuje przejście do pierwszego stanu z listy stanów w kolumnie DO, przy drugiej do drugiego, itd. W naszym przykładzie po rozpoznaniu wzorca **nazw.** lub **nazywa.** następuje przejście do stanu NAZYWA, po rozpoznaniu końca zdania (znak −) do stanu ENJAK, w pozostałych przypadkach sterowanie pozostaje w stanie SJAK i jest sprawdzany następny wyraz ze zdania użytkownika.

Na przykładzie programu ELIZA widać, że duże efekty można czasem osiągnąć bardzo prostymi środkami. Omawiany program konwersacyjny nie zmusza użytkownika do korzystania z ograniczonego podzbioru języka, który przyjęto określając strukturę programu, ale raczej nakłania go do zachowania takiej swobody wypowiedzi, z jakiej można korzystać w codziennej rozmowie. ELIZĘ zaprogramowano w taki sposób, aby ukryć przed użytkownikiem przypadki, w których program go nie rozumie.

Czy algorytmy podawane w szkole są poprawne?

Przypuśćmy, że w zadaniu z fizyki potrzebne jest wyliczenie wartości

$$y_1 = \frac{1}{x_1}, \quad y_2 = \frac{1}{x_2},$$

gdzie x_1 i x_2 są pierwiastkami równania kwadratowego

$$x^2 - 2 \cdot 10^5 x + 2 = 0.$$

Rozwiązanie wydaje się proste: współczynniki mają niewiele cyfr znaczących i można przypuszczać, że kalkulator liczący z dokładnością do 8 cyfr znaczących powinien wystarczyć. Spróbujmy:

$$\Delta = (2 \cdot 10^5)^2 - 4 \cdot 2 \approx 4 \cdot 10^{10}, \quad \sqrt{\Delta} \approx 2 \cdot 10^5,$$

$$x_1 \approx \frac{2 \cdot 10^5 - 2 \cdot 10^5}{2} = 0, \qquad x_2 \approx \frac{2 \cdot 10^5 + 2 \cdot 10^5}{2} = 2 \cdot 10^5,$$

y_1 nie istnieje, $\qquad y_2 = 5 \cdot 10^{-6}$.

W rzeczywistości

$$x_1 \approx 10^{-5}, \qquad x_2 \approx 2 \cdot 10^5 - 10^{-5},$$

$$y_1 \approx 10^5, \qquad y_2 \approx 5 \cdot 10^{-6}(1 + 5 \cdot 10^{-11}),$$

a więc można uznać, że x_2 i y_2 zostały istotnie obliczone poprawnie.

Jak jednak uzasadnić przyczynę tak dużego błędu x_1 i y_1?
W jaki sposób unikać takich błędów? Odpowiedź w numerze.

Rozwiązanie zadania M 410. Oznaczmy pole tablicy leżące na przecięciu i-tego wiersza i j-tej kolumny przez (i, j). Łatwo zauważyć, że zamiana miejscami dowolnych dwóch wierszy lub dwóch kolumn nie zmienia rozważanych własności tablicy. Weźmy największe k, dla którego można tak poprzestawiać wiersze i kolumny tablicy, aby na polach (i, i) dla $i \leqslant k$ znajdowały się zera. Niech a_{ij} będzie liczbą znajdującą się na polu (i, j) po tym przestawieniu. Zauważmy, że $a_{ij} > 0$ dla $i, j > k$ (w przeciwnym wypadku przestawiając wiersze i-ty i $k+1$-szy kolumny j-tą i $k+1$-szą otrzymalibyśmy zera na miejscach (i, i) dla $i \leqslant k+1$). Jeśli $a_{jm} = 0$ dla pewnych $l \leqslant k$ i $m > k$, to $a_{jl} > 0$ dla wszystkich $j > k$ (inaczej, przestawiając kolumny l-tą i m-tą otrzymalibyśmy zero na miejscu (j, m), $j, m > k$). Stąd

$$\sum_{j=k+1}^{n} (a_{ij} + a_{ji}) \geqslant n - k \text{ dla } i \leqslant k$$

i

$$\sum_{i,j=k+1}^{n} a_{ij} \geqslant (n-k)^2.$$

Ponieważ $a_{ii} = 0$ dla $i \leqslant k$ to

$$\sum_{j=1}^{n} (a_{ij} + + a_{ji}) \geqslant n \text{ dla } i \leqslant k.$$ Mamy więc

$$\sum_{i,j=1}^{n} a_{ij} = \frac{1}{2} \sum_{i=1}^{k} \sum_{j=1}^{n} (a_{ij} + a_{ji}) +$$

$$+ \frac{1}{2} \sum_{i=1}^{k} \sum_{j=k+1}^{n} (a_{ij} + a_{ji}) + \sum_{i,j=k+1}^{n} a_{ij} \geqslant$$

$$\geqslant \frac{1}{2} k \cdot n + \frac{1}{2} k(n-k) + (n-k)^2 =$$

$$= \frac{1}{2} (n^2 + (n-k)^2) \geqslant \frac{1}{2} n^2$$

(w sumie $\sum_{i=1}^{k} \sum_{j=1}^{n} (a_{ij} + a_{ji})$ liczba a_{ij} dla $i, j \leqslant k$ występuje dwa razy i raz dla $i > k$ lub $j > k$).

