INFORMATYKA – MÓJ SPOSÓB NA POZNANIE I OPISANIE ŚWIATA

PROGRAM NAUCZANIA INFORMATYKI Z ELEMENTAMI PRZEDMIOTÓW MATEMATYCZNO-PRZYRODNICZYCH

Informatyka – poziom rozszerzony

Granice informatyki

Paweł Perekietka

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

Tytuł: ***Granice informatyki***

Autor: ***Paweł Perekietka***

Redaktor merytoryczny: ***prof. dr hab. Maciej M. Sysło***

Materiał dydaktyczny opracowany w ramach projektu edukacyjnego
Informatyka – mój sposób na poznanie i opisanie świata. Program nauczania informatyki z elementami przedmiotów matematyczno-przyrodniczych

www.info-plus.wwsi.edu.pl

infoplus@wwsi.edu.pl

Wydawca: Warszawska Wyższa Szkoła Informatyki
ul. Lewartowskiego 17, 00-169 Warszawa
www.wwsi.edu.pl
rektorat@wwsi.edu.pl

Projekt graficzny: *Marzena Kamasa*

Warszawa 2013

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

SCENARIUSZ TEMATYCZNY

GRANICE INFORMATYKI

INFORMATYKA – POZIOM ROZSZERZONY

OPRACOWANY W RAMACH PROJEKTU:

INFORMATYKA – MÓJ SPOSÓB NA POZNANIE I OPISANIE ŚWIATA.

PROGRAM NAUCZANIA INFORMATYKI

Z ELEMENTAMI PRZEDMIOTÓW MATEMATYCZNO-PRZYRODNICZYCH

Streszczenie

Komputery to bezsprzecznie bardzo użyteczne narzędzia. Ich moc obliczeniową wykorzystuje się w badaniach naukowych i technicznych. Wspaniały rozwój komputerów, którego jesteśmy świadkami w ostatnich dziesięcioleciach, może spowodować wrażenie wszechmocy techniki komputerowej. Możliwości komputerów są jednak ograniczone. Przykładem jest symboliczna możliwość jednoczesnej reprezentacji bardzo małych i bardzo dużych liczb czy reprezentacji wartości liczbowych z dużą dokładnością. Skutkuje to na przykład kumulowaniem się błędów zaokrągleń. Od ponad 50 lat prowadzi się badania w dziedzinie analizy numerycznej, których celem jest m.in. projektowanie stabilnych algorytmów numerycznych, które umożliwiają kontrolę nawarstwienia się skutków ograniczeń arytmetyki komputerowej w procesie obliczeniowym. Absolwent zajęć z informatyki na poziomie rozszerzonym powinien mieć pewne pojęcie na temat tych ograniczeń i uświadamiać sobie ich przyczyny.

Istnieją też bardziej podstawowe ograniczenia informatyki – natury logicznej czy algorytmicznej. Po pierwsze mamy problemy niealgorytmiczne, czyli problemy decyzyjne, dla których nie istnieje algorytm, który po skończonej liczbie kroków jednoznacznie odpowie Tak lub Nie dla dowolnych danych wejściowych. Naukowcy zajmujący się matematycznymi podstawami informatyki nazywają je nierozstrzygalnymi. Po drugie istnieją całkiem „proste" problemy (proste w sfomułowaniu), które okazują się trudne obliczeniowo – czas obliczeń wymagany dla najlepszych znanych algorytmów rozwiązujących problem okazuje się... większy niż wiek Wszechświata. W czasie trzeciej i czwartej lekcji zagadnienia te zostaną zilustrowane konkretnymi przykładami.

Czas realizacji

4 x 45 minut

Tematy lekcji

1. Arytmetyka komputerowa i jej ograniczenia (2 x 45)
2. Algorytmiczne granice komputyki (2 x 45)

LEKCJA NR 1

TEMAT: Arytmetyka komputerowa i jej ograniczenia

Streszczenie

Jedną z ważniejszych dziedzin informatyki jest analiza numeryczna, która zajmuje się badaniem algorytmów aproksymujących z określoną dokładnością rozwiązania różnych problemów obliczeniowych, w szczególności zaś poszukiwaniem numerycznie stabilnych algorytmów. Chodzi tu o poszukiwanie procedur komputerowych gwarantujących kontrolę nad kumulowaniem (nawarstwianiem) się błędów zaokrągleń, czyli takich, które dla bliskich sobie danych (mały błąd względny) generują zbliżone wyniki (o porównywalnym błędzie względnym).

Nowoczesne komputery mogą wykonywać miliardy działań arytmetycznych na sekundę, co pozwala na rozwiązywanie problemów z różnych dziedzin nie tylko z wykorzystaniem metod klasycznych (używanych wcześniej do ręcznych obliczeń lub do obliczeń przy użyciu kalkulatorów), ale i metod innych niż klasyczne. Maszynowa reprezentacja liczb jest jednak niedokładna i dlatego należy szacować i kontrolować błędy, które mogą pojawić się w czasie obliczeń. Każdy informatyk powinien rozumieć podstawowe zasady analizy numerycznej, by mieć świadomość potrzeby krytycznego podejścia do wyników komputerowych obliczeń (oszacowanie błędu metody i błędów zaokrągleń jest nieodłączną częścią każdego algorytmu numerycznego). Szkolne lekcje informatyki na poziomie rozszerzonym powinny zawierać wprowadzenie do tej tematyki.

W komentarzu opracowanym przez ekspertów Ministerstwa Edukacji Narodowej do podstawy programowej przedmiotu matematyka czytamy: „Dlaczego w podstawie programowej dla gimnazjum i liceum nie wspomniano o niewymierności liczby π i liczby $\sqrt{2}$? (...). Z punktu widzenia matematyki szkolnej (a także inżynierskiej i zastosowań do fizyki) z niewymierności tych liczb nic właściwie nie wnika. Przecież wszystkie wielkości fizyczne są znane tylko w przybliżeniu, bo są efektem jakichś pomiarów. Komputery też posługują się wyłącznie liczbami wymiernymi" (s. 74).

Lekcja ta ma służyć temu, aby absolwent liceum rozumiał dokładnie, co autor powyższego komentarza miał na myśli. Wymagana jest tu znajomość zagadnień z matematyki i informatyki: reprezentacja liczb w systemie dwójkowym, notacja wykładnicza oraz błąd bezwzględny i błąd względny.

Podstawa programowa

Etap edukacyjny: IV, przedmiot: informatyka (poziom rozszerzony)

Cele kształcenia – wymagania ogólne

III. Rozwiązywanie problemów i podejmowanie decyzji z wykorzystaniem komputera, z zastosowaniem podejścia algorytmicznego.

Treści nauczania – wymagania szczegółowe

1.1. Uczeń przedstawia sposoby reprezentowania różnych form informacji w komputerze.

5.11. Uczeń opisuje podstawowe algorytmy i stosuje algorytmy na liczbach całkowitych, np.: reprezentacja liczb w dowolnym systemie pozycyjnym, w tym w dwójkowym i szesnastkowym.

5.27. Uczeń wyjaśnia źródło błędów w obliczeniach komputerowych (błąd względny, błąd bezwzględny).

Cele operacyjne

Uczeń:

- rozumie i potrafi zapisać liczby całkowite, z użyciem różnych kodów (naturalny, uzupełnień, z nadmiarem) i rozumie sens wprowadzania różnych kodów;
- przedstawia ideę zapisu zmiennopozycyjnego liczb ułamkowych;

– ma świadomość ograniczeń arytmetyki komputerowej i wyjaśnia je na przykładach;
– posługuje się narzędziami matematyki w celu wyjaśnienia zagadnień arytmetyki komputerowej;
– posługuje się narzędziami do wizualizacji zagadnień arytmetyki komputerowej;
– dostrzega znaczenie badań w dziedzinie analizy numerycznej.