Rozwiązanie zadania M 411. Weźmy dowolną prostą l na płaszczyźnie. Rzut prostopadły W_i na l jest odcinkiem. Niech A_i będzie jego „lewym" końcem, a B_i „prawym" (ustalamy pewien zwrot na l). Niech A_{i0} będzie najdalej „na prawo" położonym punktem spośród A_i. Punkt A_{i0} należy do każdego odcinka A_iB_i (W_{i0} przecina W_i), a więc prosta k prostopadła do l i przechodząca przez A_{i0} przecina wszystkie wielokąty $W_1, W_2, \ldots, W_n$.

Biorytmy

W ostatnim czasie coraz więcej osób ma dostęp do mikrokomputerów domowych, głównie Meritum oraz Spectrum. Pozwalają one na własnoręczne pisanie prostych programów. Niestety, ze względu na małą pamięć wewnętrzną i brak pamięci zewnętrznej jedynym dostępnym językiem programowania jest Basic. Nie jest to język dobry, ale trudno — jak się nie ma co się lubi, to się lubi co się ma. W *Delcie* będziemy co jakiś czas opisywać konkretne programy dla mikrokomputerów, przedstawiając w skrócie tok rozumowania — od postawienia problemu do gotowego programu w Basicu.

Zaczniemy od rozrywki astrologicznej. Wiadomo od dawna, że każdy człowiek ma w sobie zegar biologiczny mówiący, kiedy należy jeść obiad, a kiedy iść spać. Astrolodzy twierdzą, że do tego dochodzi biologiczny kalendarz: możliwości intelektualne, emocjonalne i fizyczne człowieka rosną i maleją cyklicznie, przy czym każdy cykl zaczyna się w momencie narodzin człowieka. Długość cykli wynosi odpowiednio 32, 27 i 22 dni. Wykres tych cykli, czyli tzw. biorytm, ma być użyteczny przy określaniu, kiedy będzie się najłatwiej uczyć, kiedy podejmować ważne decyzje, a kiedy wybierać się na wielokilometrowy rajd.

Obok znajduje się tekst gotowego programu dla mikrokomputera Meritum. Program ten prosi o podanie daty urodzenia oraz dnia, od którego ma się zaczynać biorytm. Następnie na ekranie zostaje wypisany wykres trzech rytmów. Punkty wykresu intelektualnego są oznaczone literą I, emocjonalnego — literą E, fizycznego — literą F. Pod wykresem program rysuje oś X i opisuje ją, podając numery kolejnych dni.