Słowa kluczowe

kod dwójkowy, kod uzupełnieniowy, kod z nadmiarem, nadmiar (przepełnienie), niedomiar, konwersja, reprezentacja zmiennopozycyjna, cecha, mantysa, błąd zaokrągleń, błąd względny przybliżenia, stabilność numeryczna, analiza numeryczna

Co przygotować



- Prezentacja „Arytmetyka komputerowa”
- Film ukazujący start bezzałogowej rakiety Ariane 501 (http://www.youtube.com/watch?v=jHc2yKZml78)
- Kompilator online C++: http://www.compileonline.com/compile_cpp_online.php oraz kody programów, używane w czasie lekcji

Przebieg zajęć

Wprowadzenie (5-10 minut)

Na początku zajęć nauczyciel wyświetla uczniom krótki (trwający ok. 2 minut) film (http://www.youtube.com/watch?v=jHc2yKZml78) ukazujący start bezzałogowej rakiety Ariane 501, zakończony jej eksplozją po ok. 40 sekundach lotu. W czasie projekcji i po jej zakończeniu objaśnia, że start rakiety miał miejsce 4 czerwca 1996 roku i było owocem ponad 10 lat badań Europejskiej Agencji Kosmicznej (inwestycji wartej kilka miliardów dolarów). W wyniku eksplozji stracone zostały cztery satelity (które rakieta miała wynieść na orbitę okołoziemską), przeznaczone do badań ziemskiej magnetosfery.

Na koniec powinien podkreślić, że z raportu specjalnej komisji wynikało, że źródłem niepomyślnego ciągu zdarzeń, które doprowadziły to wybuchu, była awaria komputera odpowiedzialnego za kontrolę przebiegu startu, spowodowana brakiem zabezpieczeń (procedury obsługi tzw. wyjątku) na okoliczność przepełnienia jednego z 16-bitowych rejestrów (komórek pamięci). Dokładniej: nie powiodła się konwersja (zamiana) wartości liczby (wyrażającej wartość przyspieszenia poziomego rakiety) z reprezentacji zmiennopozycyjnej (64-bitowej) na reprezentację całkowitoliczbową (16-bitową ze znakiem) – konwertowana wartość była większa niż 32 767 i wystąpił problem nadmiaru. Dlaczego nie opracowano procedury obsługi błędu przepełnienia w kodzie oprogramowania (napisanego w języku programowania Ada)? Dlatego, że wyniki analiz pokazywały, że przepełnienie nie ma prawa nastąpić – okazało się, że wyniki analiz były słuszne tylko dla trajektorii lotu starszej rakiety Ariane 4 (nie zaktualizowano tego elementu oprogramowania).

Nauczyciel informuje, że lekcja będzie dotyczyć zagadnień arytmetyki komputerowej, w szczególności zaś konsekwencji ograniczonej możliwości reprezentacji liczb w komputerze. Podany na początku przykład miał pokazać, że praktyka obliczeniowa ma wiele wspólnego z teorią. Inaczej mówiąc: projektant systemu komputerowego nie powinien lekceważyć zagadnień analizy numerycznej...

Zasadnicza część lekcji pierwszej (30-35 minut)

1. Zasadnicza część lekcji powinna rozpocząć się od przypomnienia zagadnienia reprezentacji liczb w systemie dwójkowym (omawianej np. przy okazji tematu „Cyfrowy zapis informacji”). Nauczyciel może poprowadzić ten fragment lekcji w następujący sposób:
   – Pyta uczniów o największą liczbę, którą można zapisać, używając ośmiu bitów.

– Prosi wybranego ucznia o zapisanie na tablicy reprezentacji dwójkowej (binarnej) dwóch liczb: 255 i jednej trochę mniejszej, np. 250.
– Prosi innego ucznia o przedstawienie sposobu zamiany (algorytmu) z reprezentacji dziesiętnej na dwójkową na przykładzie liczby, np. 250.

2. Nauczyciel stwierdza, że wszelkie dane (ale i rozkazy procesora!) reprezentowane są w komputerze za pomocą ciągów cyfr 0 i 1 (bitów, czyli cyfr dwójkowych). Podkreśla, że tę dwuwartościowość realizuje się fizycznie w różny sposób (elektronicznie, optycznie, magnetycznie itd.) – nie to jednak jest celem lekcji.

   Informuje uczniów o tym, że w informatyce (np. praktyce programistycznej) mówi się o 16-towych (2-bajtowych), 32-bitowych (4-bajtowych) czy 64-bitowych (8-bajtowych) słowach (maszynowych) i jest to ściśle związane z architekturą komputera (dlatego mówimy np. o procesorze 64-bitowym).

   W praktyce – np. używając 16-bitowego słowa jesteśmy w stanie zapisać co najwyżej 2^{16} wartości (jeśli przechowujemy informację o liczbach całkowitych bez znaku, to będą to liczby od 0 do $1 + 2 + 2^2 + ... + 2^{15}$, czyli 65 536).

   Warto podkreślić, że informacje na temat wielkości słów maszynowych mają swoje bezpośrednie odzwierciedlenie w typach całkowitych zmiennych, które deklaruje się wewnątrz kodu programu. Na przykład w języku programowania C++ typami podstawowymi dla zmiennych (które 32-bitowy procesor przetwarza najefektywniej) całkowitych są int i unsigned int (odpowiednio dla liczby ze znakiem i bez znaku) – w ich przypadku każda zmienna zajmuje w pamięci komputera cztery bajty (32 bity).

3. Następnie nauczyciel stwierdza, że w czasie pierwszej lekcji zaprezentowany zostanie sposób maszynowej reprezentacji liczb całkowitych (w tym: ujemnych).

   Może rozpocząć od wyświetlenia kodu:

```
#include <iostream>
using namespace std;

int main()
{
    int liczba = 2147483647; // =2^31–1
    cout << liczba << endl;
    liczba = liczba + 1;
    cout << liczba << endl;
    return 0;
}
```

   Do kompilacji i uruchomienia programu wystarczy kompilator online, np. http://www.compileonline.com/compile_cpp_online.php.

   Na ekranie pojawią się dwie liczby (w dwóch wierszach):

```
2147483647
–2147483648
```

   Efektem działania programu jest przekroczenie dozwolonego zakres wartości dla typu całkowitego int. I następuje nazywane czasem przez programistów zjawisko „przewijaniem licznika".

   Nauczyciel podkreśla, że liczby w zmiennej tego typu zapisywane są w kodzie uzupełnieniowym, i w konsekwencji liczby $2^{31} – 1$ nie da się powiększyć o 1, gdyż następuje przepełnienie (nadmiar). W zapisie binarnym liczba 01111111111111111111111111111111, powiększona o 1 daje 10000000000000000000000000000000, co interpretowane jest jako najmniejsza liczba ujemna typu int, czyli $–2^{31}$, gdyż najbardziej znaczący bit jest bitem znaku.

Warto zdublować w kodzie programu wiersze:

```
liczba = liczba + 1;
cout << liczba << endl;
```

aby uczniowie zobaczyli, że program wyświetli liczbę $-2^{31} + 1$, czyli –2147483647.