J. D.

```
10 REM TABLICA DM ZAWIERA LICZBE DNI W KAZDYM MIESIACU
20 REM TABLICA DP ZAWIERA LICZBE DNI OD POCZATKU ROKU
30 REM DO POCZATKU KOLEJNEGO MIESIACA
40 DIM DM(12),DP(12)
50 REM NADAJEMY WARTOSCI POCZATKOWE ELEMENTOM OBU TABLIC
60 FOR I=1 TO 12
70  READ DM(I),DP(I)
80 NEXT I
90 DATA 31,0,29,31,31,59,30,90,31,120,30,151
100 DATA 31,181,31,212,30,243,31,273,30,304,31,334
110 REM NASTEPNA INSTRUKCJA GASI CALY  EKRAN
120 CLS
130 REM PODPROGRAM WOLANY W LINIACH 180 I 210 WCZYTUJE
140 REM DATE Z KLAWIATURY I OBLICZA (JAKO ZAWARTOSC
150 REM ZMIENNEJ LD) LICZBE DNI UPLYWAJACYCH OD
160 REM 1 STYCZNIA 1900 DO PODANEJ DATY
170 PRINT "PODAJ DATE URODZENIA"
180 GOSUB570
190 DU=LD
200 PRINT "PODAJ POCZATKOWA DATE BIORYTMU"
210 GOSUB570
220 LD=LD-DU
230 CLS
240 REM PODPROGRAM WYWOLYWANY W LINIACH 330, 360 I 400
250 REM KRESLI WYKRES  JEDNEGO RYTMU
260 REM PRZED WYWOLANIEM ZMIENNA C$ MUSI ZAWIERAC LITERE
270 REM UZYTA DO KRESLENIA, ZMIENNA DC -  DLUGOSC CYKLU,
280 REM ZMIENNA MD - LICZBE DNI ODDZIELAJACYCH KOLEJNE
290 REM PUNKTY WYKRESU
300 MD=1
310 DC=22
320 C$="F"
330 GOSUB740
340 DC=27
350 C$="E"
360 GOSUB740
370 MD=2
380 DC=32
390 C$="I"
400 GOSUB740
410 REM  INSTRUKCJE  OD  420 DO  450 KRESLA OS X
420 PRINT @896,"-";
430 FOR I=1 TO 21
440  PRINT "+--";
450 NEXT I
460 PRINT @960,"";
470 REM WYWOLYWANY PODPROGRAM OPISUJE OS X
480 GOSUB860
490 REM NASTEPNA INSTRUKCJA WYKONUJE SIE CYKLICZNIE
500 REM DOPOKI UZYTKOWNIK NIE NACISNIE DOWOLNEGO KLAWISZA
510 IF LEN(INKEY$)=0 THEN GOTO510
520 CLS
530 INPUT "CZY CHCESZ KONTYNUOWAC"; C$
540 IF C$="T" THEN GOTO120
550 END
560 REM PODPROGRAM WCZYTYWANIA DATY
570 INPUT "DZIEN   ";DD
580 INPUT "MIESIAC ";MM
590 INPUT "ROK     ";RR
600 REM SPRAWDZ POPRAWNOSC DATY
610 IF MM<1 OR MM>12 THEN GOTO700
620 IF DD<1 OR DD>DM(MM) THEN GOTO700
630 IF RR<1 OR RR>99 THEN GOTO700
640 LD=365*RR+DP(MM)+DD
650 REM TERAZ NALEZY UWZGLEDNIC  LATA PRZESTEPNE
660 RR=RR/4
670 LD=LD+INT(RR)
680 IF RR=INT(RR) AND MM<3 THEN LD=LD-1
690 RETURN
700 PRINT "BLEDNA DATA, WPROWADZ JESZCZE RAZ"
710 GOTO570
720 REM PODPROGRAM  KRESLENIA RYTMU
730 REM NAJPIERW OBLICZMY  LD(MOD DC)
740 Y=LD-DC*INT(LD/DC)
750 REM PETLA OBEJMUJACA INSTRUKCJE 770-830
760 REM WYPISUJE KOLEJNE PUNKTY WYKRESU
770 FOR I=0 TO 63 STEP MD
780  IF Y<=DC/2 THEN Y1=Y ELSE Y1=DC-Y
790  Y1=64*(13-INT(Y1/MD))+I
800  REM PRZECHODZIMY DO NASTEPNEGO DNIA
810  IF Y<DC-MD THEN Y=Y+MD ELSE Y=0
820  PRINT @Y1,C$;
830 NEXT I
840 RETURN
850 REM PODPROGRAM OPISU OSI X
860 DD=DD+1
870 FOR I=1 TO 21
880  REM CZY NALEZY PRZEJSC DO NASTEPNEGO MIESIACA ?
890  IF MM<>2 OR RR=INT(RR) OR DD<=28 THEN GOTO950
900   REM NASTEPNA INSTRUKCJA WYKONA SIE TYLKO DLA
910   REM KONCA LUTEGO W ROKU NIEPRZESTEPNYM
920   DD=DD-28
930   MM=3
940  GOTO1040
950   IF DD<=DM(MM) THEN GOTO1040
960    IF MM<12 THEN GOTO1020
970     REM PRZEJSCIE DO NOWEGO ROKU
980     DD=DD-31
990     MM=1
1000     RR=RR+0.25
1010    GOTO1040
1020     DD=DD-DM(MM)
1030     MM=MM+1
1040  PRINT USING "$$-";DD;
1050  DD=DD+3
1060 NEXT I
1070 RETURN
```

Przy kreśleniu biorytmu trzeba zacząć od wyliczenia liczby dni, które upłynęły od daty narodzin do daty początkowej biorytmu. Najprościej jest wyznaczyć tę liczbę jako różnicę liczby dni dzielących każdą z dat od ustalonej daty odniesienia, np. od początku 1900 roku. W użytym algorytmie posłużono się stablicowaną funkcją liczby dni mijających od początku roku do początku kolejnych miesięcy. W sumie liczbę dni otrzymuje się ze wzoru
(nr roku) · 365 + (liczba dni do początku miesiąca) + (nr dnia w miesiącu) + + (liczba lat przestępnych).
Liczbę lat przestępnych otrzymuje się dzieląc numer roku przez 4. Należy uwzględnić to, że w latach przestępnych dodatkowy dzień jest istotny dopiero dla marca i dalszych miesięcy; w przypadku stycznia i lutego od otrzymanego wyniku należy jeszcze odjąć 1.
Po obliczeniu liczby dni dzielących datę urodzenia od daty rozpoczynającej biorytm przystępujemy do kreślenia. Wykres każdego cyklu rozpada się na część rosnącą i malejącą.
W pierwszej części wartością jest (liczba dni) *mod* (długość cyklu), w drugiej (długość cyklu) — ((liczba dni) *mod* (długość cyklu)).
Maksymalna wartość, równa połowie długości cyklu, przypada w połowie każdego cyklu.
Wyliczenie obu wartości bezpośrednio z podanych wzorów dla każdego kolejnego dnia nie jest potrzebne, można skorzystać z następującej własności operacji *mod n*:

$$(m+1) \bmod n = \begin{cases} (m \bmod n)+1, & \text{jeśli } m \bmod n < n-1 \\ 0, & \text{jeśli } m \bmod n = n-1 \end{cases}$$

Pewien problem powstał przy kreśleniu cyklu intelektualnego. Ma on 32 dni, a zatem maksymalna wartość wynosi 16. Przekracza to rozmiar ekranu Meritum. Trzeba było „przeskalować" wykres tego cyklu w skali 1:2.
Aby uniknąć „schodków" typu

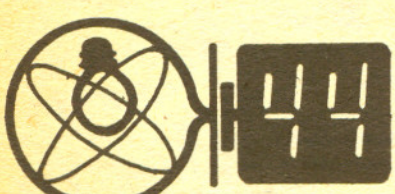
```
                II
              II
            II
```

wprowadzono zasadę, że wykres cyklu jntelektualnego jest drukowany tylko co drugi dzień.
Trochę wysiłku wymaga opisanie osi X. Ze względu na czytelność tylko co trzeci dzień jest opisywany. W sumie — 21 dat w linii. Należy zwrócić uwagę na poprawną obsługę końca miesiąca i końca roku.
Wreszcie kilka uwag dla tych, którzy chcieliby uruchomić program na innym mikrokomputerze. Przede wszystkim potrzebna będzie zmiana instrukcji wypisujących. Instrukcja PRINT @ *n*, ... rozpoczyną wypisywanie od znaku o adresie *n*, przy czym znaki w górnym wierszu mają adresy od 0 do 63, w drugim od góry — 64 do 127, ..., w najniższym — 960 do 1023. PRINT USING „# #"; *n* wypisuje liczbę *n* jako jedną lub dwie cyfry, po których następuje jeden odstęp.
W przypadku mikrokomputerów o lepszej grafice (zwłaszcza kolorowej) wykresy poszczególnych cykli mogłyby być sinusoidami, każda innego koloru lub odcienia.
A może macie jakieś uwagi na temat tego programu?
Jakie inne programy chcielibyście widzieć w *Delcie*? Napiszcie!

Klub 44

Liga zadaniowa Wydziału Matematyki, Informatyki i Mechaniki, Wydziału Fizyki Uniwersytetu Warszawskiego i Redakcji *Delty*

Termin nadsyłania rozwiązań: 30 XI 1985

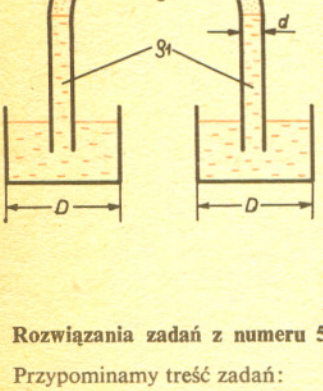


Skrót regulaminu

Każdy może nadsyłać rozwiązania zadań z numeru n w terminie do końca miesiąca $n+2$. Szkice rozwiązań zamieszczamy w numerze $n+4$. Można nadsyłać rozwiązania czterech, trzech, dwóch lub jednego zadania (każde na oddzielnej kartce), można to robić co miesiąc lub z dowolnymi przerwami. Rozwiązania zadań z matematyki i z fizyki należy przesyłać w oddzielnych kopertach, umieszczając na kopercie dopisek: **Klub 44 M** lub **Klub 44 F**. Oceniamy zadania w skali od 0 do 1 z dokładnością do 0,1. Ocenę mnożymy przez współczynnik trudności danego zadania: $WT = 4 - 3S/N$, gdzie S oznacza sumę ocen za rozwiązania tego zadania, a N — liczbę osób, które nadesłały rozwiązanie choćby jednego zadania z danego numeru w danej konkurencji (**M** lub **F**) — i tyle punktów otrzymuje nadsyłający. Po zgromadzeniu **44** punktów, w dowolnym czasie i w którejkolwiek z dwóch konkurencji (**M** lub **F**), zostaje on członkiem **Klubu 44**, a nadwyżka punktów jest zaliczana do ponownego udziału. Trzykrotne członkostwo — to tytuł **Weterana**.
Szczegółowy regulamin został wydrukowany w numerze 1/1985.

Redaguje dr Andrzej NADOLNY

Zadania z fizyki nr 13, 14



13. Czy układ przedstawiony na rysunku, w którym gęstość cieczy ϱ_2 jest większa od gęstości cieczy ϱ_1, może pozostawać w równowadze trwałej? Jeśli tak, to jaki warunek musi być spełniony? Jeżeli zaś nie, to dlaczego? D oraz d oznaczają odpowiednio średnice obu naczyń oraz średnicę (wewnętrzną) rurki.