4. W dalszej kolejności nauczyciel omawia komputerową reprezentację liczb całkowitych (kod uzupełnieniowy) i kod z przesunięciem (kod z nadmiarem), posługując się odpowiednimi slajdami prezentacji „Podstawy arytmetyki komputerowej". Do slajdów dodane są notatki dla nauczyciela, zawierające objaśnienia i dopowiedzenia.
5. Następuje faza ćwiczeń. Uczniowie otrzymują do rozwiązania następujące zadania:
   a) Określ wartość logiczną zdań dotyczących kodu uzupełnieniowego do dwóch:

	P	F
A. Kod 10011 jest reprezentacją liczby –3.		x
B. Reprezentacją liczby –6 jest 11010.	x	
C. Liczbą przeciwną do liczby, której kod to 00001, jest liczba –1.	x	
D. Pewna liczba ma kod 10101. Reprezentacją liczby przeciwnej jest 01011.	x	

   b) Załóżmy, że komputer przechowuje wartości liczbowe w kodzie uzupełnieniowym. Jaką największą i jaką najmniejszą liczbę można zapisać przy użyciu reprezentacji 6-bitowej?
   (Odp. Odpowiednio: $2^5 - 1 = 31$ oraz $-2^5 = -32$)
   c) Zapisz reprezentacje poniższych liczb w kodzie z nadmiarem 7:
   A. 5
   B. –5
   C. 0
   (Odp. Odpowiednio: 1100, 0010, 0111)

Ostatnia faza pierwszej lekcji (5-10 minut)

Ostatnia faza drugiej lekcji może służyć wyjaśnieniu wątpliwości formułowanych przez uczniów oraz prezentacji zadań domowych (zarówno ich treści, jak i zasad pracy nad nimi, możliwości konsultacji, harmonogramu, oceniania).

Zasadnicza część drugiej lekcji (35-40 minut)

1. Nauczyciel omawia reprezentację zmiennopozycyjną, która pozwala zapisywać w komputerze również liczby niecałkowite. Posługuje się odpowiednimi slajdami prezentacji „Podstawy arytmetyki komputerowej".
2. Następuje faza ćwiczeń. Uczniowie otrzymują do rozwiązania następujące zadania:
   a) Określ liczbę, której reprezentacja binarna zmiennopozycyjna to 10101100, wiedząc, że na pole części ułamkowej przeznacza się cztery najmniej znaczące bity. (Odp. –0,875)
   b) Zapisz liczbę 2 5/8 w 8-bitowej reprezentacji zmiennopozycyjnej, wiedząc, że na pole części ułamkowej przeznacza się cztery najmłodsze bity.
   (Odp. 01000101)
   c) Wskaż liczby dodatnie mniejsze od 1, które można zapisać w formacie 8-bitowym.
   Wskazówka: Najmniejsza z nich w naturalnym kodzie dwójkowym ma postać: 0,001 (dlaczego?). Jaki jest kod kolejnej liczby, którą można dokładnie zapisać?
   (Odp. 0,0010001)

d) Wyznacz największą możliwą liczbę dodatnią, jaką można w sposób dokładny zapisać w 8-bitowej reprezentacji zmiennopozycyjnej wiedząc, że na pole części ułamkowej przeznacza się cztery najmniej znaczące bity.

(Odp. $2^4 \cdot (1+1/2 + ¼ + 1/8 + 1/16) = 16 \cdot 31/16 = 31$)

e) Określ wartość logiczną zdań dotyczących 8-bitowej notacji zmiennopozycyjnej:

	P	F
A. Kod 11001000 jest reprezentacją liczby 3.		x
B. Reprezentacją liczby 4 jest 01010000.	x	
C. Wynik działania 7,5 + 0,25 + 0,125 da się zapisać dokładnie.		x
D. Wynik działania 7,5 + 0,125 + 0,125 nie da się zapisać dokładnie.		x

3. Nauczyciel stwierdza, że brak możliwości dokładnej reprezentacji dwójkowej, np. liczby 0,1, może mieć poważne konsekwencje. Poleca uczniom skompilowanie uruchomienie następującego programu:

```
#include <iostream>
using namespace std;

int main()
{
  double x;
  x = 0;

  cout.precision(55);

  while(x!= 1.0)
  {
    x = x + 0.1;
  // cout << x << endl;
  }

return 0;
}
```

Można by oczekiwać, że program zatrzyma się po wykonaniu 10 iteracji. Jest jednak inaczej – program nie zatrzymuje się. Dlaczego?

Usunięcie komentarza przed `cout` i ponowne uruchomienie programu pokaże, że po wykonaniu 10 dodawań nie otrzymamy dokładnie wartości 1. W konsekwencji warunek kontynuacji pętli `x!= 1.0` jest zawsze prawdziwy, bo x nigdy nie będzie równe 1.

To znakomita okazja, aby zwrócić uczniom uwagę na to, aby w kodzie programu komputerowego unikać porównywania dokładnych wartości liczb typu rzeczywistego!

4. Warto, by nauczyciel przywołał autentyczny przykład – zagadkowe zachowanie amerykańskiego systemu antyrakietowego Patriot podczas operacji „Pustynna Burza" podczas wojny z Zatoce Perskiej w 1991 roku. Praprzyczyną braku celności systemu po kilkudziesięciu godzinach jego bezczyn-

ności okazało się fałszywe przekonanie wojskowych inżynierów, że komputer wystarczająco dokładnie wykonuje obliczenia na całkowitych wielokrotnościach liczby 0,1. Skutki były tragiczne (więcej w załączniku).

5. Nauczyciel powinien podkreślić, że w przypadku obliczeń z dziedziny rachunkowości, finansów, ekonomii, a w szczególnie księgowości, gdzie bilanse muszą się zgadzać co do 0,01 podstawowej jednostki monetarnej (albo w przypadku przeliczeń walutowych nawet do 0,001 tej jednostki) tylko dokładny rachunek dziesiętny może być akceptowany.

   W takich przypadkach należy stosować precyzyjną arytmetykę (np. w przypadku złotówek jako jednostkę wybrać gorsze i wykonywać działania na liczbach całkowitych, które mogą być reprezentowane dokładnie lub stosować specjalne typy rzeczywiste stałopozycyjne, dostępne w kompilatorach języka programowania).

Podsumowanie drugiej lekcji (5-10 minut)

Ostatnia faza drugiej lekcji może służyć wyjaśnieniu wątpliwości formułowanych przez uczniów oraz prezentacji zadań domowych (zarówno ich treści, jak i zasad pracy nad nimi, możliwości konsultacji, harmonogramu, oceniania).

Zadanie 1

Szukając rozwiązania złożonego problemu (fizycznego, chemicznego czy interdyscyplinarnego) zwykle nie znamy dokładnych wartości niektórych parametrów i używając wartości przybliżonych, chcemy otrzymać również w przybliżeniu poprawne wyniki. W praktyce może się okazać, że nawet niewielkie zniekształcenie parametrów wejściowych da w rezultacie ogromną zmianę wyniku.

Przypuśćmy, że częścią jakiegoś algorytmu obliczeniowego jest rozwiązanie układu równań liniowych, których rozwiązania są wykorzystywane w dalszych obliczeniach:

$ax + by = c$

$ax + dy = e$

Niech współczynniki a, b i c mają wartość odpowiednio 2, 7 i 5.

Rozwiąż układ równań dla trzech wersji współczynników:

a) d = 7,0000 i e = 5,0000

b) d = 6,9999 i e = 4,9999

c) d = 7,0001 i e = 4,9999

Zwróć uwagę na to, że wykonywałeś obliczenia w sposób dokładny – wyniki obliczeń nie zostały więc zniekształcone przez błędy zaokrągleń, ale jest to uwarunkowane samym zadaniem, które jest – jeśli tak można powiedzieć – „wrażliwe" na dobór parametrów d i e.

Odp. a) układ nieoznaczony, b) (–1, 1) c) (6, –1)

Niestabilne zachowanie (np. układu równań liniowych) nazywa się wrażliwością na warunki początkowe lub efektem motyla. Dla przykładu: Prawidłowe prognozowanie pogody na więcej niż kilka kolejnych dni jest niemożliwe ze względu na nieznajomość chwilowych warunków pogodowych na tyle dokładnie, aby błąd w długookresowych obliczeniach był niezauważalny. W analizie numerycznej mówi się o źle uwarunkowanym zadaniu.

Przykład ten ma uświadomić uczniom wagę dokładności (precyzji) przetwarzanych danych. Gdyby zastosować tu arytmetykę „trzech cyfr dziesiętnych" zamiast „czterech", w każdym z przypadków uzyskalibyśmy ten sam wynik – układ nieoznaczony.

Warto zainteresować uczniów tematem złego uwarunkowania problemu rozwiązywania układów równań liniowych – niedokładność wyników nie jest związana z algorytmem rozwiązywania, ale z samym problemem. Szczegółowe informacje (podane w elementarny sposób) na ten temat można znaleźć np. w książce *Algorytmy* prof. Macieja M. Sysły.

Odsyłam do moich Algorytmów, tam elementarnie wyjaśniono, w czym problem.

Zadanie 2

Dane jest równie kwadratowe: $x^2 + 200x - 0{,}000015 = 0$.

Załóżmy, że posługujemy się arytmetyką dziesiętną 10-cyfrową, czyli wszystkie wyniki pośrednie obliczeń zapisujemy, używając 10 cyfr znaczących.

Możemy dokładnie wyznaczyć $\Delta = b^2 - 4ac = 200^2 - 4 * 1 * 0,000015 = 40\,000{,}00006$, ale pierwiastek z tej liczby musimy zapisać w przybliżeniu: 200,0000001. I dalej obliczamy, korzystając ze znanych wzorów przybliżone wartości rozwiązań: –200,00000005 oraz 0,00000005. Okazuje się, że błąd względny przybliżenia drugiego z rozwiązań jest równy ponad 33%. Natomiast błąd względny przybliżenia pierwszego z pierwiastków jest znikomy (ok. 25×10^{-10}%).

a) Wyznacz przybliżoną wartość drugiego z rozwiązań, używając wzoru:

$$x = \frac{2c}{-b - \sqrt{b^2 - 4ac}}$$

Jaki jest błąd względny tego przybliżenia?

b) Wyznacz przybliżoną wartość drugiego z rozwiązań, używając wzoru Viete'a.

c) Uzasadnij wzór $x = \frac{2c}{-b - \sqrt{b^2 - 4ac}}$.

d) Wykaż, że metody b) i c) są równoważne.

e) Przeanalizuj różne metody rozwiązywania równania $x^2 - 200x - 0{,}000015 = 0$, przy założenie, że posługujesz się arytmetyką dziesiętną 10-cyfrową.

d) Szkolny algorytm Δ okazuje się algorytmem niestabilnym. Co to znaczy? Poszukaj wyjaśnienia przyczyn niedokładności wykonywanych przez algorytm obliczeń.

Zadanie 3

Większość obliczeń naukowych i technicznych to obliczenia przybliżone, realizowane z użyciem techniki komputerowej (wykonywane na liczbach w reprezentacji zmiennoprzecinkowej). Choć to nie człowiek wykonuje obliczenia komputerowe, lecz maszyna, to jednak on wybiera metodę obliczeń i później korzysta z wygenerowanych wyników. Dlatego powinien mieć świadomość ograniczeń arytmetyki komputerowej i potrafić oceniać wybór metody, tzn. jej wpływ na wyniki obliczeń.

Przypuśćmy, że częścią jakiegoś algorytmu jest wykonanie mnożenia przez stałą, której dokładna wartość wynosi $\left(\frac{\sqrt{2}-1}{\sqrt{2}+1}\right)^3$.

a) Uzasadnij, że wartość stałej może być zapisana na kilka równoważnych sposobów:

$$\left(\frac{\sqrt{2}-1}{\sqrt{2}+1}\right)^3 = \left(\sqrt{2}-1\right)^6 = \left(3-2\sqrt{2}\right)^3 = \left(5\sqrt{2}-7\right)^2 = 99 - 70\sqrt{2}.$$

b) Napisz program komputerowy (lub zaprojektuj obliczenia w arkuszu kalkulacyjnym), który będzie służył do oceny przewidywanych konsekwencji wyboru różnych metod obliczeń wartości stałej. W obliczeniach zastąp $\sqrt{2}$ kolejnymi przybliżeniami: 1,4, 1,41, 1,414 itd.

Wyniki obliczeń zapisz w tabeli podobnej do ukazanej na następnej stronie:

$\sqrt{2}$	$\left(\frac{\sqrt{2}-1}{\sqrt{2}+1}\right)^3$	$\left(\sqrt{2}-1\right)^6$	$\left(3-2\sqrt{2}\right)^3$	$\left(5\sqrt{2}-7\right)^2$	$99-70\sqrt{2}$
1,4					
1,41					
1,414					
...					

c) Określ błędy względne wyników otrzymanych w pkt.

d) Porównaj je z błędami stosowanych przybliżeń liczby $\sqrt{2}$.

e) Poszukaj wyjaśnienia przyczyn tak dużych błędów (zwłaszcza dla ostatniego ze wzorów).

Zadanie 4

Projektowanie metody komputerowego rozwiązania problemu naukowego wymaga wykonania dokładnych badań, których celem jest zapewnienie maksymalnej dokładności. Pociąga to za sobą analizę różnego rodzaju błędów, które mogą powstać w wyniku zastosowania danej metody obliczeń.

Celem ilustracji tego zagadnienia posłużmy się przykładem zaokrąglania liczby przez dodanie 1, gdy ostatnią cyfrą rozwinięcia dziesiętnego jest 5. Mamy na przykład: 13,425 ≈ 13,43, czy 1,475 ≈1,48. Wyobraźmy sobie, że dodajemy wiele liczb, zaokrąglanych w ten sposób. Skutkuje to narastaniem niedokładności sumy (efekt kumulowania się błędów zaokrągleń).

Okazuje się, że istnieje lepsza metoda zaokrąglania liczb, których ostatnią cyfrą rozwinięcia dziesiętnego jest 5. Jaka to metoda?

a) Porównaj obie metody, dokonując sumowania losowo wybranych liczb. Stwórz odpowiedni program komputerowy lub zaprojektuj obliczenia w arkuszu kalkulacyjnym. Jako dane wejściowe możesz wykorzystać liczby z tabeli liczb losowych.

b) Przypuśćmy, że wpłacamy 1000 zł na lokatę bankową o oprocentowaniu 5% w skali roku, przy czym odsetki naliczane są codziennie (przyjmijmy w przybliżeniu, że rok ma 360 dni). Załóżmy, że bank przechowuje wartość naszego kapitału w postaci liczby całkowitej wyrażonej w groszach. Każdego dnia stan konta jest mnożony przez (1 + 0,05/360) i sprowadzany do najbliższego grosza.

- Sprawdź, jaki będzie efekt w zależności od stosowanego sposobu zaokrąglania. Stwórz odpowiedni program komputerowy lub zaprojektuj obliczenia w arkuszu kalkulacyjnym.
- Dowiedz się, w jaki sposób banki wykonują obliczenia dotyczące rachunków oszczędnościowych.

  Odp. Lepiej byłoby dodawać 1 tylko w połowie (jeśli to możliwe) przypadków, gdyż wtedy błędy będą się wzajemnie redukować. Można to zrealizować w taki sposób, żeby dodawać 1 tylko do liczb nieparzystych, parzyste pozostawiając bez zmiany.

Zadanie 5

Zamiana liczby mniejszej niż 1 z notacji dziesiętnej na binarną związana jest z mnożeniem przez 2: najpierw samej liczby, a w kolejnych krokach ułamkowej części wcześniej uzyskanego iloczynu. Cyfra znajdująca się na lewo od przecinka w iloczynie jest równa 0 lub 1 i wchodzi w skład reprezentacji binarnej, począwszy od najbardziej znaczącego bitu.