14. Oszacować, co do rzędu wielkości, stosunek mocy promieniowania widzialnego padającego na dobrze oświetloną powierzchnię, na przykład stołu oświetlonego lampą, do mocy padającego na tę powierzchnię zrównoważonego promieniowania termicznego odpowiadającego temperaturze pokojowej. Ocenić, jakiego rzędu wielkości jest energia promieniowania zawarta w przestrzeni średniej wielkości pokoju. Niezbędne dane należy przyjąć samemu lub wziąć z tablic (żarówka elektryczna emituje w postaci promieniowania widzialnego około 1/30 mocy wydzielanej).

Rozwiązania zadań z numeru 5/1985

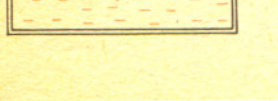
Przypominamy treść zadań:

9. Znaleźć ciśnienie pary nasyconej (tzn. pozostającej w równowadze termodynamicznej z cieczą) nad powierzchnią cieczy, która zawiera rozpuszczoną substancję nielotną w ilości x moli na 1 mol cieczy ($x \ll 1$). Ciśnienie pary nasyconej nad czystą cieczą wynosi w tej samej temperaturze p_0.
10. W książkach fantastyczno-naukowych można niekiedy spotkać opisy bliźniaczej planety Ziemi, która porusza się po tej samej co Ziemia orbicie, lecz z przeciwległej strony Słońca. Jakie byłyby parametry optymalnej orbity pojazdu kosmicznego wysłanego z Ziemi na tę planetę?

9. Rozważmy sytuację przedstawioną na rysunku: środkowy cylinder, zawierający roztwór, zamknięty jest u dołu błoną nie przepuszczającą cząsteczek substancji rozpuszczonej i zanurzony w czystej cieczy; przestrzeń nad cieczą i roztworem wypełniona jest parą nasyconą. W warunkach równowagi ciśnienie p pary nasyconej nad powierzchnią roztworu (A) jest niższe od ciśnienia p_0 pary nasyconej nad powierzchnią czystej cieczy (B) o $\Delta p = \varrho_p g h$ (ϱ_p — gęstość pary nasyconej, g — przyspieszenie ziemskie). Różnica poziomów h jest spowodowana ciśnieniem osmotycznym p_{os}:

$$h = \frac{p_{os}}{\varrho_r g} \approx \frac{p_{os}}{\varrho_c g}$$

(ϱ_r — gęstość roztworu, ϱ_c — gęstość czystej cieczy).



Dla małych stężeń roztworu cząsteczki rozpuszczonej substancji (zakładamy, że nie dysocjuje ona w roztworze) można traktować jako gaz doskonały. Ciśnienie osmotyczne p_{os}, równe ciśnieniu wywieranemu przez ten gaz, obliczamy z równania Clapeyrona $p_{os} = cRT$ (c — stężenie roztworu w molach na jednostkę objętości, R — stała gazowa, T — temperatura bezwzględna). Z powyższych wzorów mamy

$$p = \frac{\varrho_p}{\varrho_c} cRT.$$

Ponownie korzystając z równania Clapeyrona wyznaczamy gęstość pary nasyconej

$$\varrho_p = \frac{\mu p_0}{RT}$$

(μ — masa cząsteczkowa cieczy). Z ostatnich dwóch wzorów wynika

$$\Delta p = \frac{\mu}{\varrho_c} c p_0 = x p_0$$

i w konsekwencji $p = (1-x)p_0$.

10. Poszukiwana orbita będzie wokółsłoneczną orbitą eliptyczną, styczną do orbity ziemskiej w miejscu wystrzelenia. W tym samym miejscu winno nastąpić spotkanie z planetą bliźniaczą. Okres obiegu będzie więc wynosił

$$T = (n+1/2)T_z \qquad (n = 0, 1, 2, \ldots), \quad \text{gdzie } T_z = 1 \text{ rok}.$$

Z trzeciego prawa Keplera wyznaczamy średnią odległość od Słońca dla tej orbity:

$$a = a_z \left(\frac{T}{T_z}\right)^{2/3} = a_z \left(n + \frac{1}{2}\right)^{2/3},$$

gdzie a_z — średnia odległość Ziemi od Słońca. Najkorzystniejsza energetycznie jest orbita dla $n = 0$, dla której $a = 0{,}63\ a_z$ i odległości od Słońca: w peryhelium — $0{,}26\ a_z$, w aphelium — a_z.

Zadania z matematyki nr 115, 116

Redaguje dr Marcin E. KUCZMA

Czołówka ligi zadaniowej "Klub 44M" po uwzględnieniu ocen rozwiązań zadań 107 /WT=2,57/ i 108 /WT=2,42/ z numeru 3/1985

Marcin Mazur	- Białystok	48,05pkt
Anna Gluza	- Toruń	43,93pkt
Jacek Mańdziuk	- Lublin	41,77pkt
Marian Roman	- Ełk	41,28pkt
Tomasz Szymczyk	- Bielsko-B	40,97pkt
Grzegorz Kuś	- Kraków	39,93pkt
Andrzej Sudoł	- Nowy Sącz	39,08pkt

Pan Marcin Mazur jest trzydziestym piątym członkiem Klubu 44.