Przykład:

0,3 x 2 = 0,6

0,6 x 2 = 1,2

0,2 x 2 = 0,4

0,4 x 2 = 0,8

0,8 x 2 = 1,6 itd.

Stąd 0,3 ≈ 0,01001010012

a) Wyznacz 16 cyfr rozwinięcia binarnego liczby 0,1.

b) Niech $F < 1$.

Wówczas $F = a \times 2^{-1} + b \times 2^{-2} + c \times 2^{-3} + ...$ gdzie współczynniki a, b, c... są równe 0 lub 1.

Co otrzymamy, jeśli pomnożymy powyższą równość przez 2?

Korzystając z tego spostrzeżenia, uzasadnij poprawność ww. algorytmu zamiany notacji.

c) Wykaż, że ułamek 0,1 jest sumą wyrazów nieskończonego ciągu o wyrazie $a_k = 1/2^{4k} + 1/2^{4k+1}$, tzn. $0,1 = (1/2^4 + 1/2^5) + (1/2^8 + 1/2^9) + (1/2^{12} + 1/2^{13}) + ...$

Zadanie 6

Postawmy problem obliczenia sumy odwrotności wszystkich liczb naturalnych, czyli szeregu 1 + ½ + 1/3 + ¼ +..., zwanego szeregiem harmonicznym.

Intuicja podparta wyliczeniem kilku jego początkowych sum częściowych sugeruje, że wielkość ta jest skończona, a nawet niezbyt duża. Czy zatem rzeczywiście szereg zwany szeregiem harmonicznym jest zbieżny?

a) Napisz program w języku programowania (lub zaprojektuj obliczenia w arkuszu kalkulacyjnym), który pokaże kolejno wartości sum częściowych: 1 + ½, 1 + ½ + 1/3, 1 + ½ + 1/3 + ¼ itd. Od którego momentu te sumy nie zmieniają się? Dlaczego?

b) Można udowodnić, że szereg harmoniczny jest rozbieżny do nieskończoności. Znajdź elementarny dowód i zapisz kolejne kroki dowodu na slajdach prezentacji.

Zadanie 7

Wyobraź sobie, że doszło do poważnego wypadku, spowodowanego awarią systemu komputerowego w szpitalu. Okazało się, że przyczyną była kumulacja błędów zaokrągleń. Sformułuj odpowiedzi na poniższe pytania:

Kto (jeśli ktokolwiek) jest odpowiedzialny za wypadek? Projektant sprzętu? Projektant oprogramowania? Programista, który napisał określony fragment programu? Osoba, która podjęła decyzję o zakupie sprzętu? A jeśli oprogramowanie zostało poprawione przez firmę, która je pierwotnie opracowała, ale uaktualnienie nie zostało nabyte i wprowadzone w tym krytycznym przypadku?

Ocenianie

Nauczyciel może oceniać osiągnięcia uczniów na podstawie obserwacji ich pracy i zaangażowania na lekcji oraz na podstawie prac przygotowanych w ramach zadania domowego.

Informacje o materiałach źródłowych

Nauczycieli zainteresowanych pogłębieniem tematu odsyłam do źródeł wykorzystywanych podczas pisania scenariusza:

1. M.M. Sysło, *Algorytmy,* WSiP 1997 (i wiele późniejszych wydań).
2. J.G. Brookshear, *Informatyka w ogólnym zarysie*, WNT 2003.
3. J. Nievergelt i in., *Informatyczne rozwiązywanie zadań matematycznych*, WNT 1978.
4. W. Gąsowski, M. Kopyt, *Komputer. Jaki jest, każdy zrozumieć może*, WNT 1990.

5. W. Stallings, *Organizacja i architektura systemu komputerowego*, WNT 2000.
6. M. Kubale, *Łagodne wprowadzenie do analizy algorytmów*, Wydawnictwo Politechniki Gdańskiej 2004
7. A. Walat, *Zarys dydaktyki informatyki*, OEKiZK 2006

Załącznik

„Przenieśmy się odrobinę w czasie i przestrzeni, do położonej w Arabii Saudyjskiej miejscowości Dahran. Podczas wojny w Zatoce Perskiej stacjonujące tam odziały Stanów Zjednoczonych wykorzystywały rakiety patriot do namierzania i zestrzeliwania wrogich pocisków, zanim zniszczą one ważne obiekty naziemne.

25 lutego 1991 roku systemy obronne zawiodły – iracka rakieta typu scud trafiła w baraki armii amerykańskiej zabijając 28 żołnierzy i raniąc dalsze 100 osób. Okazało się, że winę za tę katastrofę ponoszą błędy arytmetyki komputerowej. Otóż po uruchomieniu wewnętrzny zegar baterii rakiet patriot zaczynał odliczać czas mierzony w dziesiątych częściach sekundy, ale przechowywany jako liczba całkowita (np. 32, 33 dziesiętne sekundy od uruchomienia). Jednak do obliczenia przyspieszenia oraz położenia śledzonej rakiety potrzebny był czas wyrażony wielkością rzeczywistą. Zatem liczba całkowita podawana przez zegar systemowy była mnożona przez 1/10 i przechowywana w 24-bitowym rejestrze już jako wartość rzeczywista. Pamiętamy jednak, że rozwinięcie dwójkowe 1/10 jest nieskończone. Każdorazowe obcięcie go do 24 bitów (24 miejsc po przecinku w rozwinięciu dwójkowym) powodowało błąd równy ok. 0,000 000 095 w zapisie dziesiętnym. Nie jest to dużo, jednak ten niewielki błąd nawarstwiając się przez np. 100 godzin działania urządzenia robił się już znaczący: $0{,}000\,000\,095 \cdot 100 \cdot 60 \cdot 60 \cdot 10 = 0{,}342$.

Rakieta scud poruszająca się z prędkością 1676 m/s może pokonać w tym czasie ponad pół kilometra, co na ogół wystarczy do tego, by znalazła się poza zasięgiem baterii patriotów".

Łukasz Bodzon, *Błądzić rzeczą nie tylko ludzką... czyli arytmetyka z komputera*, „Matematyka – społeczeństwo – nauczanie", nr 38/2005, s. 39.

LEKCJA NR 2

TEMAT: Algorytmiczne granice informatyki

Streszczenie

Postęp technologiczny zdaje się sugerować, że możliwości techniki komputerowej są nieograniczone. Ogromna liczba dziedzin, w których komputery znalazły swoje zastosowanie, wzmaga jeszcze to wrażenie. Entuzjaści są przekonani, że jeśli nawet dzisiaj coś jest poza zasięgiem maszyn cyfrowych, to na pewno da się te granice przekroczyć w nieodległej przyszłości. Tymczasem okazuje się, że w wielu przypadkach granice możliwości informatyki tkwią nie w ograniczeniach technologicznych, ale w samej naturze rozważanych problemów!

Współczesna matematyka i informatyka zna wiele takich zagadnień. Jednym z nich jest tzw. dziesiąty problem Hilberta. W roku 1900 ten wielki matematyk niemiecki – David Hilbert – postawił następujący problem: czy istnieje algorytm umożliwiający dla każdego równania diofantycznego, czyli wielomianu wielu zmiennych o współczynnikach całkowitych, rozstrzygnięcie, czy ma ono rozwiązanie w liczbach całkowitych? Na przykład wielomian $x^2 + 1 = 0$ takich pierwiastków nie ma, natomiast dla wielomianu $2x^2 + y^2 - 2 = 0$ jedynymi rozwiązaniami są pary liczb (–1,0) i (1,0). Z kolei równanie $x^2 + y^2 - z^2 = 0$ ma nieskończenie wiele rozwiązań. Pytanie to czekało aż 70 lat na odkrycie odpowiedzi – nie da się napisać programu, który by rozwiązał do zagadnienie dla dowolnego równania.