115. Niech a będzie dodatnim pierwiastkiem równania $x^2-x-1=0$. Dowieść, że dla każdej liczby naturalnej n zachodzi równość $[a^2n] = [a[an]]+1$. ([t] oznacza, jak zwykle, największą liczbę całkowitą $\leqslant t$.)

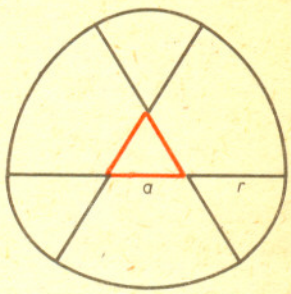
116. Liczby dodatnie a, b, 1 są różne. Czy wykresy funkcji wykładniczych $y = a^x$ i $y = b^x$, rozpatrywane jako podzbiory płaszczyzny, są figurami podobnymi?
Zadanie **116** przysłał pan Werner Mnich z Opola.

Rozwiązanie zadań z numeru 5/1985

Przypominamy treść zadań:

111. Czy istnieje w przestrzeni zbiór wypukły nie będący kulą, ograniczony powierzchnią mającą w każdym punkcie płaszczyznę styczną, przez którego każdy punkt przechodzi odcinek w nim zawarty, o długości równej jego średnicy?
112. Wykazać, że równanie $x^n+y^n = z^{2n+1}$ ma dla każdej liczby naturalnej n nieskończenie wiele rozwiązań w liczbach naturalnych x, y, z.

111. Istnieje; oto przykład. Weźmy dowolne liczby dodatnie a i r, trójkąt równoboczny o boku długości a oraz przedstawioną na rysunku krzywą zamkniętą, leżącą w płaszczyźnie tego trójkąta, posklejaną z łuków okręgów o środkach w wierzchołkach trójkąta i o promieniach r oraz $a+r$. Jest to krzywa gładka (tj. mająca styczną w każdym punkcie); ogranicza ona zbiór płaski o stałej szerokości $a+2r$. Obracając ten zbiór wokół jednej z jego osi symetrii otrzymujemy bryłę obrotową, spełniającą warunki zadania.



112. Równanie spełniają na przykład liczby $x = y = 2^{2kn+k+2}$, $z = 2^{kn+1}$ $(k = 0, 1, 2, \ldots)$.

Uniwersalny czy specjalizowany

Zarówno w przyrodzie, jak i w technice ewolucja przebiega w różnych kierunkach. Z jednej strony powstają układy uniwersalne, dysponujące „nadmiarem" możliwości, mocy lub inteligencji, zdolne do pracy w różnych, być może zmienionych, warunkach. Z drugiej strony istnieją też układy specjalizowane, dokładnie dopasowane do jednego konkretnego zastosowania lub środowiska (lub co najwyżej do niewielkiej ich grupy).

Układy specjalizowane są albo tańsze od układów uniwersalnych o takich samych możliwościach, albo mają większe możliwości od układów uniwersalnych o takiej samej cenie. Na przykład ciężarówka przeznaczona do jazdy po szosie w tropiku jest tańsza od pojazdu kołowo-gąsienicowego przeznaczonego do przewozu towarów i osób w dowolnym terenie i w dowolnych strefach klimatycznych.

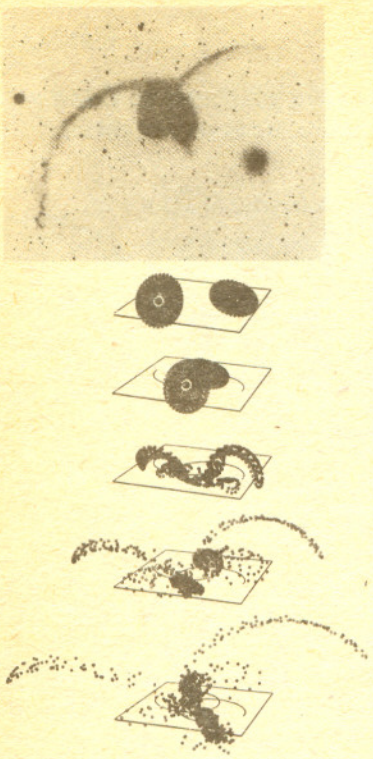
Podobne trendy objawiają się w projektowaniu i produkcji komputerów. Ostatnio niesłychanie modne są uniwersalne mikroprocesory. Taki sam mikroprocesor może znaleźć zastosowanie w komputerze służącym głównie do gier, w komputerze obsługującym radar albo tomograf, a także w komputerze kierującym rakietą samosterującą. Dla usprawnienia pracy dołącza się jednak zazwyczaj dodatkowe, specjalizowane układy.