Istnieje wiele praktycznych problemów, dla których znane są metody rozwiązania, ale okazują się, że algorytmy są na tyle czasochłonne, że nie mogą być stosowane w praktyce. Problemy te wymagają bowiem stosowania algorytmów o wykładniczej złożoności obliczeniowej, co w przypadku już średnio dużych rozmiarów danych oznacza czas obliczeń porównywalny z wiekiem Wszechświata...

Klasycznym przykładem takiego trudnego obliczeniowo problemu jest tzw. problem komiwojażera: Należy wyznaczyć trasę, po jakiej powinien poruszać się wędrowny sprzedawca tak, by odwiedził wszystkie zadane miejscowości, a łączna długość drogi, którą pokona była najmniejsza. Znaczenie praktyczne tego zagadnienia jest ogromne. Umiejętność jego rozwiązywania (w praktyce metodami przybliżonymi) jest ważna dla wielu zadań związanych z transportem, a nawet w przypadku planowania ruchu robota przemysłowego, wiercącego otwory w pewnym elemencie...

Z innym tego typu zadaniem, dla którego nie są znane efektywne metody rozwiązania, spotykamy się co roku w każdej szkole, gdzie wkłada się wiele wysiłku w ułożenie planu zajęć. W dużej szkole nierzadko pojawia się pytanie: czy optymalnie wykorzystujemy czas uczniów i nauczycieli?

Podstawa programowa

Etap edukacyjny: IV, przedmiot: informatyka (poziom rozszerzony)

Cele kształcenia – wymagania ogólne

III. Rozwiązywanie problemów i podejmowanie decyzji z wykorzystaniem komputera, z zastosowaniem podejścia algorytmicznego.

Treści nauczania – wymagania szczegółowe

5.1. Uczeń analizuje, modeluje i rozwiązuje sytuacje problemowe z różnych dziedzin;

5.2. Uczeń stosuje podejście algorytmiczne do rozwiązywania problemu;

5. 10. Uczeń stosuje podejście zachłanne w rozwiązywaniu problemów;

5.16. Uczeń opisuje własności algorytmów na podstawie ich analizy;

5.21. Uczeń przeprowadza komputerową realizację algorytmu i rozwiązania problemu.

Cele operacyjne

Uczeń:

- podaje przykłady problemów niealgorytmicznych (nierozstrzygalnych) i problemów trudnych obliczeniowo;
- ma świadomość ograniczeń informatyki, które mają źródło w samej naturze problemów;
- wyjaśnia na przykładach, czym jest wykładnicza złożoność obliczeniowa;
- rozumie konieczność stosowania metod przybliżonych w praktycznych zastosowaniach;
- dostrzega znaczenie badań w dziedzinie analizy złożoności algorytmów.

Słowa kluczowe

problem trudny obliczeniowo, problem nierozstrzygalny, algorytm zachłanny, cykl Hamiltona, problem komiwojażera, minimalne drzewo rozpinające w grafie, algorytm przybliżony (aproksymacyjny)

Przebieg zajęć

Wprowadzenie do pierwszej lekcji (10–15 minut)

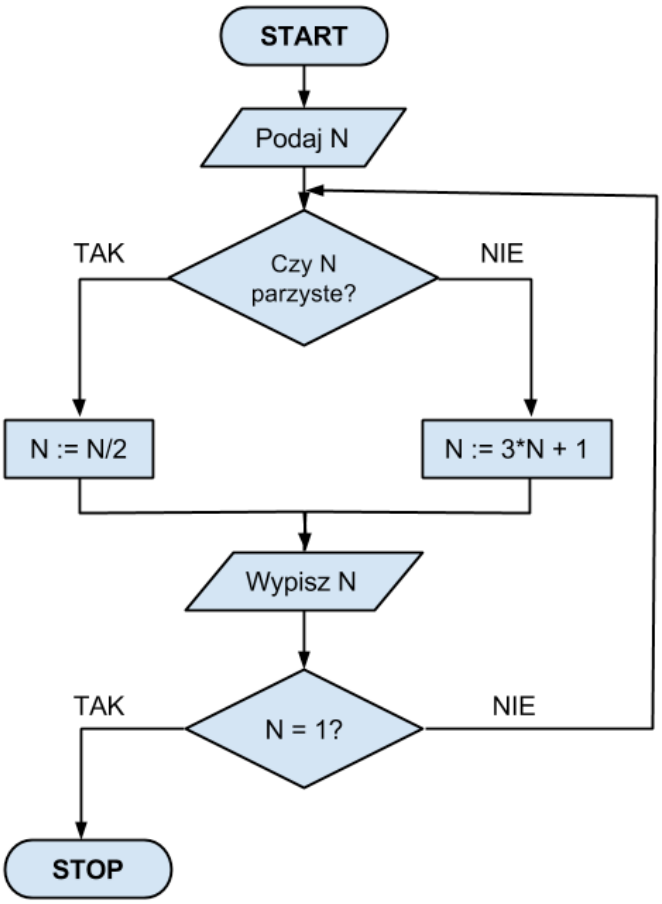
Na początku zajęć nauczyciel formułuje cztery problemy, które będą analizowane przez uczniów podczas pierwszej lekcji. Nie ma konieczności, by wszyscy zajmowali się tymi sami zagadnieniami równolegle – uczniowie powinni pracować w parach (trójkach) i mieć ograniczoną możliwość wyboru problemu, którym będą się zajmować jako pierwszym. Każda para uczniów powinna zajmować się w czasie lekcji dwoma zadaniami: 1 lub 2 oraz 3 lub 4.

W czasie tej lekcji ważne jest, aby uczniowie nie szukali odpowiedzi w Internecie. Zainteresowani będą mogli pogłębić zrozumienie tematu poprzez lekturę internetowych zasobów w domu.

1. Nauczyciel przedstawia drugi z problemów (odmianę **problemu plecakowego**, zwaną problemem sumy podzbioru), formułując go w następujący sposób: „Należy wybrać z listy 191 691 573 337 365 730 651 493 177 354 takie liczby, aby ich suma była równa 2063”.

   Aby upewnić się, że uczniowie właściwie rozumieją treść zadania, nauczyciel może wspólnie z uczniami rozwiązać problem dla zbioru: 1, 4, 5, 2, 3, 6, 7 i sumy 16.

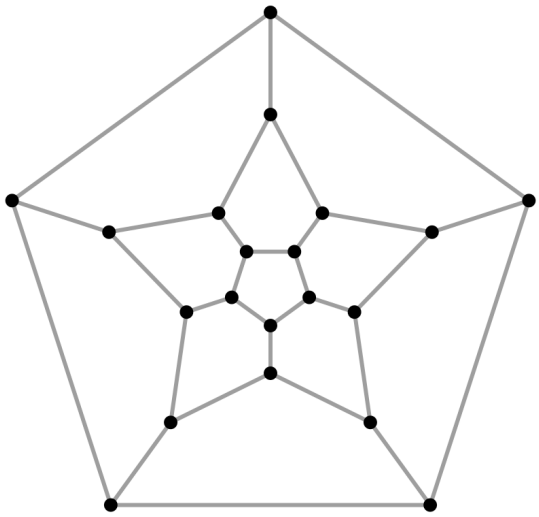
2. Nauczyciel przedstawia uczniom schemat blokowy algorytmu wyznaczania kolejnych wyrazów ciągu Collatza, jeśli mamy dany wyraz początkowy *N*:



(Nie ma potrzeby, aby nauczyciel od razu informował uczniów o nazwie zagadnienia).

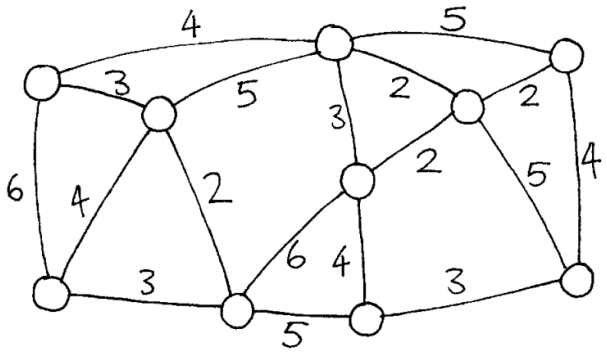
Następnie wspólnie z uczniami sprawdza działanie algorytmu np. dla N = 11 – otrzymany ciąg to: 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1.

3. Trzeci z problemów to tzw. **gra ikosjańska**, czyli łamigłówka spopularyzowana w połowie XIX w. przez W. Hamiltona. Oto słowa, jakich może użyć nauczyciel podczas prezentacji problemu: „Na rysunku przedstawiony jest rzut dwunastościanu foremnego na płaszczyznę. Należy wyjść od pewnego wierzchołka, przejść przez pozostałe, odwiedzając każdy tylko raz, i wrócić do początkowego wierzchołka".



4. Ostatni problem, znany w informatyce pod nazwą **problemu minimalnego (najkrótszego) drzewa rozpinającego**, nauczyciel formułuje w ten sposób: „Rysunek przedstawia sieć dróg powiatowych i miasta. Przy każdej z dróg jest podany szacowany koszt jej odśnieżania (w tysiącach złotych). Starosta, w celu oszczędności, polecił odśnieżać tylko niektóre z nich. Określił przy tym dwa warunki:
   – z każdej miejscowości do każdej innej będzie można dojechać (niekoniecznie najkrótszą drogą),
   koszt odśnieżania będzie jak najmniejszy.

   Które drogi powinny być odśnieżane? Jako odpowiedź podaj łączny koszt odśnieżania".



Zasadnicza część pierwszej lekcji (30-35 minut)

1. Jeśli uczniowie pracujący nad pierwszym zadaniem nie będą mieć wyjątkowego szczęścia, to przekonają się, że zadanie jest czasochłonne.
2. Zadanie uczniów zajmujących się drugim problemem polega na przeanalizowaniu algorytmu. Powinni sprawdzić działanie programu dla możliwie dużej liczby wartości początkowych N, aby zaobserwować zjawisko „wpadania" ciągu w cykl 4, 2, 1 i sformułować hipotezę, nazywaną hipotezą Collatza (Ulama), która można sformułować tak: „Niezależnie od jakiej liczby zaczniemy, w końcu i tak dojdziemy do liczby 1".
3. Zadanie trzecie ma 30 rozwiązań, więc przy odrobinie cierpliwości uczniowie mogą znaleźć przynajmniej jedno z nich. Nauczyciel może nieco utrudnić zadanie, wskazując uczniom początkową sekwencję kilku wierzchołków i polecając dokończenie przejścia.
4. Nauczyciel powinien upewnić się, że uczniowie poprawnie zrozumieli treść czwartego zadania. Odpowiedź do zadania to 25. Uczniowie nie powinni mieć większego problemu ze znalezieniem rozwiązania – powinni odkryć, że skuteczne jest postępowanie zachłanne...

Druga lekcja

Druga lekcja to podsumowanie pracy. Przedstawiciele poszczególnych grup powinni zaprezentować pozostałym uczniom efekty pracy, a nauczyciel je skomentować. Poniżej znajdują się informacje, które nauczyciel może w sposób twórczy wykorzystać w czasie lekcji.

1. Pierwszy z problemów, zwany **problemem sumy podzbioru** (ang. *subset-sum problem*) należy do klasy problemów trudnych obliczeniowo (NP-trudnych). To znaczy, że nie znamy algorytmu o wielomianowej złożoności obliczeniowej. Algorytm naiwny polega na sprawdzaniu wszystkich możliwych kombinacji aż do znalezienia rozwiązania. Jeśli w zestawie liczb jest do wyboru n wartości, to do sprawdzenia mamy 2^n różnych kombinacji. Dwukrotne zwiększenie rozmiaru danych oznacza wykładnicze wydłużenie czasu potrzebnego do wykonania obliczeń. Mamy bowiem $2^{2n} : 2^n = 2^n$: 1, co oznacza, że jeśli zamiast 10 wartości do sprawdzenia mamy ich 20, to czas działania algorytmu jest 2^{10} (≈1000) razy dłuższy.

   Efektywność rozwiązania można nieznacznie poprawić, stosując metodę dziel i zwyciężaj. W tym celu należy w jednym wierszu zapisać wszystkie możliwe sumy dla podzbioru, składającego się z połowy

elementów (jeśli jest ich parzysta liczba), a w drugim wierszu wszystkie sumy dla podzbioru, składającego się z reszty elementów. Następnie trzeba sprawdzić, czy jakakolwiek suma dwóch liczb z różnych wierszy jest równa liczbie, o której jest mowa w zadaniu (najszybciej można to zrobić, jeśli liczby w obu wierszach są uporządkowane).

Są znane inne metody rozwiązania zadania (oparte na technice programowania dynamicznego), o tzw. złożoności pseudowielomianowej, których nie można stosować w praktyce ze względu na ich złożoność pamięciową (wymagania pamięci).

Problem należy do tzw. zadań NP-zupełnych. Mają one zdumiewającą własność, że gdyby do któregokolwiek z nich udało się znaleźć efektywny algorytm, to na jego podstawie można by skonstruować efektywne algorytmy dla wszystkich zagadnień klasy NP. Do grupy problemów NP-zupełnych należą też m.in.: problem komiwojażera, tworzenia harmonogramów (np. układania planu lekcji) oraz następujące: sprawdzić, czy daną mapę można pokolorować trzema kolorami w taki sposób, aby każde dwa obszary mające wspólną granicę były odmiennego koloru. Do tej grupy nie należy prawdopodobnie np. problem faktoryzacji (rozkładu liczby na czynniki pierwsze), który jest podstawą bezpieczeństwa systemu kryptograficznego RSA.

2. Algorytm przedstawiony w postaci schematu blokowego, to algorytm obliczający kolejne liczby ciągu Collatza dla zadanej liczby naturalnej (pierwszego wyrazu ciągu). Problem Collatza jest prawdopodobnie nierozstrzygalny, tzn. przypuszczalnie nie istnieje algorytm pozwalający rozstrzygnąć hipotezę. Nie wiadomo więc, czy istnieje kontrprzykład, czyli taka liczba (dana początkowa), dla której algorytm nigdy się nie zatrzyma.

   W teorii obliczeń, która zajmuje się logicznymi podstawami algorytmiki, o programie realizującym dany algorytm, który zatrzymuje się dla wszystkich możliwych danych mówi się, że ma własność stopu. W przypadku problemu Collatza pytanie o własność stopu tego programu pozostaje otwarte.

   Można udowodnić się, że problem stopu jest nierozstrzygalny, to znaczy, że nie istnieje uniwersalny algorytm rozstrzygający problem stopu dla dowolnego algorytmu (założenie o istnieniu takiego algorytmu prowadzi do sprzeczności). Istnienie problemów, których maszyna cyfrowa nie jest w stanie rozwiązać za pomocą żadnego algorytmu, udowodnił w roku 1937 Alan Turing, który powszechnie uznawany jest za ojca współczesnej informatyki.