Na przykład mikrokomputer obsługujący radar musi szybko analizować widmo otrzymanego sygnału, a zatem przeprowadzić transformację Fouriera (algorytmem FFT, patrz *Delta* 12/1984). Kierowanie rakietą wymaga ciągłego porównywania obrazu terenu z zapamiętaną mapą — specjalizowany układ może to przyspieszyć. W mikrokomputerach domowych zazwyczaj wyspecjalizowane układy zajmują się wyświetlaniem informacji na ekranie. Odciąża to bardzo sam mikroprocesor.

Weźmy z kolei pod uwagę liczbę operacji arytmetycznych (na liczbach rzeczywistych) wykonywanych w czasie jednej sekundy. Najszybsze komputery uniwersalne wykonują do 30 milionów takich operacji (30 MFLOPS — ang. Mega Floating Point Operations Per Second). Znaczne zwiększenie szybkości komputerów wymagałoby opracowania zupełnie nowych technologii. Okazało się jednak, że w większości zagadnień wymagających wykonania dużej liczby operacji arytmetycznych (np. w algebrze liniowej) jest możliwe zastosowanie pewnych rozwiązań szczególnych. Można np. równolegle wykonywać jedną operację na kilkudziesięciu zestawach danych (wektorach). Udało się w ten sposób osiągnąć efektywną szybkość nawet rzędu 150 MFLOPS (superkomputer CRAY-1) w „tradycyjnej" technologii układów scalonych.

Innym przykładem jest pojemność pamięci zewnętrznych. Obecnie najczęściej korzysta się z pamięci na dyskach magnetycznych. Jednostka takiej pamięci zapamiętuje do kilkuset milionów bajtów (MB — Mega bajtów; jeden bajt odpowiada jednemu znakowi, a zatem 100 MB odpowiada 50 tys. stron maszynopisu). Jeśli trzeba zapamiętać więcej informacji, to można oczywiście połączyć wiele takich jednostek. Często jednak okazuje się, że duża część informacji nie będzie nigdy modyfikowana. Wówczas można je zapisać na dysku optycznym, działającym podobnie jak coraz popularniejszy „compact-disc". Dysk taki ma pojemność kilku miliardów bajtów (GB — Giga bajtów), a jego cena jest dziesięciokrotnie niższa od ceny typowego dysku magnetycznego o dużej pojemności.

J. D.



Komputerowa symulacja skutków wzajemnego działania na siebie dwóch mijających się galaktyk jako model rzeczywistej sytuacji przedstawionej na górnym zdjęciu.

Komputer zamiast teleskopu

Zjawiska przyrodnicze opisywane są często przez równania różniczkowe. Fakt ten w naturalny sposób narzuca metodę praktycznego śledzenia zjawisk w przypadku, gdy równania te nie dadzą się rozwiązać analitycznie. Mianowicie: każde równanie różniczkowe mniej lub bardziej bezpośrednio mówi, o ile zmieni się jakiś parametr występujący w zjawisku, gdy jego obserwator przemieści się w przestrzeni o małą odległość dx, lub gdy upłynie mały odstęp czasu dt. Na przykład, określiwszy warunki (temperaturę, gęstość, ciśnienie i in.) na powierzchni gwiazdy można — mając komplet równań budowy gwiazdy — obliczyć, jakie warunki panują na małej głębokości dr pod jej powierzchnią. Startując z tak obliczonych parametrów obliczamy na podstawie tych samych równań warunki panujące o dr głębiej w gwieździe. Tak krok za krokiem tworzymy model gwiazdy o zadanych parametrach powierzchniowych. Podobnie mając określone początkowe położenia i prędkości układu punktów materialnych działających na siebie siłami grawitacji możemy — znając równania ruchu — obliczyć prędkości i położenia tych ciał po upływie małego odstępu czasu dt. Wielkości te można znowu uważać za początkowe dla następnego kroku i w ten sposób modelujemy mechaniczną ewolucję układu planetarnego lub gromady gwiazd.

Postępowanie takie, zasadniczo proste, jest z reguły niewykonalne „na papierze" wskutek ogromnej ilości obliczeń niezbędnych do wykonania nawet jednego kroku — tak w każdym razie jest w większości interesujących zagadnień. Wykonanie obliczeń nawet dla uproszczonego modelu gwiazdy czy ruchu komety z uwzględnieniem działania ze strony największych planet byłoby bardzo poważnym przedsięwzięciem. Problemy te znikają przy zastosowaniu komputera. Dzięki wielkiej szybkości obliczeń jest on w stanie w sensownie krótkim czasie nie tylko wymodelować gwiazdę, ale również prześledzić zmiany modelu w czasie, tzn. prześledzić ewolucję gwiazdy. W zastosowaniu do mechaniki komputer umożliwia wręcz oglądanie — np. na ekranie monitora — ewolucji układu wielu tysięcy „gwiazd" (patrz rysunki). Jeszcze 30 lat temu prace takie były nie do wykonania.

T. K.

Zaćmienie Księżyca

Bieżący rok przynosi szczególne warunki obserwacji całkowitych zaćmień Księżyca widocznych z terenu Polski. Dwa zjawiska tego typu, występujące w ciągu jednego roku w ograniczonym obszarze długości geograficznych należą do rzadkości. Pierwsze z nich mogliśmy obejrzeć 4 maja, a drugie — przy sprzyjających warunkach atmosferycznych — mamy szansę zaobserwować 28 października.

Zjawisko rozpocznie się jeszcze przed wschodem Księżyca (Księżyc wschodzi o godzinie 16^h10^m czasu zimowego). Od godziny 15^h38^m Księżyc zacznie wchodzić w półcień naszej planety, czego efektem będzie słabe, praktycznie niezauważalne pociemnienie jego tarczy świecącej jeszcze bardzo nisko nad horyzontem. Znacznie atrakcyjniejsza dla obserwacji faza rozpocznie się o godzinie 16^h55^m, tj. w momencie rozpoczęcia zaćmienia częściowego. Stopniowo w cieniu pogrążać się będzie lewa część tarczy Księżyca i przez blisko półtorej godziny cień będzie „wygryzał" coraz większy jej fragment. Już po zakończeniu zmierzchu astronomicznego — o godzinie 18^h20^m rozpocznie się zaćmienie całkowite, które potrwa do godziny 19^h05^m. W tym czasie pogrążona całkowicie w cieniu Ziemi tarcza naszego satelity powinna być jednak widoczna w całości, choć blask jej pozostanie silnie osłabiony i przyjmie ona barwę miedzianoczerwoną. Jest to wynik „rozjaśniania" cienia Ziemi przez ugięte w otaczającej ją atmosferze promienie słoneczne. Maksymalna faza, tj. najgłębsze zanurzenie Księżyca w cieniu Ziemi wypada w połowie czasu trwania zaćmienia całkowitego — o godzinie 18^h42^m. Następnie w odwrotnej kolejności wystąpią fazy takie, jak na początku trwania zjawiska. Od momentu zakończenia zaćmienia całkowitego do godziny 20^h30^m będzie można obserwować zaćmienie częściowe, podczas którego stopniowo wyłoni się tarcza Księżyca począwszy od lewego jej brzegu. Pozostanie ona pogrążona w półcieniu Ziemi do końca zjawiska, tj. do godziny 21^h46^m.

Obserwatorów posiadających aparat fotograficzny zachęcamy do utrwalenia zjawiska na kliszy. Aby otrzymać obraz tarczy Księżyca z wieloma szczegółami, warto przyłączyć aparat do lunetki astronomicznej lub użyć teleobiektywu. Efektowne zdjęcie całego zjawiska na jednej klatce można uzyskać choćby i „Druhem" unieruchamiając aparat i wykonując ekspozycję co np. 10^m (Księżyc będzie przesuwał się na kliszy na skutek ruchu dziennego). Najciekawsze zdjęcia opublikujemy w naszym czasopiśmie. Życzymy dobrej pogody i udanych obserwacji.

J. U.

Po próbach wyszukiwania kolorowych gwiazd chcemy teraz zaprosić Was do odnajdywania na niebie układów podwójnych złożonych z tak blisko siebie położonych składników, że trudno je gołym okiem rozdzielić.

Wrzesień jest wyjątkowo wdzięcznym miesiącem do obserwacji nieba ze względu na często występującą bardzo dobrą, bezchmurną pogodę.

Wielu Czytelników zapewne wie o starym indiańskim sposobie sprawdzania ostrości wzroku u młodych wojowników. Jeśli poddany testowi potrafi narysować, jak wzajemnie układają się dwie gwiazdy w gwiazdozbiorze Wielkiej Niedźwiedzicy — Alkor i Mizar — to znaczy, że jego wzrok jest dostatecznie „bystry".

Chcemy Wam zaproponować również trudniejszy test. W gwiazdozbiorze Łabędzia, w połowie drogi między α (Denebem) a δ znajduje się słaba gwiazda σ. Otóż jest ona podwójna wizualnie. Odległość obu składników wynosi około 5′. Każda z tych gwiazd jest również podwójna, ale tego już nie da się zauważyć nawet przez największe teleskopy. O ich podwójności wiemy z obserwacji zmian położeń linii widmowych poruszających się składników.

Spójrzcie również na α i β Koziorożca świecące nisko nad horyzontem. Czy widzicie, że są one również układami podwójnymi?

Dokładniejsze obserwacje tych dwóch gwiazd wykazały, że podczas gdy β Koziorożca jest układem dwóch obiegających się wzajemnie gwiazd (tak naprawdę tu znowu składniki są podwójne), to składniki układu α znajdują się blisko siebie jedynie przez przypadek. Słabsza gwiazda znajduje się kilkanaście razy dalej od nas niż gwiazda jaśniejsza.

A więc nie każde dwie gwiazdy znajdujące się blisko siebie na sferze niebieskiej tworzą układ podwójny.

Zachęcamy, abyście pamiętając o tym poszukali innych gwiazd będących sprawdzianem ostrości wzroku.