   Jak krótko można podsumować ten przykład? Na przykład słowami: Algorytm rozwiązywania danego problemu można uznać za samo rozwiązanie – wydaje się, że jeśli mamy algorytm, to wystarczy uruchomić realizujący go program i poczekać na rozwiązanie... Jak długo należy czekać? Wydaje się, że czasami nieskończenie długo...

3. Trzecia łamigłówka ilustruje szczególny przypadek problemu cyklu Hamiltona, który rozważany jest w teorii grafów: Czy można znaleźć ścieżkę startującą z pewnego wierzchołka, która przechodzi po krawędziach i odwiedza każdy wierzchołek dokładnie raz i wraca do punktu początkowego?

   Niektóre grafy zawierają taki cykl (np. graf dwunastościanu foremnego), a inne nie. Nie jest znany żaden efektywny algorytm, który dla dowolnego grafu pozwoliłby rozstrzygnąć, czy istnieje w nim cykl Hamiltona czy nie. Problem cyklu Hamiltona należy do problemów NP-zupełnych.

   W typowych sytuacjach modelowania (np. organizacji wycieczek, odbioru i dostaw towarów, łańcuchów zaopatrzenia itp.) mamy do czynienia nie tylko z samym grafem (jak dla cyklu Hamiltona), ale także z wartościami przypisanymi jego krawędziom (koszt podróży, jej czas, odległość itp.), a zatem szukamy w grafie nie tylko cyklu, ale chcemy jednocześnie zminimalizować pewne parametry – koszt, czas albo odległość. Tego rodzaju problem określa się zwykle **problemem komiwojażera**, czyli wędrownego sprzedawcy, który chce ukończyć podróż, przebywszy jak najkrótszą trasę. Algorytm polegający na systematycznym sprawdzaniu wszystkich dostępnych rozwiązań jest wykładniczy, ponieważ liczba możliwych dróg rośnie z liczbą miast jak $n!$ W praktyce stosuje się algorytmy przybliżone (aproksymacyjne) – gdy zgodzimy się na to, aby droga była nie więcej niż o 50% większa od optymalnej, to problem da się rozwiązać algorytmem o złożoności wielomianowej.

4. Problem wyznaczania najkrótszego drzewa rozpinającego występuje w wielu sytuacjach praktycznych (np. poszukiwanie połączenia zbioru terminali za pomocą najkrótszej sieci kablowej). Istnieją

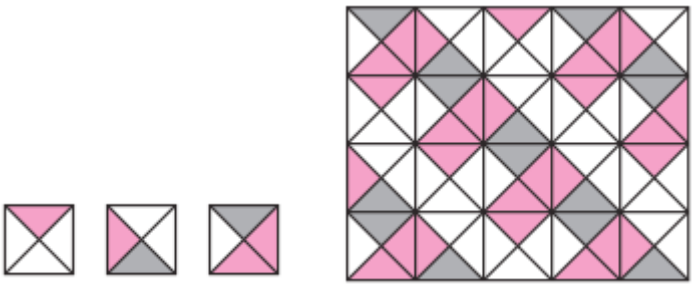
efektywne algorytmy rozwiązujące ten problem, to znaczy pozwalające uzyskać drzewo rozpinające graf (a więc taki podgraf bez cykli wyjściowego grafu, zawierający wszystkie jego wierzchołki) o minimalnej sumie wag krawędzi. Należą one do grupy algorytmów zachłannych. Jednym z nich jest algorytm zaproponowany w roku 1956 przez Josepha Kruskala: na początku porządkujemy krawędzie grafu względem rosnących wag, a w następnych krokach dodajemy kolejne krawędzie z listy, jeśli nie tworzą cyklu z krawędziami już dodanymi.

Korzystając z minimalnych drzew rozpinających można konstruować algorytmy aproksymacyjne dla problemu komiwojażera.

Zadanie 8

W informatyce teoretycznej badanie problemów nierozstrzygalnych ma niebagatelne znaczenie. Są to zwykle zagadnienia mocno teoretyczne, przez co niełatwo dostrzec istotę rzeczy. David Harel w książce *Algorytmika. Rzecz o istocie informatyki* zagadnienie to wprowadza, używając dość prostego przykładu... klocków.

Przypuśćmy, że mamy skończoną liczbę rodzajów kwadratowych klocków (na przykład trzy rodzaje), które mają pokolorowane ćwiartki (jak na rysunku) – klocków każdego rodzaju jest w zestawie nieskończenie wiele. Czy mając tylko takie klocki, można nimi pokryć całą płaszczyznę (w sposób przedstawiony na rysunku)?



Korzystając z edytora grafiki wektorowej, stwórz inny przykład mozajki.

Uwaga: Zakładamy, że klocki mają stałą orientację i nie można ich obracać.

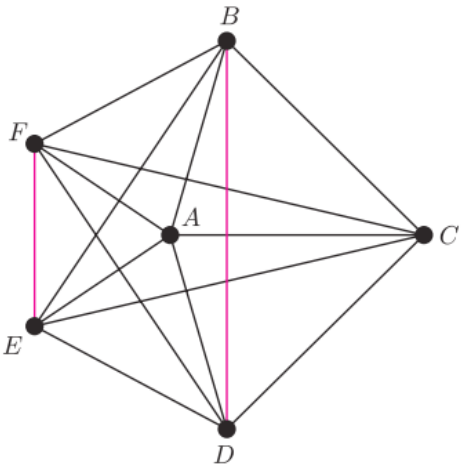
Zadanie 9

Algorytmy aproksymacyjne (przybliżone) są aktywnie badane, ponieważ są jednym ze sposobów efektywnego rozwiazywania trudnych problemów, których nie potrafimy efektywnie dokładnie rozwiązać.

W przypadku problemu komiwojażera algorytm polegający na systematycznym sprawdzaniu wszystkich dostępnych rozwiązań jest wykładniczy, ponieważ liczba możliwych dróg rośnie z liczbą miast jak $n!$ W praktyce stosuje się algorytmy przybliżone – gdy zgodzimy się na to, aby droga była nie więcej niż dwukrotnie większa od optymalnej, to problem da się rozwiązać algorytmem o złożoności wielomianowej.

a) Obejrzyj animację „Komiwojażer", która przedstawia na przykładzie szczegółowo szybki (wielomianowy) algorytm 2-aproksymacyjny dla problemu komiwojażera (w wersji metrycznej, czyli na płaszczyźnie), który posiada następującą własność: koszt znalezionej trasy jest nie większy niż dwukrotny koszt minimalnego drzewa rozpinającego i mniejszy niż dwukrotny koszt najtańszego cyklu Hamiltona.

b) Znajdź inną trasę dla grafu, przedstawionego na animacji, korzystając z innego drzewa rozpinającego.



Ocenianie

Nauczyciel może oceniać osiągnięcia uczniów na podstawie obserwacji ich pracy i zaangażowania na lekcji oraz na podstawie prac przygotowanych w ramach zadania domowego.

Dostepne pliki

1. Prezentacja
2. Animacja
3. Zadania 1-9
4. Test

Informacje o materiałach źródłowych

Nauczycieli zainteresowanych pogłębieniem tematu odsyłam do źródeł wykorzystywanych podczas pisania scenariusza:

1. M.M. Sysło, *Algorytmy*, Warszawa 1997 (i wiele późniejszych wydań).
2. J.G. Brookshear, *Informatyka w ogólnym zarysie*, Warszawa 2003.
3. D. Harel, *Algorytmika. Rzecz o istocie algorytmiki*, Warszawa 1998 (i następne).
4. M. Kubale, *Łagodne wprowadzenie do analizy algorytmów*, Gdańsk 2004.
5. A. Niewiarowska, *Komiwojażer*, „Delta" nr 2/2008.

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska
Wyższa Szkoła
Informatyki

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego