informatyka+

Algorytmika i programowanie

Bazy danych

Multimedia, grafika i technologie internetowe

Sieci komputerowe

Tendencje w rozwoju informatyki i jej zastosowań

Człowiek – najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

informatyka+

Kuźnia Talentów Informatycznych: Algorytmika i programowanie

Przygotowanie do egzaminu maturalnego z informatyki

Maciej M. Sysło

Człowiek – najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

Przygotowanie do egzaminu maturalnego z informatyki

Rodzaj zajęć: Kuźnia Talentów Informatycznych
Tytuł: Przygotowanie do egzaminu maturalnego z informatyki
Autor: prof. dr hab. Maciej M Sysło

Redaktor merytoryczny: prof. dr hab. Maciej M Sysło

Zeszyt dydaktyczny opracowany w ramach projektu edukacyjnego **Informatyka+** – ponadregionalny program rozwijania kompetencji uczniów szkół ponadgimnazjalnych w zakresie technologii informacyjno-komunikacyjnych (ICT).
www.informatykaplus.edu.pl
kontakt@informatykaplus.edu.pl

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

Skład: Recontra Studio Graficzne

Warszawa 2010

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

Przygotowanie do egzaminu maturalnego z informatyki

Maciej M. Sysło
Uniwersytet Wrocławski, UMK w Toruniu
syslo@ii.uni.wroc.pl, syslo@mat.uni.torun.pl

Streszczenie

Ten kurs jest poświęcony przygotowaniu uczniów do zdawania egzaminu maturalnego z informatyki.

Informatyka jako przedmiot szkolny obrosła w wiele nieporozumień. Niektórzy uczniowie, sprawni w posługiwaniu się komputerem uważają, że matura z informatyki jest sprawdzianem głównie tych ich umiejętności. Jest jednak inaczej – ten egzamin wymaga solidnego przygotowania się z uwzględnieniem standardów wymagań egzaminacyjnych z informatyki. Zwracamy również uwagę na nie mniej ważną stronę techniczną matury z Informatyki, gdyż w znaczący sposób powodzenie na tym egzaminie zależy od prawidłowego i bezawaryjnego posługiwania sprzętem komputerowym i jego oprogramowaniem przez zdających.

W Części I materiałów prowadzimy rozważania ogólne na temat matury z informatyki oraz komentujemy najważniejsze dokumenty i wypływające z nich wnioski związane z tym egzaminem.

W Części II zamieszczamy zadania z poprzednich egzaminów maturalnych z informatyki, większość z nich komentując. Zadania w tej części sa podzielone na kilka grup.

Zajęcia kursu mają charakter warsztatowy i polegają głównie na rozwiązywaniu zadań maturalnych z poprzednich egzaminów lub podobnych zadań.

Spis treści

WPROWADZENIE

Zgodnie z obowiązującymi przepisami, informatyka jest jednym z dodatkowych przedmiotów do wyboru na egzaminie maturalnym i może być zdawana na poziomie podstawowym albo rozszerzonym. Ten kurs ma na celu, w formie zajęć warsztatowych, udzielenie pomocy uczniom przygotowującym się do tego egzaminu.

CZĘŚĆ I. DOKUMENTY I ROZWAŻANIA OGÓLNE

Niewątpliwie pewną trudnością w przygotowaniach do tego egzaminu a później w jego zdawaniu jest fakt, iż jest to jedyny przedmiot ogólnokształcący, z którego egzamin maturalny odbywa się z wykorzystaniem, istotnych dla tego egzaminu i zdających, pomocy dydaktycznych, jakimi są komputer i jego oprogramowanie. Trudność ta leży zarówno po stronie organizatorów tego egzaminu, jak i zwłaszcza po stronie zdających. Z tego względu istotne są przepisy i ustalenia związane z przebiegiem tego egzaminu, w tym – dotyczące posługiwania się komputerem i jego oprogramowaniem. Te kwestie formalne omawiamy w Części I, odsyłając uczniów głównie do aktualnych dokumentów, ogłaszanych przez Centralną Komisję Egzaminacyjną (CKE) przynajmniej na dwa lata przez egzaminem[1].

1. EGZAMIN MATURALNY Z INFORMATYKI – PRZEPISY, DOKUMENTY

Podstawowe dokumenty, związane z egzaminem maturalnym, są publikowane przez **Centralną Komisję Egzaminacyjną** (**CKE**), www.cke.edu.pl.

Najważniejsze kwestie formalne, związane z przeprowadzeniem i przebiegiem egzaminu maturalnego z informatyki są poruszone w dokumencie *Informator o egzaminie maturalnym od 2009 roku. Informatyka*, dalej zwanym w tym materiale ***Informatorem***. Zawiera on m.in.:

- opis struktury i formy egzaminu maturalnego z informatyki, w tym:
  - zasady oceniania arkuszy egzaminacyjnych,
  - informacje i zalecenia dla zdających egzamin maturalnych z informatyki,
  - opis przebiegu egzaminu maturalnego z informatyki w części drugiej,
  - opis technicznych warunków przeprowadzenia egzaminu,
  - opis obowiązków i zadań administratora pracowni komputerowej;
- standardy oraz opis wymagań egzaminacyjnych z informatyki dla obu poziomów egzaminu maturalnego;
- przykładowe arkusze z zadaniami egzaminacyjnymi, schematy oceniania tych zadań i przykładowe rozwiązania zadań podane przez uczniów.

Informator jest bardzo ważnym dokumentem, zarówno dla nauczyciela, jak i dla uczniów przygotowujących się do egzaminu maturalnego, gdyż zawiera podstawowe informacje i materiały, które mogą pomóc uczniom bliżej zapoznać się z przebiegiem takiego egzaminu oraz z postacią zadań maturalnych i sposobami ich oceniania. Mamy nadzieję, że coraz więcej uczniów sięga po *Informator*.

Ważnym dokumentem jest również *Egzamin maturalny od 2010 roku. Aneks*, zwany dalej ***Aneksem***. Ten dokument, na str. 215-220, zawiera uaktualnioną, w porównaniu z *Informatorem*, wersję opisu i struktury egzaminu maturalnego z informatyki.

W części praktycznej egzaminu maturalnego z informatyki uczeń pracuje w wybranym przez siebie środowisku programistycznym. Listę środowisk, języków programowania i programów użytkowych, z której mogą wybierać zdający egzamin maturalny z informatyki, ogłasza dyrektor CKE przynajmniej na jeden rok przed egzaminem.

Wymienione powyżej dokumenty znajdują się na stronie Centralnej Komisji Egzaminacyjnej [2]:
Informator: http://www.cke.edu.pl/images/stories/Inf_mat_08/informatyka_1.pdf.
Aneks: http://www.cke.edu.pl/images/stories/Aneks_inf_mat/Aneksy_2010/aneks_2010.pdf.
Opis środowiska: http://www.cke.edu.pl/index.php?option=content&task=view&id=807&Itemid=2.

[1] Kopie wszystkich dokumentów związanych z egzaminem maturalnym z informatyki, wymienionych w tych materiałach, zostały zamieszczone na portalu edukacyjnym tego kursu w folderze z zasobami.

2. PRZEBIEG EGZAMINU MATURALNEGO Z INFORMATYKI

Warto dokładnie zapoznać się z przebiegiem egzaminu maturalnego, a w szczególności ze strukturą i formą egzaminu maturalnego z informatyki.

Egzamin maturalny jest **egzaminem zewnętrznym** pod wieloma względami:

- zadania egzaminacyjne, w formie arkuszy egzaminacyjnych, są przygotowywane poza szkołą, centralnie w CKE, i są identyczne dla wszystkich uczniów zdających egzamin maturalny w danym roku;
- rozwiązania zadań, traktowane anonimowo, są sprawdzane przez egzaminatorów, powoływanych przez CKE;
- egzamin może się odbyć poza szkołą, do której uczęszcza uczeń, np. wtedy, gdy w jego szkole niewielu uczniów wybrało informatykę – w takich przypadkach uczniowie z kilku szkół są gromadzeni w wybranej szkle;
- w komisji egzaminacyjnej w szkole, tworzonej przez odpowiednią Okręgową Komisję Egzaminacyjną (OKE), nie może zasiadać nauczyciel, który uczył zdających przed tą komisją uczniów.

Szczegółowy opis struktury i formy egzaminu maturalnego z informatyki jest zamieszczony w rozdz. IV w *Informatorze*. Komentujemy poniżej najważniejsze z tych ustaleń.

1. Informatykę może być zdawana na maturze na poziomie podstawowym albo na poziomie rozszerzonym. Wyboru poziomu zdający dokonuje w deklaracji składanej do dyrektora szkoły.
   1.1. Egzamin na **poziomie podstawowym** trwa 195 minut i składa się z dwóch części:
   a) część pierwsza trwa 75 minut i polega na rozwiązaniu zestawu zadań bez korzystania z komputera;
   b) część druga trwa 120 minut i polega na rozwiązaniu zadań przy użyciu komputera.
   Zadania egzaminacyjne obejmują zakres wymagań dla poziomu podstawowego.
   W każdej części egzaminu zdający otrzymuje jeden arkusz egzaminacyjny.
   1.2. Egzamin na **poziomie rozszerzonym** trwa 240 minut i składa się z dwóch części:
   a) część pierwsza trwa 90 minut i polega na rozwiązaniu zestawu zadań bez korzystania z komputera;
   b) część druga trwa 150 minut i polega na rozwiązaniu zadań przy użyciu komputera.
   Zadania egzaminacyjne obejmują zakres wymagań dla poziomu rozszerzonego z uwzględnieniem umiejętności wymaganych na poziomie podstawowym.
   W każdej części egzaminu zdający otrzymuje jeden arkusz egzaminacyjny.

2. Uczeń zdaje drugą część egzaminu przy stanowisku komputerowym, którego wyposażenie w oprogramowanie wybrał wcześniej, z listy ogłoszonej przez CKE. Na tej liście na ogół znajduje się oprogramowanie wykorzystywane przez ucznia na zajęciach z informatyki w szkole[2]. Uczeń ma prawo sprawdzić komputer, na którym będzie zdawał egzamin, oraz wybrane przez siebie oprogramowanie, w dniu poprzedzającym egzamin – gorąco zachęcamy do tego, by w czasie egzaminu nie okazało się nagle, że sprzęt jest niesprawny. Nawet jeśli są to komputery szkolne, na których uczniowie pracowali wcześniej przez kilka lat, to warto skorzystać z tej możliwości dokładnego zapoznania się z przygotowanym stanowiskiem do pracy w czasie egzaminu. Może ono bowiem nieco różnić się od tego, przy którym uczeń pracował na lekcjach, np. nie jest możliwy dostęp do Internetu, nie wszystkie systemy oprogramowania będą zainstalowane. Zakłada się, że uczeń pracuje tylko z tym oprogramowaniem, np. językiem programowania, które wybrał na egzamin.

3. W czasie egzaminu, każdy komputer pracuje jako autonomiczna jednostka – nie jest włączony ani do sieci lokalnej, ani do sieci Internet. Można więc przyjąć, że żadne z zadań nie będzie wymagało wykonania *on-line* poszukiwań w sieci. Może jednak pojawić się zadanie wymagające przeszukania pewnych zasobów informacyjnych, np. pochodzących z sieci, ale znajdujących się na płycie.

4. Dane do zadań w drugiej części egzaminu mogą znajdować się na płycie. Podobnie, rozwiązania zadań w tej części należy zapisać na płycie. Należy przestrzegać zaleceń podanych w treści zadań, dotyczących nazw plików, w których mają być umieszczone rozwiązania.

[2] Wyjątkiem jest środowisko komputerów Apple, które nie jest uwzględnione na liście możliwych wyborów przez uczniów.

5. Dla własnego dobra, w drugiej części egzaminu uczeń powinien co jakiś czas zapisywać wyniki swojej pracy, by w razie awarii komputera móc skorzystać z częściowych rozwiązań po przeniesieniu się na inny komputer.

6. Dla zapewnienia, że ewentualne kłopoty techniczne ze sprzętem lub oprogramowaniem nie będą miały wpływu ani na przebieg egzaminu ani na ocenę prac, w sali egzaminacyjnej jest obecny specjalista informatyk, który może udzielać pomocy jedynie w zakresie sprzętu i oprogramowania, jeśli doszło do ich awarii.

Spodziewamy się, że te i inne zasady regulaminu egzaminu maturalnego są przedmiotem dyskusji na zajęciach w szkole. Oficjalnie nie są organizowane próbne egzaminy maturalne z informatyki, ale zapewne poszczególne szkoły organizują je dla swoich uczniów.

3. WYMAGANIA EGZAMINACYJNE Z INFORMATYKI A ZAKRES ZAJĘĆ SZKOLNYCH

Wymagania egzaminacyjne z informatyki są zawarte w rozdz. V *Informatora*. Zamieszczono tam standardy wymagań egzaminacyjnych, jak i szczegółowy ich opis. Opis wymagań ma formę „operacyjną", czyli można się z niego dowiedzieć, co uczeń powinien umieć wykonać.

Należy zwrócić uwagę, że zarówno standardy, jak i szczegółowy ich opis, zawierają zapisy odnoszące się również do materiału nauczania, który wchodzi w zakres **technologii informacyjnej** w szkole ponadgimnazjalnej, będącej rozszerzeniem zakresu zajęć z informatyki w szkole podstawowej i w gimnazjum. Powtórzmy, technologia informacyjna to zastosowania informatyki i ich znajomość wchodzi w zakres wymagań maturalnych z informatyki. Może więc pojawić się zadanie, wymagające użycia edytora tekstu (na ogół do sporządzenia raportu) lub arkusza kalkulacyjnego (do wykonania obliczeń).

Z wykazu standardów egzaminacyjnych i z ich opisów korzystają autorzy zadań maturalnych, należy je więc potraktować jako wykaz wiadomości i umiejętności, które uczeń przystępujący do egzaminu maturalnego powinien opanować.

Należy zwrócić uwagę, że z opisu wymagań egzaminacyjnych wynika, iż zadania maturalne nie mają na celu sprawdzenie umiejętności posługiwania się konkretnym zestawem komputerowym i wybranym oprogramowanie. Na egzaminie maturalnym z informatyki nie są również sprawdzane umiejętności technicznych w posługiwaniu się komputerem, ale wiadomości informatyczne i umiejętności rozwiązywania problemów z pomocą środków i narzędzi informatyki.

4. ALGORYTM, ALGORYTMIKA I ALGORYTMICZNE ROZWIĄZYWANIE PROBLEMÓW

W tym rozdziale skupiamy uwagę na dwóch podstawowych pojęciach, **algorytm** (a ogólniej – **algorytmika**) i **programowanie**, które są przedmiotem większości zadań maturalnych z informatyki.

ALGORYTM

Powszechnie przyjmuje się, że **algorytm** jest opisem krok po kroku rozwiązania postawionego problemu lub sposobu osiągnięcia jakiegoś celu. To pojęcie wywodzi się z matematyki i informatyki – za pierwszy algorytm uznaje się bowiem algorytm Euklidesa, podany ponad 2300 lat temu. W ostatnich latach algorytm stał się bardzo popularnym synonimem przepisu lub instrukcji postępowania.

W szkole, algorytm pojawia się po raz pierwszy na lekcjach matematyki już w szkole podstawowej, na przykład jako algorytm pisemnego dodawania dwóch liczb, wiele klas wcześniej, zanim staje się przedmiotem zajęć informatycznych.

O znaczeniu algorytmów w informatyce może świadczyć następujące określenie, przyjmowane za definicję informatyki:

informatyka jest dziedziną wiedzy i działalności zajmującą się algorytmami

W tej definicji informatyki nie ma dużej przesady, gdyż zawarte są w niej pośrednio inne pojęcia stosowane do definiowania informatyki: **komputery** – jako urządzenia wykonujące odpowiednio dla nich zapisane

algorytmy (czyli niejako wprawiane w ruch algorytmami); **informacja** – jako materiał przetwarzany i produkowany przez komputery; **programowanie** – jako zespół metod i środków (np. języków i systemów użytkowych) do zapisywania algorytmów w postaci programów.

Położenie nacisku w poznawaniu informatyki na algorytmy jest jeszcze uzasadnione tym, że zarówno konstrukcje komputerów, jak i ich oprogramowanie bardzo szybko się starzeją, natomiast podstawy stosowania komputerów, które są przedmiotem zainteresowań algorytmiki, zmieniają się bardzo powoli, a niektóre z nich w ogóle nie ulegają zmianie.

Algorytmy, zwłaszcza w swoim popularnym znaczeniu, występują wszędzie wokół nas – niemal każdy ruch człowieka, zarówno angażujący jego mięśnie, jak i będący jedynie działaniem umysłu, jest wykonywany według jakiegoś przepisu postępowania, którego nie zawsze jesteśmy nawet świadomi. Wiele naszych czynności potrafimy wyabstrahować i podać w postaci precyzyjnego opisu, ale w bardzo wielu przypadkach nie potrafimy nawet powtórzyć, jak to się dzieje lub jak to się stało[3].

Nie wszystkie postępowania z naszego otoczenia, nazywane algorytmami, są ściśle związane z komputerami i nie wszystkie przepisy działań można uznać za algorytmy w znaczeniu informatycznym. Na przykład nie są nimi na ogół przepisy kulinarne, chociaż odwołuje się do nich David Harel w swoim fundamentalnym dziele o algorytmach i algorytmice [7]. Otóż przepis np. na sporządzenie „ciągutki z wiśniami", którą zachwycała się Alicja w Krainie Czarów, nie jest algorytmem, gdyż nie ma dwóch osób, które na jego podstawie, dysponując tymi samymi produktami, zrobiłyby taką samą, czyli jednakowo smakującą ciągutkę. Nie może być bowiem algorytmem przepis, który dla identycznych danych daje różne wyniki w dwóch różnych wykonaniach, jak to najczęściej bywa w przypadku robienia potraw według „algorytmów kulinarnych".

ALGORYTMIKA

Algorytmika to dział informatyki, zajmujący się różnymi aspektami tworzenia i analizowania algorytmów, przede wszystkim w odniesieniu do ich roli jako precyzyjnego opisu postępowania, mającego na celu znalezienie rozwiązania postawionego problemu. Algorytm może być wykonywany przez człowieka, przez komputer lub w inny sposób, np. przez specjalnie dla niego zbudowane urządzenie. W ostatnich latach postęp w rozwoju komputerów i informatyki był nierozerwalnie związany z rozwojem coraz doskonalszych algorytmów.

Informatyka jest dziedziną zajmującą się rozwiązywaniem problemów z wykorzystaniem komputerów. O znaczeniu algorytmu w informatyce może świadczyć fakt, że każdy program komputerowy działa zgodnie z jakimś algorytmem, a więc zanim zadamy komputerowi nowe zadanie do wykonania powinniśmy umieć „wytłumaczyć" mu dokładnie, co ma robić. Bardzo trafnie to sformułował Donald E. Knuth, jeden z najznakomitszych, żyjących informatyków:

Mówi się często, że człowiek dotąd nie zrozumie czegoś,
zanim nie nauczy tego – kogoś innego.
W rzeczywistości,
człowiek nie zrozumie czegoś naprawdę,
zanim nie zdoła nauczyć tego – komputera.

Staramy się, by prezentowane algorytmy były jak najprostsze i by działały jak najszybciej. To ostatnie żądanie może wydawać się dziwne, przecież dysponujemy już teraz bardzo szybkimi komputerami i szybkość działania procesorów stale rośnie (według prawa Moore'a podwaja się co 18 miesięcy). Mimo to istnieją problemy, których obecnie nie jest w stanie rozwiązać żaden komputer i zwiększenie mocy komputerów niewiele pomoże, kluczowe więc staje się opracowywanie coraz szybszych algorytmów. Jak to ujął Ralf Gomory, szef ośrodka badawczego IBM:

Najlepszym sposobem przyspieszania komputerów
jest obarczanie ich mniejszą liczbą działań.

Z powyższych wypowiedzi, których autorami są znakomici i doświadczeni informatycy, wypływają także wnioski dla adeptów informatyki, w tym dla uczniów stawiających pierwsze kroki w rozwiązywaniu

[3] Interesująco ujął to J. Nievergelt – *Jest tak, jakby na przykład stonoga chciała wyjaśnić, w jakiej kolejności wprawia w ruch swoje nogi, ale z przerażeniem stwierdza, że nie może iść dalej.*

problemów z pomocą komputerów, w szczególności dla tych, którzy decydują się na zdawanie egzaminu maturalnego z informatyki:

- rozwiązania problemów informatycznych, zwłaszcza te, które są przeznaczone do wykonania na komputerach, powinny cechować się jasnością i precyzją sformułowań tak, aby „rozumiał" je komputer, który nie może poprosić o wyjaśnienie, tylko bezwzględnie informuje o znalezionych błędach nie podając ich źródeł i sposobów usunięcia;
- nie mniej istotną cechą rozwiązań komputerowych powinna być dbałość o jak najprostsze metody rozwiązywania – stąd w zadaniach maturalnych mogą pojawiać się pytania dotyczące złożoności (efektywności, pracochłonności) przedstawianych lub dyskutowanych rozwiązań.

ALGORYTMICZNE ROZWIĄZYWANIE PROBLEMÓW

Rozwiązywanie problemów z pomocą komputera rządzi się pewnymi zasadami, które dobrze jest znać i stosować. Te zasady mogą być pomocne również w przypadku opracowywania rozwiązań zadań maturalnych. Przedstawiamy te zasady poniżej spodziewając się, że były one również stosowane na lekcjach informatyki.

Komputer jest stosowany do rozwiązywania problemów zarówno przez profesjonalnych informatyków, którzy projektują i tworzą oprogramowanie, jak i przez tych, którzy stosują tylko technologię informacyjno-komunikacyjną, czyli nie wykraczają poza posługiwanie się gotowymi narzędziami informatycznymi. W obu przypadkach ma zastosowanie podejście do **rozwiązywania problemów algorytmicznych**, która polega na systematycznej pracy nad komputerowym rozwiązaniem problemu i obejmuje cały proces projektowania i otrzymania rozwiązania. Celem nadrzędnym tej metodologii jest otrzymanie **dobrego rozwiązania**, czyli takiego, które jest:

- **zrozumiałe dla każdego**, kto zna dziedzinę rozwiązywanego problemu i użyte narzędzia komputerowe,
- **poprawne**, czyli spełnia specyfikację problemu, a więc dokładny opis problemu,
- **efektywne**, czyli niepotrzebnie nie marnuje zasobów komputerowych, czasu i pamięci.

Ta metoda składa się z następujących sześciu etapów:

1. *Opis i analiza sytuacji problemowej.* Na podstawie opisu i analizy sytuacji problemowej należy w pełni zrozumieć, na czym polega problem, jakie są dane dla problemu i jakich oczekujemy wyników, oraz jakie są możliwe ograniczenia.

2. *Sporządzenie specyfikacji problemu*, czyli dokładnego opisu problemu na podstawie rezultatów etapu 1. **Specyfikacja problemu** zawiera:
   - opis danych,
   - opis wyników,
   - opis relacji (powiązań, zależności) między danymi i wynikami.

   Specyfikacja jest wykorzystana w następnym etapie jako specyfikacja tworzonego rozwiązania (np. programu). Często zadanie maturalne opisuje problem językiem potocznym i zdający ma dokładnie opisać problem w postaci jego specyfikacji.

3. *Zaprojektowanie rozwiązania.* Dla sporządzonej na poprzednim etapie specyfikacji problemu, jest projektowane rozwiązanie komputerowe (np. program), czyli wybierany odpowiedni algorytm i dobierane do niego struktury danych. Wybierane jest także środowisko komputerowe (np. język programowania), w którym będzie realizowane rozwiązanie na komputerze. To jest najważniejszy etap rozwiązywania problemów z pomocą komputerów.

4. *Komputerowa realizacja rozwiązania*. Dla projektu rozwiązania, opracowanego na poprzednim etapie, jest budowane kompletne rozwiązanie komputerowe, np. w postaci programu w wybranym języku programowania. Następnie, testowana jest poprawność rozwiązania komputerowego i badana jego efektywność działania na różnych danych.

5. *Testowanie rozwiązania*. Ten etap jest poświęcony na systematyczną weryfikację poprawności rozwiązania i testowanie jego własności, w tym zgodności ze specyfikacją.

6. *Prezentacja rozwiązania.* Dla otrzymanego rozwiązania należy jeszcze opracować dokumentację i pomoc dla (innego) użytkownika. Cały proces rozwiązywania problemu kończy prezentacja innym zainteresowanym osobom (uczniom, nauczycielowi) sposobu otrzymania rozwiązania oraz samego rozwiązania wraz z dokumentacją. Nieodłączną częścią rozwiązań zadań maturalnych, tworzonych przez uczniów, są słowne opisy tych rozwiązań i sposobów ich otrzymywania.

Chociaż powyższa metodologia jest stosowana głównie do otrzymywania komputerowych rozwiązań, które mają postać programów napisanych w wybranym języku programowania, może być zastosowana również do otrzymywania rozwiązań komputerowych większości problemów z obszaru zastosowań informatyki i posługiwania się technologią informacyjno-komunikacyjną, czyli gotowym oprogramowaniem.

Dwie uwagi do powyższych rozważań.

Uwaga 1. Wszyscy, w mniejszym lub większym stopniu, zmagamy się z problemami, pochodzącymi z różnych dziedzin (przedmiotów). W naszych rozważaniach, problem nie jest jednak wyzwaniem nie do pokonania, przyjmujemy bowiem, że **problem** jest sytuacją, w której uczeń ma przedstawić jej rozwiązanie bazując na tym, co wie, ale nie ma powiedziane, jak to ma zrobić. Problem na ogół zawiera pewną trudność, nie jest rutynowym zadaniem. Na takie sytuacje problemowe rozszerzamy pojęcie problemu, wymagającego przedstawienia rozwiązania komputerowego.

Uwaga 2. W tych rozważaniach rozszerzamy także pojęcie **programowania**. Jak powszechnie wiadomo, komputery wykonują tylko programy. Użytkownik komputera może korzystać z istniejących programów (np. za pakietu Office), a może także posługiwać się własnymi programami, napisanymi w języku programowania, który „rozumieją" komputery. W szkole nie ma zbyt wiele czasu, by uczyć programowania, uczniowie też nie są odpowiednio przygotowani do programowania komputerów. Istnieje jednak wiele sposobności, by kształcić zdolność komunikowania się z komputerem za pomocą programów, które powstają w inny sposób niż za pomocą programowania w wybranym języku programowania. Szczególnym przypadkiem takich programów jest oprogramowanie edukacyjne, które służy do wykonywania i śledzenia działania algorytmów. „Programowanie" w przypadku takiego oprogramowania polega na dobieraniu odpowiednich parametrów, które mają wpływ na działanie algorytmów i tym samym umożliwiają lepsze zapoznanie się z nimi.

5. ZAKRES I POSTAĆ ZADAŃ

Na egzaminie maturalnym nie ma zadań bardziej lub mniej ważnych. O znaczeniu zadania dla końcowej punktacji świadczy liczba punktów przydzielonych za jego rozwiązanie lub za poszczególne części rozwiązania (zadania dość często składają się z kilku części, które są osobno oceniane). W treści zadania jest dokładnie opisane, co należy przedłożyć jako rozwiązanie, które podlega ocenie.

Teksty zadań są starannie i precyzyjnie opracowane przez zespół specjalistów i ewentualne ich niezrozumienie na ogół może być spowodowane pewnymi brakami w przygotowaniu się do tego egzaminu. Należy wielokrotnie i z uwagą przeczytać treść zadań, by wyeliminować ewentualne wątpliwości. Niestety, podczas egzaminu nie można nikogo prosić o dodatkowe wyjaśnienie, ani innych zdających uczniów, ani członków komisji. Jedynymi dopuszczalnymi pomocami są dokumentacje oprogramowania, które zostało wybrane do egzaminu.

Jeśli chodzi o zakres zadań egzaminacyjnych, to zakłada się, że sprawdzają one w miarę równomiernie wszystkie standardy wymagań egzaminacyjnych. Aby się o tym przekonać można posłużyć się odpowiednim diagramem w postaci tablicy, w której w wierszach umieszczamy poszczególne zadania (ewentualnie z rozbiciem na części), a w kolumnach – poszczególne elementy standardów wymagań i na przecięciu zadania ze standardem stawiamy znak X, jeśli to zadanie ma na celu sprawdzenie u zdających przygotowania w zakresie tego standardu. Taka tabela jest opracowywana wspólnie dla obu arkuszy egzaminacyjnych. Na jej podstawie można ocenić, że zestaw arkuszy w miarę równomiernie służy do wersyfikacji wiedzy i umiejętności uczniów, jeśli jej wypełnienie znakami X jest w miarę równomierne.

6. OCENIANIE ROZWIĄZAŃ

W każdym arkuszu egzaminacyjnym jest podane, ile punktów można otrzymać w sumie za rozwiązanie wszystkich zadań z tego arkusza. W treści każdego zadania jest określone, ile punktów można otrzymać za to zadanie, a na końcu zadania ta liczba punktów jest rozbita na poszczególne części zadania, jeśli zadanie wyraźnie podzielono na części (patrz rozdz. 13).

W przypadku zadań egzaminacyjnych, pochodzących z odbytych egzaminów, dostępne są również modele odpowiedzi i schematy oceniania poszczególnych zadań (rozdz. 13). Na zajęciach tego kursu, podczas rozwiązywania zadań egzaminacyjnych z poprzednich egzaminów maturalnych, będziemy posługiwali się ich modelami odpowiedzi i schematami oceniania. Pozwoli nam to poznać, co i w jakiej wysokości jest oceniane w rozwiązaniach zadań. Analiza sposobu oceniania rozwiązań poszczególnych zadań może wskazać na te elementy rozwiązań, które są brane pod uwagę przy ocenianiu, a które nie zawsze są dostrzegane przez rozwiązujących.

7. WSKAZÓWKI METODYCZNE

ANALIZA STANDARDÓW WYMAGAŃ EGZAMINACYJNYCH

Standardy wymagań egzaminacyjnych i ich szczegółowy opis, zawarte w *Informatorze* powinny być znane uczniom, przygotowującym się do egzaminu maturalnego. Uczeń zaznajomiony ze standardami nie będzie zaskoczony zadaniami maturalnymi. Standardy wyznaczają także zakres przygotowania do egzaminu maturalnego.

CZĘŚĆ TEORETYCZNA I CZĘŚĆ PRAKTYCZNA

Egzamin maturalny składa się z dwóch części. Można powiedzieć, że pierwsza część jest teoretyczna, bo nie wymaga użycia komputera, a druga – praktyczna, bo polega na posłużeniu się komputerem i jego oprogramowaniem. Do obu tych części należy się odpowiednio przygotować – część zadań należy rozwiązywać bez dostępu do komputera, a część – przy komputerach.

Podczas tego kursu wiele zadań będzie rozwiązywanych bez pomocy komputera, by przygotować uczniów do części teoretycznej matury.

CZYTANIE ZE ZROZUMIENIEM TEKSTÓW ZADAŃ

W przypadku egzaminu zewnętrznego, jakim jest egzamin maturalny, jedną z niezbędnych umiejętności uczniów jest czytanie ze zrozumieniem tekstów zadań. Zgodnie z regulaminem tego egzaminu, zdający sam interpretuje treść zadań i członkowie zespołu nadzorującego egzamin nie mają prawa odpowiadać na żadne pytania związane z interpretacją treści zadań.

Podczas tego kursu, zwykle pod koniec zajęć danego dnia, uczniowie będą otrzymywać zadania do w pełni samodzielnego wykonania, bez pomocy innych uczestników zajęć lub prowadzącego.

ZADANIA „OPISOWE"

Jedną z umiejętności wśród standardów wymagań jest poprawne posługiwanie się terminologią informatyczną, a w konsekwencji także rozumienie tekstów informatycznych i ich tworzenie. Stąd arkusze egzaminacyjne z informatyki zawierają czasem zadania polegające na interpretacji tekstu informatycznego. W odpowiedzi nie wystarczy napisać cokolwiek, jak sądzi wielu uczniów (patrz rozdz. 12 i 13).

Do tej grupy zadań można zaliczyć również zadania polegające na dokładnym określeniu (lub porównaniu) znaczenia odpowiednio wybranych pojęć. Tym zadaniom również poświęcimy odpowiednio dużo czasu podczas tego kursu (w formie testów), gdyż na ogół uczniowie nie przywiązują większej wagi do precyzyjnego określenia znaczenia pojęć związanych z informatyką. Zwłaszcza, ze mając na stałe dostęp do Internetu, mogą zajrzeć do słownika lub encyklopedii on-line. Podczas egzaminu maturalnego nie ma natomiast dostępu do Internetu, ani do żadnej encyklopedii, czy słownika. Kształtowanie znaczenia pojęć może się odbywać podczas rozwiązywania związanych z nimi zadań – tak będziemy postępować podczas tych zajęć.

ROLA PRÓBNEJ MATURY

Zajęcia kursy przejmą rolę próbnej matury z informatyki – wiele zadań uczniowie będą rozwiązywać samodzielnie. Posłużymy się przy tym zadaniami z wcześniejszych egzaminów maturalnych, oraz modelami odpowiedzi i schematami oceniania rozwiązań poszczególnych zadań.

CZĘŚĆ II. PRZYKŁADOWE ZADANIA MATURALNE

Zamieszczamy w tej części wybrane zadania maturalne z informatyki. Zadania te pochodzą z przeprowadzonych egzaminów maturalnych we wcześniejszych latach. Większość zadań jest opatrzonych krótkim komentarzem, który ma ułatwić ich rozwiązanie.

Wiele zadań maturalnych jest zamieszczonych w odpowiednich fragmentach w podręcznikach [5], wskazując jednocześnie ich powiązania z nauczanym materiałem.

W rozwiązywaniu zadań maturalnych związanych z algorytmiką pomocna może być książka [10], w której (od wydania VI, 2008) znajduje się rozdział z wybranymi zadaniami maturalnymi, komentarzami i odsyłaczami do odpowiednich fragmentów tej książki.

Pliki z zadaniami maturalnymi i ich rozwiązaniami znajdują się między innymi na płytach dołączonych do podręczników [5], patrz również: http://www.wsipnet.pl/kluby/informatyka_ekstra.php?k=69.

W kolejnych podrozdziałach zamieszczamy:

- zadania algorytmiczne z arkuszy I, czyli zadania do wykonania bez pomocy komputera;
- zadania algorytmiczne z arkuszy II, czyli zadania, w rozwiązaniu których na ogół trzeba posłużyć się komputerem – wymagane jest załączenie do rozwiązania własnego programu, napisanego podczas egzaminu;
- zadania algorytmiczne z matury na poziomie podstawowym (od 2009 roku);
- zadania algorytmiczne z matury na poziomie rozszerzonym (od 2009 roku);
- zadania niealgorytmiczne, np. związane z bazami danych, arkuszem kalkulacyjnym lub Internetem;
- przykłady zadań z pełną dokumentacją, czyli z modelem odpowiedzi i schematem oceniania.

Na zajęciach będą omawiane również inne typy zadań, w szczególności zadania testowe.

Każdego dnia zajęć, uczniowie będą mieli okazję spróbować swoich sił rozwiązując na zakończenie dnia jedno zadanie w pełni samodzielnie, jak na maturze.

8. ZADANIA ALGORYTMICZNE Z ARKUSZY I

Zadanie: Kraje
(Próbny egzamin maturalny z informatyki, Arkusz I, OKE Wrocław, październik 2001)

Cena (w walucie W) zapinek do skarpetek w Eurolandii, gdzie obowiązuje dziesiętny system liczenia, wynosi 21_{10} W, w Dwójkolandii, gdzie obowiązuje system dwójkowy, tę cenę zapisuje się jako □■□■$_2$ W, zaś w Trójkolandii, gdzie posługują się systemem trójkowym – jako O●○$_3$ W.

W tych trzech krajach wszystkie ceny są liczbami naturalnymi. Nie zawsze jednak ten sam towar ma taką samą cenę w różnych krajach. Na przykład, w Dwójkolandii cena półpancerza wynosi □■□□■□■$_2$ W, a w Trójkolandii – ●O○●$_3$ W.

a) Oblicz ceny półpancerzy praktycznych w Dwójkolandii i Trójkolandii w systemie dziesiętnym. Wyniki wpisz w poniższą ramkę.

Cena półpancerza w Dwójkolandii zapisana w systemie dziesiętnym wynosi:
Cena półpancerza w Trójkolandii zapisana w systemie dziesiętnym wynosi:

b) Oblicz różnicę między cenami wyższą i niższą półpancerzy praktycznych (w Dwójkolandii lub Trójkolandii) i tę różnicę ogłoś w każdym z trzech krajów, czyli zapisz w systemach liczenia tych krajów. Wyniki wpisz w poniższą ramkę.

Różnica w cenie półpancerza praktycznego, zapisana w systemie liczenia danego kraju, wynosi:
w Eurolandii:
w Dwójkolandii:
w Trójkolandii:

c) Podaj algorytm, w postaci listy kroków, schematu blokowego lub w języku programowania, który dokonuje zamiany liczby *k*, zapisanej w systemie pozycyjnym o podstawie *p*, na jej postać w systemie dziesiętnym, gdzie p jest dowolną liczbą naturalną z przedziału [2, 9]. Przyjmij, że:

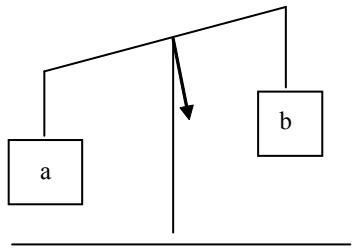
Danymi w algorytmie są:
p, *n*, a_n, a_{n-1}, ..., a_0, gdzie *p* jest podstawą systemu liczenia, $n+1$ jest liczbą cyfr liczby *k*, a a_n, a_{n-1}, ..., a_0 są kolejnymi cyframi liczby *k* (w systemie *p*), począwszy od cyfry najbardziej znaczącej.
Wynikiem jest wartość liczby *k* zapisana w systemie dziesiętnym.

KOMENTARZ
W punkcie a) należy określić, jakie cyfry dwójkowe odpowiadają znakom □ i ■, a jakie cyfry trójkowe odpowiadają znakom ❍, ● i ○. W punkcie b) należy przedstawić liczbę dziesiętną (różnicę między ceną półpancerza w dwóch krainach) w systemie dwójkowym i trójkowym. Zaś w punkcie c), najwyżej punktowany był algorytm korzystający ze schematu Hornera.

[Pełne rozwiązanie tego zadania jest umieszczone w pliku Kraje.pdf.]

Zadanie: Ważenie
(Egzamin maturalny z informatyki w 2002 roku, Arkusza I).

Danych jest *n* przedmiotów o niewielkich gabarytach i różnych wagach. Jest też do dyspozycji waga z dwiema szalkami, ale nie ma odważników. Kładąc na wadze przedmioty a i b, za pomocą jednego ważenia można ustalić, który przedmiot jest lżejszy (zob. rysunek).



Trzeba wybrać najlżejszy i najcięższy przedmiot spośród *n* przedmiotów, posługując się tylko taką wagą.
a) Jaka jest najmniejsza liczba ważeń, którą trzeba wykonać, aby znaleźć najlżejszy przedmiot? Odpowiedź uzasadnij.
b) Podaj specyfikację zadania jednoczesnego znajdowania najlżejszego i najcięższego przedmiotu za pomocą tej wagi. Zapisz algorytm (w postaci listy kroków, schematu blokowego lub wykorzystując język programowania) dla tego zadania, który wykonuje możliwie najmniej ważeń.
c) Podaj, jaka jest liczba ważeń, którą trzeba wykonać w podanym przez Ciebie algorytmie jednoczesnego znajdowania najlżejszego i najcięższego przedmiotu. Odpowiedź uzasadnij.

KOMENTARZ
Przede wszystkim należy zauważyć, że waga w treści zadania to nic innego, jak „urządzenie" do porównywania (ciężaru) przedmiotów – z każdego ważenia otrzymujemy informację, który z dwóch przedmiotów jest lżejszy, a który jest cięższy.

W części a) zadania, chodzi więc o algorytm znajdowania minimum.

Część b) dotyczy problemu jednoczesnego znajdowania minimum i maksimum mamy tutaj do wyboru dwie metody, działające zgodnie z zasadą dziel i zwyciężaj: złożoną z dwóch kroków i metodę rekurencyjną.

W części c) natomiast masz wyznaczyć, ile porównań wykonuje Twój algorytm rozwiązywania problemu z części b).

[Pełne rozwiązanie tego zadania jest umieszczone w pliku Wazenie.pdf.]

Zadanie: Nagroda
(Egzamin maturalny z informatyki w 2002 roku, Arkusz I.)

Pływak Daniel Wodnik jest sponsorowany przez swojego wuja, który na zakończenie kariery pływackiej postanowił ufundować mu specjalną nagrodę pieniężną (w złotówkach).

Daniel miał odnotowane wszystkie czasy uzyskiwane przez siebie w swojej koronnej konkurencji. Były one mierzone z dokładnością do setnych części sekundy.

Wysokość nagrody będzie uzależniona od *najlepszego podciągu. Najlepszym podciągiem* jest najdłuższy malejący podciąg, złożony z kolejnych czasów. Nagrodą będzie tysiąckrotność długości najlepszego podciągu.

Przykład.
Dla następującego ciągu czasów: 23,60; 23,40; 22,61; 24,42; 22,40; 22,22; 21,80; 22,80; 20,80; jego najlepszy podciąg ma długość 4 – jest nim podciąg: 24,42; 22,40; 22,22; 21,80.

a) Uzupełnij specyfikację zadania: Jakiej wysokości nagrodę otrzyma Daniel?

Dane:
Wyniki: Tysiąckrotność długości najlepszego podciągu z ciągu danych.

b) Kolega napisał Danielowi poniższy algorytm znajdowania najlepszego podciągu. Algorytm ten ma błędy. Aby się o tym przekonać, zastosuj go do ciągu z przykładu powyżej. Znajdź te błędy, podkreśl je w wydrukowanym algorytmie i popraw je.

Algorytm	*Poprawne fragmenty wpisz obok błędnych:*
Krok 1. Pobierz pierwszy czas z ciągu danych i zapamiętaj go jako aktualny czas. Ustaw długość aktualnego podciągu równą 0. Ustaw długość najlepszego podciągu równą 0. *Krok 2.* Powtarzaj *Krok 3* dopóki w ciągu danych jest czas, którego jeszcze nie sprawdziłeś; następnie przejdź do *Kroku 4*. *Krok 3.* Aktualny czas zapamiętaj jako poprzedni czas. Pobierz kolejny czas z ciągu danych i zapamiętaj go jako aktualny czas.	....

Jeśli aktualny czas jest mniejszy niż poprzedni czas, to zwiększ długość aktualnego podciągu o 1. W przeciwnym razie, jeśli długość aktualnego podciągu jest mniejsza od długości najlepszego podciągu, to zapamiętaj długość aktualnego podciągu jako długość najlepszego podciągu i ustaw długość aktualnego podciągu na 0. *Krok 4.* Jeśli długość aktualnego podciągu jest większa od długości najlepszego podciągu, to zapamiętaj długość aktualnego podciągu jako długość najlepszego podciągu. *Krok 5.* Ustal nagrodę jako długość najlepszego podciągu pomnożoną przez 1000 i zakończ algorytm.	.....

c) Podkreśl w poniższym ciągu danych elementy najlepszego podciągu:

22,43; 22,42; 23,29; 24,35; 25,37; 24,36; 29,25; 28,30; 26,28; 26,25; 25,21; 25,19; 24,21; 22,20; 22,17; 22,16; 23,15; 23,13; 23,10; 23,09; 23,12; 24,13; 22,12; 21,14.

KOMENTARZ

Zadanie to ma na celu zwrócenie uwagi na znaczenie specyfikacji, zarówno dla określenia problemu, jak i poprawności algorytmu jego rozwiązywania.

W części a) zadania należy podać specyfikację problemu, opisanego na początku zadania słownie.

W części b) jest podany algorytm rozwiązywania tego problemu, zawiera on jednak błędy. Aby je znaleźć, wystarczy zastosować go do przykładu z treści zadania. Znalezione błędy należy poprawić tak, aby algorytm dawał poprawny wynik, czyli zgodny ze specyfikacją problemu, podanej w punkcie a). Poprawiony algorytm należy zastosować w części c) do znalezienia najlepszego podciągu.

[Pełne rozwiązanie tego zadania jest umieszczone w pliku Nagroda.pdf.]

Zadanie: Potęgowanie
(Informator maturalny od 2005 z informatyki, Arkusz I, CKE, Warszawa 2003)

Poniżej podane są dwa sposoby obliczania wartości potęg liczb o wykładnikach naturalnych. Pierwszy sposób opisany jest za pomocą definicji indukcyjnej, a drugi sposób za pomocą algorytmu zapisanego w postaci listy kroków.

Sposób I: $a_0 = 1$ dla $a \in R\backslash\{0\}$,
$a_n = a^{n-1} \cdot a$ dla $n \in N+$, $a \in R\backslash\{0\}$,

Sposób II:
Specyfikacja problemu:

Dane: a – podstawa potęgi, n – wykładnik potęgi dla $n \in N+$, $a \in R\backslash\{0\}$
Wyniki: *wynik* – wartość potęgi o podstawie a i wykładniku n, *wynik* $\in R$
Zmienne pomocnicze: x, k

Krok 1. Nadaj wartości zmiennym: zmiennej *wynik* wartość 1, zmiennej *x* wartość *a*, zmiennej *k* wartość *n*,
Krok 2. Dopóki $k \neq 0$, powtarzaj *Krok 3*,
Krok 3. Jeśli *k* jest liczbą nieparzystą, to *wynik* pomnóż przez *x*, zaś *k* zmniejsz o 1, w przeciwnym przypadku *k* podziel przez 2, zaś *x* pomnóż przez *x*,
Krok 4. Wypisz wartość *wynik*.

Wykonaj polecenia:
a) Zapisz rekurencyjną funkcję obliczania potęgi a^n w wybranym przez siebie języku (pseudojęzyku) programowania.
b) Utwórz schemat blokowy algorytmu opisanego jako Sposób II.
c) Załóżmy, że mamy obliczyć wartość 15^{1000}. Którego sposobu należy użyć? Przed podjęciem decyzji wyznacz złożoność obliczeniową (czasową) i opisz złożoność pamięciową obu wymienionych sposobów. Krótko uzasadnij swój wybór.

KOMENTARZ
Zauważmy, że Sposób I obliczania wartości potęgi jest algorytmem rekurencyjnym, odwołującym się tylko do poprzedniej wartości wykładnika. Sposób II zaś to algorytm rekurencyjny, w którym odwołania są do wykładników prawie o połowę mniejszych. To znacznie przyspiesza obliczenia, o czym można się przekonać wykonując część c) zadania. Warto zauważyć, że kolejność mnożeń przy obliczaniu potęgi, wynikająca ze Sposobu II jest taka sama, jak w algorytmie, który wynika z rozkładu wykładnika na postać binarną i zastosowania schematu Hornera do tej postaci.

Zadanie: Rozmnażanie się pszczół
(Informator maturalny od 2005 z informatyki, Arkusz I, CKE, Warszawa 2003)

Pszczoły rozmnażają się tak, że z zapłodnionych jaj rodzą się samice, a z niezapłodnionych samce (trutnie). Rodzina trutnia jest nietypowa: brak ojca, tylko jeden dziadek i jedna babcia, jeden pradziadek, ale dwie prababcie itd.

Uwaga: Rozwiązując zadania przyjmij, że 0. pokolenie to pokolenie rodziców, 1. to pokolenie dziadków, 2. – pradziadków itd.

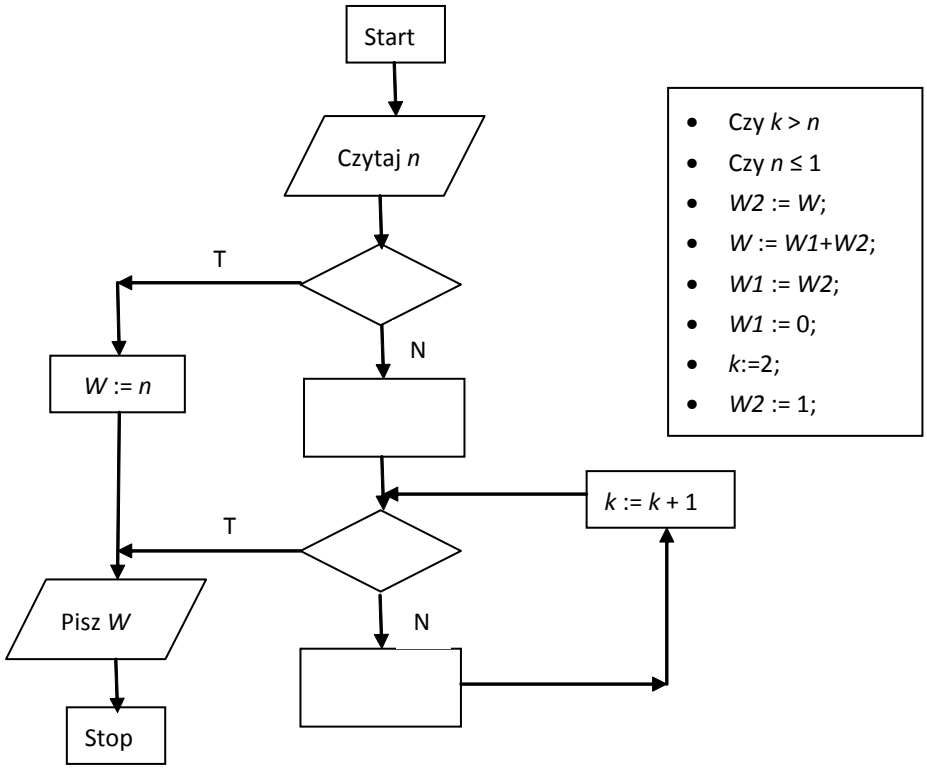
a) Narysuj drzewo genealogiczne trutnia do piątego pokolenia wstecz włącznie.
b) Zapisz rekurencyjny wzór ciągu, który pozwala obliczyć liczbę męskich przodków w *n*-tym pokoleniu.
c) Oblicz, ilu męskich przodków ma truteń w piątym i dziesiątym pokoleniu. Zapisz obliczenia.
d) Poniżej podany jest schemat blokowy algorytmu służącego do obliczania liczby męskich przodków trutnia w *n*-tym pokoleniu wstecz w sposób iteracyjny. Schemat ten zawiera luki. Uzupełnij puste miejsca odpowiednimi instrukcjami i warunkami z listy zamieszczonej obok schematu. Zwróć uwagę na odpowiednią kolejność wpisywanych instrukcji. Uzupełnij również opisy użytych zmiennych.

Specyfikacja problemu

Dane wejściowe	$n \in \mathbb{N}+$
Wynik	$W \in \mathbb{N}+$

Nazwa zmiennej	Opis zmiennej
k	.....
W1, W2	.....

Schemat blokowy z blokami częściowo pustymi, a częściowo wypełnionymi poniższymi napisami:



KOMENTARZ

Po narysowaniu przykładowego drzewa genealogicznego trutnia (punkt a) stanie się jasne, że ich liczby w poszczególnych pokoleniach mają coś wspólnego z… królikami Fibonacciego. Udzielenie odpowiedzi na pozostałe punkty zadania staje się już łatwe. *Uwaga.* W punkcie d), w niektóre bloki schematu należy wpisać więcej niż jedną instrukcję.

Zadanie: Szeregi nieskończone i funkcje elementarne
(Egzamin maturalny z informatyki, Arkusz I, 2005)

Wartości funkcji elementarnych, takich jak sin, cos, log, są obliczane za pomocą komputera w sposób przybliżony. Często stosuje się w tym celu wzory, które mają postać nieskończonych sum. Na przykład prawdziwy jest następujący wzór na wartość logarytmu naturalnego z liczby 2:

$$\ln 2 = \frac{2}{3}(1+\frac{1}{3}\cdot\frac{1}{9}+\frac{1}{5}\cdot\frac{1}{9^2}+\frac{1}{7}\cdot\frac{1}{9^3}+\frac{1}{9}\cdot\frac{1}{9^4}+\frac{1}{11}\cdot\frac{1}{9}+\ldots)$$

W oparciu o powyższy wzór można zaprojektować i napisać program, który dla danej liczby ε (ε > 0) oblicza przybliżoną wartość ln 2, sumując jak najmniej wyrazów, aby różnica między dwoma ostatnimi przybliżeniami była mniejsza niż ε.

Wprowadźmy oznaczenie:

dla n ≥ 1

$$l_n = \frac{2}{3}(1+\frac{1}{3}\cdot\frac{1}{9}+\frac{1}{5}\cdot\frac{1}{9^2}+\frac{1}{7}\cdot\frac{1}{9^3}+\ldots+\frac{1}{2n+1}\cdot\frac{1}{9^n})$$

$$l_0 = 2/3$$

a) Wypełnij tabelę:

N	I_n
0	
1	
2	
3	

Poniżej podaj zależność pomiędzy wartościami I_n i I_{n-1} dla każdego $n = 1, 2, \ldots$
Podaj wzór rekurencyjny na różnicę $r_n = I_n - I_{n-1}$ dla $n>0$:

b) Podaj algorytm ze specyfikacją (w postaci listy kroków, schematu blokowego lub w języku programowania), który dla danej liczby ε (ε > 0) oblicza przybliżoną wartość ln 2, sumując jak najmniej wyrazów we wzorze podanym w treści zadania, aby różnica między dwoma ostatnimi przybliżeniami była mniejsza niż ε.

KOMENTARZ
Algorytm, który należy podać w punkcie c), jest podobny do algorytmu iteracyjnego, służącego do obliczania przybliżonej wartości pierwiastka kwadratowego – we wzorze na ln 2, w nawiasie należy dodać kolejny składnik, jeśli kolejna różnica r_n nie jest mniejsza od ε. *Uwaga*. Do rozwiązania tego zadania nie trzeba wiedzieć ani co to jest logarytm naturalny, ani w jaki sposób otrzymano podany wzór na wartość ln 2.

Zadanie: Ewolucja
(Egzamin maturalny z informatyki, Arkusz I, 2005)

Na planecie MLAP każdy żyjący organizm ma postać napisu złożonego z dużych liter alfabetu łacińskiego. Każdy nowo powstały organizm opisywany jest literą **A**. Po każdym roku życia wielkość organizmu podwaja się w taki sposób, że każda z liter zostaje zastąpiona dwiema literami zgodnie z pewnym ustalonym zbiorem reguł postaci:

$$L \rightarrow F\ S$$

oznaczających, że literę L można zastąpić przez dwie litery: F S. O literze L mówimy wówczas, że występuje po lewej stronie reguły, a F i S występują po prawej stronie reguły.

Przez wielkość organizmu rozumiemy tutaj długość odpowiedniego napisu.

Rozważmy następujący zbiór reguł:

$A \rightarrow B\ C$ $A \rightarrow C\ D$ $B \rightarrow A\ D$
$C \rightarrow B\ A$ $D \rightarrow A\ A$ $D \rightarrow B\ B$

Wówczas organizmy roczne mogą przyjąć jedną z postaci: B C, C D, zaś dwuletnie

A D B A ($A \rightarrow B\ C \rightarrow A\ D\ B\ A$) B A A A ($A \rightarrow C\ D \rightarrow B\ A\ A\ A$)
B A B B ($A \rightarrow C\ D \rightarrow B\ A\ B\ B$)

O dwóch organizmach mówimy, że są w danym momencie odróżnialne, jeśli różne są odpowiadające im napisy (mają różne długości lub różnią się na co najmniej jednej pozycji).

a) Wypisz poniżej wszystkie odróżnialne organizmy trzyletnie, które można uzyskać z organizmu dwuletniego o postaci A D B A.
b) Podaj sposób sprawdzania dla danej liczby naturalnej $n \geq 1$, czy mogą istnieć organizmy o długości n. W przypadku odpowiedzi pozytywnej należy również ustalić wiek organizmu o wielkości n. Podaj, ile poprawnych wielkości organizmów występuje w przedziale $(n, m]$ dla liczb naturalnych n i m, gdzie $n < m$. Odpowiedź uzasadnij.

c) Przyjmijmy, że każda litera pojawiająca się w regułach występuje dokładnie raz po lewej stronie reguły, przed „strzałką" (zauważmy, że powyższy przykład nie spełnia tego warunku, ponieważ litery A i D występują każda z lewej strony w dwóch regułach). Ile odróżnialnych organizmów w wieku 1, 2, 3 itd. może wówczas występować? Odpowiedź uzasadnij.

d) Poniżej przedstawiona jest funkcja wspomagająca realizację następującego zadania: dla zadanego zbioru reguł, nowo powstałego organizmu *start* i danego napisu należy ustalić, czy napis ten przedstawia organizm, który można uzyskać przy pomocy reguł zadanych w treści zadania.

Niech: $L_1 \rightarrow F_1\ S_1$, $L_2 \rightarrow F_2\ S_2$, ..., $L_p \rightarrow F_p\ S_p$ – dany zbiór reguł

Specyfikacja funkcji *sprawdź*:

Dane: *napis* – ..... ,
start – ..... ,

Wynik: odpowiedź, czy *napis* przedstawia organizm, który można uzyskać przy pomocy podanych reguł, gdy nowo powstały organizm jest opisywany przez *start*.

Treść funkcji *sprawdź:*

jeśli długość *napisu* nie jest potęgą liczby 2, to zakończ wykonywanie funkcji z odpowiedzią NIE0
w przeciwnym razie wykonuj:

- jeśli *napis* = *start*, to zakończ wykonywanie funkcji z odpowiedzią TAK;
- jeśli długość napisu jest równa 1, to zakończ wykonywanie funkcji z odpowiedzią NIE;
- podziel *napis* na dwie równe części: *napis1* i *napis2*;
- dla $i = 1, 2, ..., p$ wykonuj:
  jeśli L_i = *start*, to
  - wykonaj funkcję *sprawdź* rekurencyjnie dla *napis* = *napis1*, *start* = F_i oraz dla *napis* = *napis2* i *start* = S_i;
  - jeśli oba rekurencyjne wywołania funkcji *sprawdź* zakończyły się odpowiedzią TAK, to zakończ wykonywanie funkcji z odpowiedzią TAK;
- jeśli w powyższej pętli nie zakończyliśmy działania funkcji, to zakończ jej wykonywanie z odpowiedzią NIE.

Dla podanej powyżej funkcji uzupełnij jej specyfikację.

Podaj parametry wszystkich rekurencyjnych wywołań funkcji *sprawdź* przy uruchomieniu jej dla następującego zbioru reguł:

A → B C A → C D B → A D C → B A
D → A A D → B B

oraz *napis* = B C A A A D C D i *start* = A.

Jaką odpowiedź da funkcja w tym przypadku?

KOMENTARZ

Odpowiedź na pierwsze pytanie w punkcie b) jest zawarta… w treści funkcji *sprawdź*. Dalsza część odpowiedzi w punkcie b) wymaga posłużenia się funkcją logarytm lub dzieleniem przez 2 (funkcja logarytm i dzielenie są działaniami odwrotnymi do potęgowania). Funkcja *sprawdź* jest rekurencyjną realizacją działania odwrotnego do tworzenia organizmów.

Zadanie: Kodowanie liczb
(Próbny egzamin maturalny z informatyki, Arkusz I, OKE Warszawa, październik 2004)

a) Jaką największą dodatnią liczbę dwójkową można przedstawić za pomocą N cyfr (N dowolna liczba naturalna)?

Wpisz odpowiedź. .....

b) Dane są dwie liczby binarne A=(1001 1000)$_2$ i B= (1001)$_2$
Oblicz A+B, A–B, A*B.

Wynik podaj w kodach dwójkowym i szesnastkowym.

Działanie	BIN	HEX
A+B		
A–B		
A*B		

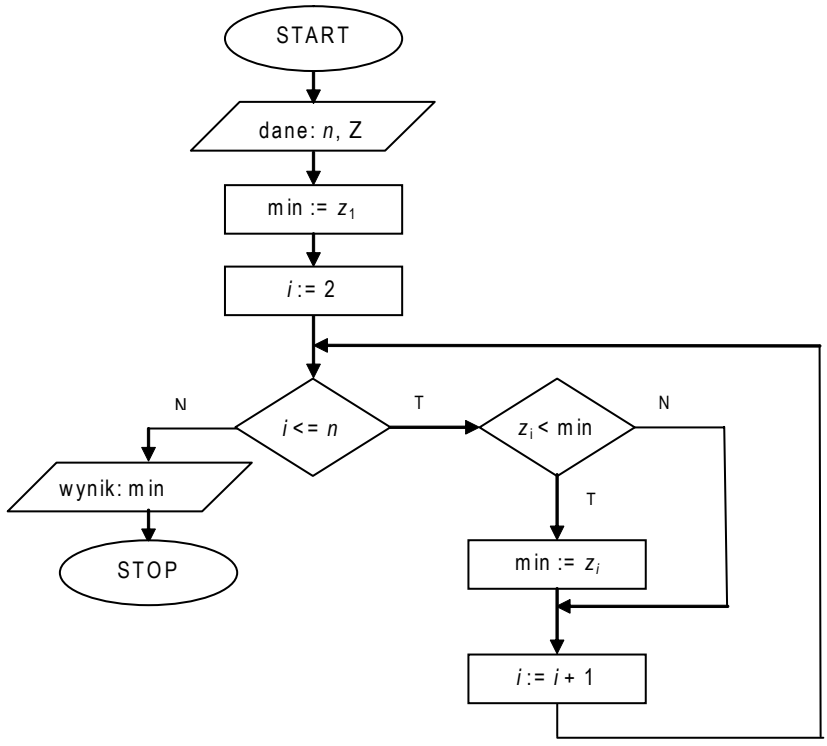
c) Znakiem × zaznacz prawdę lub fałsz dla podanych poniżej określeń. *Tabeli, która następuje, nie zamieszczamy, nie odnosi się bowiem do algorytmiki.*

KOMENTARZ
W punkcie a) należy zauważyć, że największa N-cyfrowa liczba dwójkowa ma same jedynki, a jedynki odpowiadają potęgom liczby 2. Należy je więc zsumować. W punkcie b) należy posłużyć się arytmetyką binarną, a przy zamianie liczb binarnych na szesnastkowe warto skorzystać z zależności między tymi dwoma reprezentacjami – jakiej?

Zadanie: Min-Max
(Próbny egzamin maturalny z informatyki, Arkusz I, UMK Toruń, grudzień 2005)

Dany jest niepusty zbiór $Z = \{z_1, ..., z_n\}$. Dla ułatwienia rozważań zakładamy, że n jest liczbą parzystą. Poniżej przedstawiono, zapewne znany Tobie, algorytm znajdowania min, czyli najmniejszego elementu w zbiorze Z.

ALGORYTM A:



W podobny sposób możemy znaleźć max, czyli największy element w zbiorze Z. Zastosowana w Algorytmie A metoda nazywa się **przeszukiwaniem liniowym**.

Jeśli jednak chcemy znaleźć obydwie wartości min i max jednocześnie, możemy skorzystać z innej metody, dzięki której otrzymujemy algorytm bardziej efektywny ze względu na liczbę wykonywanych porównań elementów:

ALGORYTM B:
W tym algorytmie są wykorzystywane dwa zbiory: N – kandydatów na minimum, i M – kandydatów na maksimum. Na początku oba te zbiory są puste.

1. Połącz elementy zbioru Z w pary. Porównaj elementy w parach: przypuśćmy, że dla pary x, y mamy $x < y$; wtedy dołącz x do zbioru N kandydatów na min, a y do zbioru M, kandydatów na max.
2. Znajdź min w zbiorze N za pomocą algorytmu A.
3. Znajdź max w zbiorze M za pomocą algorytmu A dokonując w nim odpowiednich zmian.

Odpowiedz na pytania:

a) Jakim sposobem przedstawiono opis:
algorytmu A: .....
algorytmu B:.....
b) Wzorując się na algorytmie A podaj łączną liczbę porównań potrzebnych do znalezienia min i max metodą przeszukiwania liniowego. Odpowiedź uzasadnij.
c) Podaj liczbę porównań wykonywanych w poszczególnych krokach algorytmu B oraz w całym algorytmie B.
d) Algorytm B jest algorytmem optymalnym. Co to oznacza? Podaj przykład innego optymalnego algorytmu i problemu, który on rozwiązuje.
e) Podaj nazwę i opisz, na czym polega metod zastosowana w algorytmie B. Podaj przykład innego problemu, do którego rozwiązania można zastosować podobną metodę i oszacuj złożoność opartego na tej metodzie algorytmu.

Zadanie: Co robi ten algorytm?
(Egzamin maturalny z informatyki, Arkusz I, sesja zimowa, CKE, styczeń 2006)

Przeanalizuj działanie poniższego algorytmu, jeżeli tablica A zawiera n liczb całkowitych z zakresu $<0, k>$.

```
1  for i := 0 to k do B[i] := 0;
2  pozycja := 0;
3  for i := 1 to n  do B[A[i]] := B[A[i]] + 1;
4  for i := 0 to k do
5    for j := 1 to B[i] do
6    begin pozycja := pozycja + 1; A[pozycja] := i end;
```

a) Uzupełnij tabelę – określ typy zmiennych: *i*, *j*, *A*, *B*, *pozycja* i opisz ich przeznaczenie:

Zmienna	Typ	Przeznaczenie
i, j		
A		
B		
pozycja		

b) Opisz znaczenie czynności wykonywanych w wierszach o numerach: 3, 4-6.
c) Uzupełnij podane niżej zdania:
Tablica B jest tablicą pomocniczą. Jeśli tablica A zawiera n liczb z zakresu $<0, k>$, to tablica B zawiera ..... liczb z zakresu ..... .
Dla A = [1, 2, 4, 2, 0], po wykonaniu algorytmu, tablica B = [....., ..., ..., ..., ...].
Z uwagi na konieczność zastosowania dodatkowej tablicy, powyższego algorytmu nie można określić mianem ..... .

d) Przeprowadź analizę złożoności czasowej algorytmu i uzupełnij poniższy wniosek.
Załóżmy, że *k* jest ustalone, np. zawsze równe 5. Wówczas:
- złożoność czasowa przedstawionego algorytmu ma charakter: (podkreśl prawidłową odpowiedź):
  liniowy *kwadratowy* *sześcienny* *wykładniczy*;
- symbolicznie złożoność taką można zapisać jako ..... .

KOMENTARZ
Algorytm, o którym jest mowa w tym zadaniu nosi nazwę **porządkowania przez zliczanie**. Jest to szczególna wersja **algorytmu kubełkowego** (**koszykowego**). Złożoność tych algorytmów jest proporcjonalna do liczby elementów w porządkowanym ciągu.

Zadanie: Suma silni
(Egzamin maturalny z informatyki, Arkusz I, 2006)

Pojęcie **silni** dla liczb naturalnych większych od zera definiuje się następująco:
$n! = 1$, dla $n = 1$
$n! = (n-1)!*n$, dla $n > 1$

Rozpatrzmy funkcję $ss(n)$ zdefiniowaną następująco:
$ss(n) = 1! + 2! + 3! + 4! + \ldots + n!$ (*)

gdzie n jest liczbą naturalną większą od zera.

a) Podaj, ile mnożeń trzeba wykonać, aby obliczyć wartość funkcji $ss(n)$, korzystając wprost z podanych wzorów, tzn. obliczając każdą silnię we wzorze (*) oddzielnie.
Uzupełnij poniższą tabelę.

Wartość funkcji	Liczba mnożeń
$ss(3)$	
$ss(4)$	
$ss(n)$	

b) Zauważmy, że we wzorze na $ss(n)$, czynnik 2 występuje w $n-1$ silniach, czynnik 3 w $n-2$ silniach, ..., czynnik n w 1 silni. Korzystając z tej obserwacji przekształć wzór funkcji $ss(n)$ tak, aby można było policzyć wartość $ss(n)$, wykonując dokładnie $n-2$ mnożenia dla każdego $n \geq 2$. Uzupełnij poniższą tabelę (w ostatnim wierszu wypełnij tylko wykropkowane miejsca).

Wartość funkcji	Przekształcony wzór	Liczba mnożeń
$ss(1)$	1	0
$ss(2)$	1+2	0
$ss(3)$	1+2*(1+3)	1
$ss(4)$	1+2*(1+3*(1+4))	2
$ss(5)$		
$ss(n)$	1+2*(1+3*(1+…(n − 2)*(……)…))	$n-2$

Zapisz w wybranej przez siebie notacji (lista kroków, schemat blokowy lub język programowania) algorytm obliczania wartości funkcji $ss(n)$ zgodnie ze wzorem zapisanym przez Ciebie w tabeli. Podaj specyfikację dla tego algorytmu.

KOMENTARZ
Sposób obliczania wartości *ss*(*n*) przypomina schemat Hornera.

Zadanie: Liczby pierwsze
(Egzamin maturalny z informatyki, Arkusz I, 2006)

Poniżej przedstawiono algorytm wyznaczający wszystkie liczby pierwsze z przedziału [2, *N*], wykorzystujący metodę Sita Eratostenesa. Po zakończeniu wykonywania tego algorytmu, dla każdego $i = 2, 3, \ldots, N$, zachodzi $T[i]=0$, jeśli i jest liczbą pierwszą, natomiast $T[i]=1$, gdy i jest liczbą złożoną.

Dane: Liczba naturalna $N \geq 2$.
Wynik: Tablica $T[2...N]$, w której $T[i] = 0$, jeśli i jest liczbą pierwszą, natomiast $T[i]=1$, gdy i jest liczbą złożoną.

Krok 1. Dla $i = 2, 3, ..., N$ wykonuj $T[i] := 0$
Krok 2. $i := 2$
Krok 3. Jeżeli $T[i] = 0$, to przejdź do kroku 4., w przeciwnym razie przejdź do kroku 6.
Krok 4. $j := 2 * i$
Krok 5. Dopóki $j \leq N$ wykonuj
$T[j] := 1$,
$j := j + i$
Krok 6. $i := i + 1$
Krok 7. Jeżeli $i < N$, to przejdź do kroku 3, w przeciwnym razie zakończ wykonywanie algorytmu.

Uwaga: Znak „:=", złożony z dwóch symboli, oznacza instrukcję przypisania.

a) Dane są: liczba naturalna $M \geq 1$ i tablica $A[1...M]$ zawierająca M liczb naturalnych z przedziału [2, *N*]. Korzystając z powyższego algorytmu, zaprojektuj algorytm, wyznaczający te liczby z przedziału [2, *N*], które nie są podzielne przez żadną z liczb $A[1],..., A[M]$. Zapisz go w wybranej przez siebie notacji (lista kroków, schemat blokowy lub język programowania) wraz ze specyfikacją.
b) Do algorytmu opisanego na początku zadania wprowadzamy modyfikacje, po których ma on następującą postać:

Krok 1. Dla $i = 2, 3, ..., N$ wykonuj $T[i] := 0$
Krok 2. $i := 2$
Krok 3. Jeżeli $T[i] = 0$ to przejdź do kroku 4, w przeciwnym razie przejdź do kroku 6
Krok 4. $j := 2 * i$
Krok 5. Dopóki $j \leq N$ wykonuj
$T[j] := T[j] + 1$,
$j := j + i$.
Krok 6. $i := i + 1$
Krok 7. Jeżeli $i < N$, to przejdź do kroku 3, w przeciwnym razie zakończ wykonywanie algorytmu.

Podaj, jakie będą wartości $T[13]$, $T[24]$, $T[33]$ po uruchomieniu tak zmodyfikowanego algorytmu dla $N = 100$.

Podaj, dla jakiej wartości $T[i]$, dla i z przedziału [2, *N*], i jest liczbą pierwszą.

Napisz, jaką własność liczb $i = 2, ..., N$ określają wartości $T[i]$ po wykonaniu tak zmodyfikowanego algorytmu.
c) Sito Eratostenesa służy do wyznaczania wszystkich liczb pierwszych z zadanego przedziału [2, *N*]. Podaj w wybranej przez siebie notacji (lista kroków, schemat blokowy lub język programowania) inny algorytm, który sprawdza, czy podana liczba naturalna $L>1$ jest liczbą pierwszą. Zauważ, że chcemy sprawdzać pierwszość tylko liczby L, natomiast nie jest konieczne sprawdzanie pierwszości liczb mniejszych od L. Przy ocenie Twojego algorytmu będzie brana pod uwagę jego złożoność czasowa.

Specyfikacja:
Dane: Liczba naturalna $L > 1$.
Wynik: Komunikat „Tak", jeśli L jest liczbą pierwszą, komunikat „Nie" w przeciwnym razie.

KOMENTARZ
Zauważ, że algorytm zmodyfikowany w punkcie b) różni się od algorytmu z punktu a) tylko jedną instrukcją w Kroku 5. Zastanów się, co oznacza ta zmiana, czyli jaką interpretację ma wartość $T[i]$ – jest to pytanie z końca punktu b).

Zadanie: Wypłata
(Próbny egzamin maturalny z informatyki, Arkusz I, CKE, grudzień 2006)

Pracownicy pewnego zakładu pracy otrzymują pensje w kwotach będących wielokrotnością 10 złotych. Kasjer, przygotowując wypłatę, przed pobraniem pieniędzy z banku musi obliczyć, ile potrzebuje banknotów o poszczególnych nominałach (10 zł, 20 zł, 50 zł, 100 zł, 200 zł) do zrealizowania wypłaty. Kasjer każdemu pracownikowi chce wypłacić pensję w możliwie najmniejszej liczbie banknotów.

Przyjmijmy, że kwoty wypłat dla poszczególnych pracowników są podane w n-elementowej tablicy WYPŁATY [1...n], gdzie n jest liczbą pracowników zakładu.

Zaproponuj algorytm obliczania liczby banknotów w poszczególnych nominałach, które kasjer musi pobrać z banku. Wynik obliczeń należy umieścić w tablicy LICZBY [1...5], gdzie:

LICZBY[1] to liczba banknotów o nominale 200 zł,
LICZBY[2] to liczba banknotów o nominale 100 zł,
LICZBY[3] to liczba banknotów o nominale 50 zł,
LICZBY[4] to liczba banknotów o nominale 20 zł,
LICZBY[5] to liczba banknotów o nominale 10 zł.

Podaj specyfikację algorytmu i zapisz go w wybranej przez siebie notacji (lista kroków, schemat blokowy, język programowania).

KOMENTARZ
Rozwiązanie tego zadania otrzymujemy jako iterację rozwiązania problemu reszty dla kwot zapisanych w tablicy WYPŁATY

Zadanie: Dziwny ciąg
(Próbny egzamin maturalny z informatyki, Arkusz I, CKE, grudzień 2006)

Rozważamy ciąg liczb naturalnych $D(n)$ dla $n = 0, 1, 2, \ldots$, zdefiniowany następująco:

$D(n) = 1$ dla $n = 0$ lub $n = 1$
$D(n) = D(n \text{ div } 4) + 1$ dla parzystego $n > 1$
$D(n) = D(3n + 1) + 1$ dla nieparzystego $n > 1$

Uwaga: operator div oznacza dzielenie całkowite, np.:

$3 \text{ div } 4 = 0$,
$15 \text{ div } 2 = 7$,
$9 \text{ div } 3 = 3$.

Na przykład:

$D(5) = D(16) + 1 = D(4) + 2 = D(1) + 3 = 4$

a) Korzystając z powyższej definicji oblicz $D(3)$, $D(17)$, $D(31)$. Zapisz poniżej swoje obliczenia.
b) Przedstaw w wybranej przez siebie notacji (lista kroków, schemat blokowy lub język programowania)

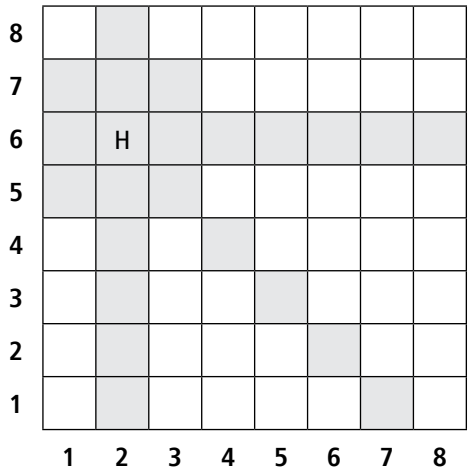
nierekurencyjny algorytm obliczania wartości $D(n)$ dla danej liczby naturalnej n. Podaj specyfikację tego algorytmu.

KOMENTARZ

Algorytm, o który chodzi w punkcie b), polega na obliczaniu kolejnych wartości $D(k)$ dla $k = 1, 2, 3, \ldots, n$ odwołując się do poprzednich wartości tego ciągu, Jest to możliwe, gdyż jeśli n jest liczbą parzystą, to indeks elementu ciągu po prawej stronie wzoru na $D(n)$ maleje co najmniej 4 razy, a jeśli n jest liczbą nieparzystą, to w dwóch kolejnych krokach indeks ten również maleje.

Zadanie: Szachownice
(Egzamin maturalny z informatyki, Arkusz I, CKE, maj 2007)

Zgodnie z regułami gry w szachy, hetman (królowa) może atakować figury ustawione na polach w kolumnie, wierszu oraz dwóch przekątnych przechodzących przez pole, w którym jest ustawiony. O tych polach mówimy, że są atakowane przez hetmana.



Na rysunku hetman stoi w polu (2,6) i atakuje $(7+7+6+3) = 23$ pola. Zostały one zamalowane kolorem szarym.

a) Poniżej znajduje się tabela o wymiarach **5x5**. Korzystając z powyższej obserwacji, uzupełnij pola tabeli wpisując do każdego z nich liczbę pól, które atakowałby hetman znajdujący się w tym polu. Hetman stojący w polu (1,1) atakuje 12 pól planszy.

5					
4					
3					
2					
1	12				
	1	2	3	4	5

b) Określ liczbę atakowanych pól na szachownicy **32x32**, gdy dane są współrzędne ustawienia hetmana dla: (2,2) – wynik = ... (5,4) – wynik = ... (20,18) – wynik = ... (25,30) – wynik = ...

c) Podaj specyfikację i zapisz algorytm (w postaci listy kroków, schematu blokowego lub w języku programowania), który dla dowolnej dodatniej liczby całkowitej $n \leq 50$ i położenia hetmana (x, y) na szachownicy o wymiarach $n \times n$, gdzie $1 \leq x, y \leq n$, pozwoli obliczyć liczbę pól atakowanych przez tego hetmana.

KOMENTARZ
Aby podać algorytm w punkcie c) należy najpierw określić warunek dla pola szachownicy (*i*, *j*), aby było ono szachowane przez hetmana stojącego na pozycji (*x*, *y*). Uwzględnić należy również położenie pola (*x*, *y*) względem brzegów szachownicy.

Zadanie: Sumy
(Egzamin maturalny z informatyki, Arkusz I, CKE, maj 2007)

W tabeli podany jest algorytm, który pozwala obliczyć wartość pewnej *sumy* dla danej dodatniej liczby całkowitej *n*.

1	*p*1←1
2	*suma*←0
3	dla *k*←1...n wykonuj
4	*p*1←*p*1*n
5	*p*2←1
6	dla *i*←1...n wykonuj
7	*p*2←*p*2*k
8	*suma*← *suma* + *p*1 + *p*2

1. Podaj, jaką wartość przyjmie zmienna *p*1 w wyniku działania powyższego algorytmu dla $n = 3$.
   *p*1 = ...
2. Podaj, jaką wartość przyjmie zmienna *p*2 w wyniku działania powyższego algorytmu dla $n = 3$.
   *p*2 = ...
3. Podaj, jaką wartość przyjmie zmienna *suma* w wyniku działania powyższego algorytmu dla $n = 3$.
   suma = ...
4. Zakreślając właściwą odpowiedź, zaznacz, jaką wartość przyjmie zmienna *suma* w wyniku działania powyższego algorytmu.

   a) $\sum_{k=1}^{n}(k^k + n^2)$ b) $\sum_{k=1}^{n}(n^n + k^n)$ c) $\sum_{k=1}^{n}(n^k + k^2)$

   d) $\sum_{k=1}^{n}(n^k + k^n)$ e) $\sum_{k=1}^{n}(n^n + k^k)$

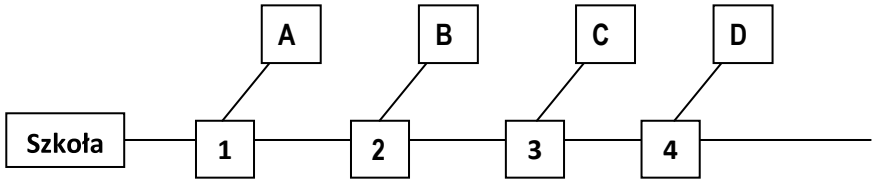
   gdzie $\sum_{k=1}^{n} a_n = a_1 + a_2 + \ldots + a_n$
5. Zakreślając właściwą odpowiedź, podaj, ile wynosi liczba operacji arytmetycznych (dodawań i mnożeń) wykonywanych w czasie realizacji przedstawionego algorytmu.
   a) $3n$ b) $n^2 + 3n$ c) $2^n + n^2$
   d) $n^n + 2^n$ e) $n! + 2^n$
6. Zmień wiersze 6 i 7 w rozważanym algorytmie w taki sposób, aby po jego wykonaniu wartością zmiennej *suma* było $\sum_{k=1}^{n}(n^k + k!)$.

KOMENTARZ
W tym zadaniu należy dobrze zinterpretować działanie instrukcji iteracyjnych.

9. ZADANIA ALGORYTMICZNE Z ARKUSZY II

Zadanie: Eksperyment
(Informator 2002, Arkusz II, CKE, Warszawa 2000))

Przy ulicy prowadzącej do szkoły zostały ustawione cztery kioski A, B, C i D z różnymi artykułami. Rozmieszczenie kiosków przedstawiono na rysunku.



Przeprowadzono eksperyment: wybrany uczeń przez pewien czas odwiedzał codziennie dokładnie jeden z kiosków. Zasada wyboru kiosku była następująca: na skrzyżowaniu uczeń rzucał monetą; wypadnięcie orła oznaczało pójście do kiosku przypisanego temu skrzyżowaniu, wypadnięcie reszki – pójście dalej. Jeśli w ten sposób dotarł do skrzyżowania 4, szedł już bezwarunkowo do kiosku D. Notowano liczbę odwiedzin każdego kiosku.

Do oceny oddajesz: Wydrukowany dokument tekstowy – *raport5* – z rozwiązaniami zadań a), b) i c) oraz plik wymieniony zadaniu a).

a) Powtórz ten eksperyment za pomocą komputera.
   Do oceny oddajesz w *raporcie5*: opis metody, postać danych i postać wyników wykonanego eksperymentu; fragment realizacji w komputerze opisanej metody i otrzymane wyniki eksperymentu prowadzonego przez 10 dni, 100 dni i 1000 dni; plik źródłowy o nazwie ....., zawierający komputerową realizację eksperymentu.
b) Przedstaw na odpowiednich wykresach wyniki doświadczenia dla 10 i 1000 dni trwania Twojego eksperymentu i dla jednych danych z pliku tekstowego Do_zad_5.txt.
   Do oceny oddajesz w *raporcie5*: utworzone wykresy; odpowiedź z uzasadnieniem na pytanie „czy prawidłowo przeprowadzono eksperyment, którego wyniki są zamieszczone w pliku Do_zad_5.txt?".
c) Zakładając, że uczeń korzysta z usług kiosków przez trzy lata, i przyjmując, że koszt jednorazowych zakupów w poszczególnych kioskach wynosi: A – 1,50 zł, B – 0,60 zł, C – 2,50 zł, D – 0,80 zł, zaproponuj rozmieszczenie kiosków z pozycji interesów rodziców i z pozycji interesu właściciela wszystkich czterech kiosków.
   Do oceny oddajesz w *raporcie5*: propozycje rozmieszczenia kiosków i uzasadnienie odpowiedzi.

KOMENTARZ
Program w języku Pascal, rozwiązujący to zadanie, jest podany w pliku Kioski. W rozwiązaniu zadania do wyboru kiosku jest wykorzystana symulacja rzutu monetą. Wyniki eksperymentu są drukowane w postaci tabeli podanej w pliku Do_zad_5.txt, znajdującym się w tym samym folderze. Na podstawie tej tabeli sporządza się *raport5*.

[Pełne rozwiązanie tego zadania jest umieszczone w pliku Eksperyment.pdf.]

Zadanie: Pakujemy plecak
(Próbny egzamin maturalny z informatyki, Arkusz II, OKE Dolny Śląsk, październik 2001)

Wybierasz się na wycieczkę i przed wyjazdem zebrałeś rzeczy, które chciałbyś zapakować do plecaka. Plecak ma jednak ograniczoną wytrzymałość i na pierwszy rzut oka nie wszystkie rzeczy się do niego

zmieszczą. Na szczęście, bez uszczerbku dla przebiegu podróży możesz zrezygnować z niektórych rzeczy – najważniejsze abyś nie przeładował plecaka. Każda rzecz ma dla Ciebie jakąś wartość, proponujemy Ci więc, abyś zaproponował taką zawartość plecaka, która będzie miała największą wartość.

Dla przykładu przypuśćmy, że Twój plecak wytrzymuje obciążenie masą 12 kg i przygotowałeś sześć rzeczy, których masa i wartość są podane w tabeli.

Tabela.
Dane dla przykładowego problemu plecakowego

numer rzeczy	**1**	**2**	**3**	**4**	**5**	**6**
wartość (w zł)	40	30	24	20	35	52
masa (w kg)	10	4	3	2	5	8

a) Które rzeczy z tabeli zapakujesz do plecaka, aby miał on największą wartość, a jednocześnie nie ważył więcej niż 12 kg? Odpowiedź uzasadnij.
b) Na ogół plecak pakuje się wkładając do niego po jednej rzeczy, aż niczego więcej nie można dołożyć. Znasz dopuszczalną masę plecaka oraz masę i wartość poszczególnych rzeczy. W jakiej kolejności będziesz pakował do plecaka rzecz po rzeczy, by go nie przeciążyć i jednocześnie zapakować do niego możliwie najcenniejszą zawartość? Swoją odpowiedź uzasadnij. Sprawdź na powyższym przykładzie, czy za pomocą zaproponowanej strategii pakowania plecaka przez dokładanie do niego po jednej rzeczy otrzymasz rozwiązanie znalezione w punkcie a).
c) Napisz algorytm pakowania plecaka w postaci listy kroków, schematu blokowego lub w języku programowania, który dokłada do plecaka rzecz po rzeczy w kolejności zaproponowanej w punkcie b).

KOMENTARZ
W punkcie b) należało zastosować jedną z zachłannych strategii pakowania plecaka. Omawiamy je szczegółowo w naszej propozycji rozwiązania.

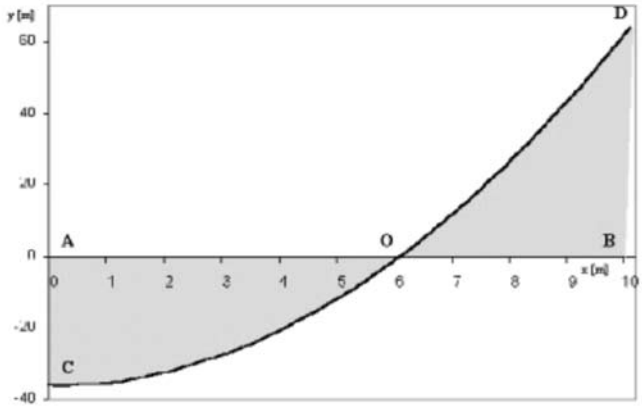
[Pełne rozwiązanie tego zadania jest umieszczone w pliku Plecak.pdf.]

Zadanie: Darń
(Próbny egzamin maturalny z informatyki, Arkusz II, OKE Dolny Śląsk, październik 2001)

Na rysunku zaznaczono szarym kolorem obszar trawnika. Interesuje nas przybliżona wartość jego pola powierzchni. Przyjmijmy, że w punkcie A jest początek układu współrzędnych, zaś krzywa COD jest opisana wzorem $f(x) = x^2 - 36$.

Na dyskietce WYNIKI oddajesz plik o nazwie: ..... zawierający komputerową realizację Twoich obliczeń oraz plik – *RaportD* – zawierający dokument tekstowy.

Do oceny oddajesz: wydrukowany dokument *RaportD*, zawierający rozwiązania zadań a), b), c).



a) Szary obszar z rysunku należy wyłożyć darnią. Przyjmujemy, że darń jest sprzedawana w rolkach o szerokości 0,5 m i długości 4 m. Ile co najmniej rolek należy zakupić, aby pokryć ten obszar dla AB = 10 m. *Do oceny oddajesz* w dokumencie *RaportD* odpowiedź na to pytanie i uzasadnienie tej odpowiedzi.
b) Podaj opis algorytmu obliczania pola szarego obszaru. Przy opisie posłuż się odpowiednim fragmentem realizacji komputerowej swoich obliczeń. Podaj wynik działania tego algorytmu w przypadkach, gdy B znajdzie się w punktach (6, 0) i (20, 0).
c) Przybliżoną metodę obliczania pola powierzchni można przedstawić na rysunku. Przedstaw graficznie ilustrację metody obliczania pola powierzchni z powyższego rysunku. Zaproponuj metodę obliczania wielkości tego pola z niedomiarem i nadmiarem. Rysunki skomentuj. *Do oceny oddajesz* – w dokumencie *RaportD* – dwie ilustracje graficzne i komentarz do nich. Rysunki mogą być odręczne.

KOMENTARZ

Rolki darni w tym zadaniu, to prostokąty z metody prostokątów. Aby w całości pokryć zaznaczony obszar darnią, wysokości tych prostokątów należy brać w jednym z końców przedziału całkowania.

Zauważ, że aby obliczyć wielkość obszaru leżącego pod osią Ox, wartości funkcji w kwadraturach należy brać ze znakiem przeciwnym.

Obliczenia z niedomiarem w metodzie prostokątów odpowiadają braniu prostokątów wewnątrz zaznaczonego obszaru, a obliczenia z nadmiarem – braniu prostokątów, które w całości pokrywają zaznaczony obszar. Odpowiednie ilustracje znajdziesz w edukacyjnym programie Całkowanie.

[Pełne rozwiązanie tego zadania jest umieszczone w pliku Darn.pdf.]

Zadanie: Wartość wyrażenia
(Egzamin maturalny z informatyki w 2002 roku, Arkusz II)

Następujące dwa punkty są definicją prostego wyrażenia arytmetycznego **W** oraz określeniem sposobu obliczania jego wartości *wart*(**W**).

1. dowolna nieujemna, jednocyfrowa liczba całkowita **L** jest prostym wyrażeniem arytmetycznym **W**; wartością takiego wyrażenia jest **L**, czyli *wart*(**L**)=**L**;
2. jeśli $\mathbf{W}_1$ i $\mathbf{W}_2$ są prostymi wyrażeniami arytmetycznymi, a **op** jest jednym ze znaków działania dwuargumentowego: +, – lub *, to

   $\mathbf{W} = \mathbf{W}_1\ \mathbf{W}_2$ **op**

jest również prostym wyrażeniem arytmetycznym i jego wartość wynosi:

wart(**W**) = *wart*($\mathbf{W}_1$) op *wart*($\mathbf{W}_2$).

Przykłady:

Jeśli **W = 6**, to *wart*(**W**) = **6**
Jeśli **W = 28 –**, to *wart*(**W**) = **2 – 8 = –6**
Jeśli **W = 281–***, to *wart*(**W**) = **2 * (8 – 1) = 14**

Do oceny oddajesz: wydrukowany dokument tekstowy – *RaportW* – z rozwiązaniem zadań: a), b) i c). Dodatkowo, umieszczasz na dyskietce WYNIKI: plik o nazwie ....., zawierający źródłowy tekst programu wymienionego w punkcie b) oraz plik o nazwie ....., zawierający *RaportW*.

a) Podaj – w dokumencie *RaportW* – dwa różnej długości przykładowe wyrażenia w postaci określonej powyżej, inne niż podano w całej treści zadania, w których każde z trzech działań występuje przynajmniej raz, i oblicz ich wartości.
b) Napisz program przeznaczony do obliczania wartości dowolnego, prostego wyrażenia **W**, zbudowanego zgodnie z przedstawionymi regułami (patrz przykłady) oraz następującą specyfikacją:

Dane: Wyrażenie **W** jest podane jako ciąg znaków bez spacji pomiędzy kolejnymi znakami. Długość wyrażenia wynosi co najmniej 1 znak i nie więcej niż 80 znaków.

Wynik: Wartość danego wyrażenia **W**.

Zamieść – w dokumencie *RaportW* – treść programu i wyniki jego działania na trzech następujących danych testowych:

9
47–
25+17–*32++

c) Opisz – w dokumencie *RaportW* – algorytm, jakiego użyłeś w swoim programie obliczania wartości wyrażenia **W** oraz wymień struktury danych wykorzystywane w tym programie. W opisie algorytmu posłuż się skomentowanymi fragmentami swojego programu.

KOMENTARZ
Dane do zadania, którym jest wyrażenie **W**, są zdefiniowane rekurencyjnie, dlatego do otrzymania rozwiązania, czyli wartości wyrażenia **W**, wygodnie jest posłużyć się algorytmem rekurencyjnym. Algorytm wyprowadza się wprost z określenia wyniku. Trudność może polegać na posługiwaniu się w algorytmie implementacją wyrażenia **W**, jako napisu, i korzystaniu z tego napisu, jak z tablicy znaków, w której zmienna globalna ma wartość równą indeksowi analizowanego znaku w wyrażeniu-napisie.

[Pełne rozwiązanie tego zadania jest umieszczone w pliku Wyrazenie.pdf.]

Zadanie: Najlepsze sumy, najpopularniejsze elementy
(Informator 2005, Arkusz II, 2005)

Najlepszą sumą ciągu liczb $a_1, a_2, \ldots, a_n$ nazywamy największą wartość wśród sum złożonych z **sąsiednich** elementów tego ciągu. Na przykład dla ciągu: 1, 2, –5, 7 mamy następujące sumy:

1, 1+2 = **3**, 1+2+(–5) = **–2**, 1+2+(–5)+7 = **5**, **2**, 2+(–5) = **–3**, 2+(–5)+7 = **4**, **–5**, –5+7 = **2**, **7**.

Zatem najlepszą sumą jest 7 (zwróć uwagę, że jeden element też uznajemy za sumę).

Do oceny oddajesz:
Na nośniku *WYNIKI* dokument tekstowy *Raport5* zawierający odpowiedzi do punktów a), b), c).
Wykonaj poniższe polecenia.

a) Dany jest następujący ciąg liczb całkowitych: 1, –2, 6, –5, 7, –3. Wyznacz najlepszą sumę dla tego ciągu i wpisz jej wartość: **Najlepsza suma:** .....
Czy na podstawie uzyskanego wyniku można podać wartość najlepszej sumy dla ciągu:
1, –2, 2, 2, 2, –5, 3, 3, 1, –3.
Do oceny oddajesz w dokumencie *Raport5* wartości najlepszej sumy dla ciągu oraz odpowiedź z uzasadnieniem na powyższe pytanie.

b) Zaproponuj algorytm wyznaczania najlepszej sumy dla dowolnego ciągu liczb całkowitych. Na jego podstawie napisz program do obliczenia najlepszych sum ciągów liczb podanych w plikach *dane5-1.txt*, *dane5-2.txt*, *dane5-3.txt* (znajdującym się na nośniku *DANE*).
Do oceny oddajesz także w dokumencie *Raport5*:
- opis algorytmu zawierającego odpowiednie fragmenty kodu Twojego programu,
- wartości najlepszych sum dla poszczególnych plików, które wpisałeś do powyższej tabeli.

c) Wyznacz „najpopularniejszy" element w ciągu, czyli element występujący największą liczbę razy. Zaprojektuj jak najszybszy algorytm wyznaczania najpopularniejszego elementu ciągu oraz oszacuj liczbę wykonywanych przez niego operacji (czas działania) jako funkcję od liczby elementów w ciągu. Zaprogramuj swój algorytm i zastosuj go do ciągów znajdujących się w plikach *dane5-1.txt*, *dane5-2.txt*, *dane5-3.txt*. W przypadku, gdy w ciągu jest więcej niż jeden najpopularniejszy element, jako wynik podajemy dowolny z nich. Na przykład dla ciągu 1, 3, 5, 1, 3 poprawną odpowiedzią jest zarówno 1, jak i 3 (oba elementy występują dwa razy).

Do oceny oddajesz w dokumencie *Raport5:*

- najpopularniejsze elementy w plikach *dane5-1.txt*, *dane5-2.txt*, *dane5-3.txt* umieszczone w tabeli czytelnie prezentującej te wyniki,
- opis algorytmu zawierającego odpowiednie fragmenty kodu Twojego programu oraz oszacowanie czasu jego działania.

KOMENTARZ

W rozwiązaniu punktu c) zadania nie korzysta się z rozwiązania części a) i b). Punkt c) zadania można rozwiązać przynajmniej na trzy sposoby (załóżmy, że ciąg danych zawiera *n* liczb):

Algorytm 1. Bez względu na charakter danych, najpierw porządkujemy dany ciąg, a następnie przeglądając go od lewej do prawej zliczamy, ile jest najwięcej takich samych elementów. Złożoność tego algorytmu wynosi około $n\log n + n$, pierwszy składnik to złożoność porządkowania *n* liczb, a drugi – to złożoność przejrzenia wszystkich elementów ciągu.

Algorytm 2. Najpierw zobaczmy, jaki charakter mają ciągi w danych do tego zadania. Okazuje się, że nawet w najdłuższym z nich jest niewiele elementów różnych. W takim przypadku można skorzystać z algorytmu kubełkowego. Złożoność tego algorytmu jest proporcjonalna do *n*, liczby elementów w ciągu, pod warunkiem, że dane ciągi zawierają niewiele elementów różnych.

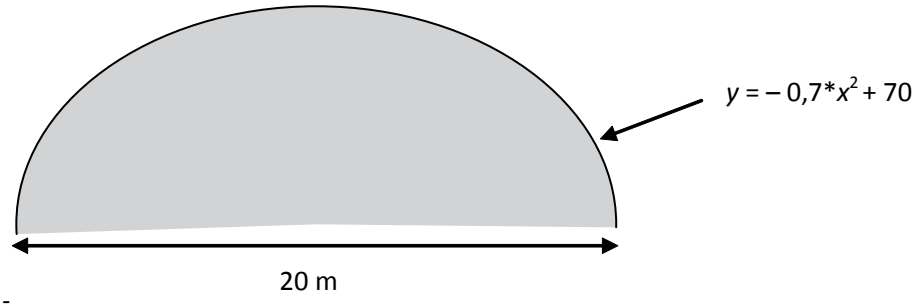
Algorytm 3. Podobnie jak w przypadku Algorytmu 2, ten sposób rozwiązania zależy również od wartości elementów w danych ciągach. W tym algorytmie można posłużyć się arkuszem kalkulacyjnym Excel i jego predefiniowaną funkcją **CZĘSTOŚĆ**. Szczegóły pozostawiamy do samodzielnego wykonania. Złożoność algorytmu w tym przypadku jest trudno określić, gdyż nie wiemy, jak ta funkcja jest zrealizowana w arkuszu. W najgorszym przypadku ta złożoność będzie ograniczona przez złożoność porządkowania.

Zadanie: Kopiec
(Próbny egzamin maturalny z informatyki, Arkusz II, UMK Toruń, grudzień 2005)

Postanowiono usypać kopiec, którego każdy pionowy przekrój poprzeczny opisuje równanie

$y = -0{,}7*x^2 + 70$

Podstawa kopca ma średnicę 20 m. Oblicz, z dokładnością *d* podaną w m^3, ile ziemi potrzeba na usypanie kopca. Liczba rzeczywista *d* ma być czytana z klawiatury.



-

Wykonaj następujące zadania:

a) narysuj schemat blokowy algorytmu do wyznaczania objętości kopca;
b) podaj kryterium użyte do określenia dokładności wyznaczenia objętości kopca;
c) scharakteryzuj zastosowany do rozwiązania algorytm;
d) napisz program wyznaczający, z podaną dokładnością, ile ziemi potrzeba na usypanie kopca. Program czyta dane (liczbę *d*) z klawiatury i wynik zapisuje do pliku kopiec.txt;
e) podaj wynik działania programu otrzymany dla dokładności 5 m^3, 2 m^3:

Dokładność	Wynik [m^3]
5 m^3	
2 m^3	

Do oceny oddajesz plik kopiec.* , zapisany na nośniku WYNIKI, zawierający kompletny program (z funkcjami i procedurami), napisany w wybranym przez Ciebie języku.

Pamiętaj, że ocenie podlega również styl programowania (odpowiednie nazewnictwo zmiennych, stosowanie niezbędnych komentarzy i wcięć w zapisie kodu).

Zadanie: Dodawanie liczb trójkowych
(Próbny egzamin maturalny z informatyki, Arkusz II, CKE, grudzień 2006)

W plikach pary_1.txt i pary_2.txt znajduje się po 50 par dodatnich liczb całkowitych zapisanych w systemie trójkowym – w każdym wierszu jedna para liczb rozdzielonych znakiem odstępu. Każda z liczb ma co najwyżej 64 cyfry.

Napisz program, który dla każdej pary liczb wczytanej z pliku pary_*j*.txt, gdzie $j = 1, 2$, obliczy ich sumę i wynik zapisze w systemie trójkowym w pliku wyniki_*j*.txt, gdzie $j = 1, 2$ – jedna suma w jednym wierszu i bez nieznaczących zer. Liczba w *i*-tym wierszu pliku wyniki_*j*.txt powinna być sumą liczb z *i*-tego wiersza pliku pary_*j*.txt.

Przykład

Gdyby plik pary_*j*.txt zawierał tylko 2 pary liczb:
12 1
22 10
to plik wyniki_*j*.txt miałby postać:
20
102

Do oceny oddajesz pliki wyniki_1.txt i wyniki_2.txt oraz plik o nazwie ..... zawierający pełny kod źródłowy programu.

Zadanie: Liczby superpierwsze
(Egzamin maturalny z informatyki, Arkusz I, CKE, maj 2007))

Liczba **super pierwsza**, to taka liczba naturalna, która spełnia następujące warunki:
- jest liczbą pierwszą,
- suma cyfr tej liczby jest również liczbą pierwszą.

Liczba **super B pierwsza**, oprócz wymienionych dwóch warunków, spełnia warunek trzeci:
- suma cyfr w jej zapisie binarnym jest także liczbą pierwszą.

a) Dla każdego z podanych niżej przedziałów oblicz, ile jest liczb super B pierwszych w tym przedziale. Wyniki wpisz do tabeli. Dodatkowo, w plikach o nazwach *1.txt*, *2.txt* i *3.txt* zapisz wszystkie liczby super B pierwsze odpowiednio z przedziałów 1., 2. i 3., po jednej liczbie w każdym wierszu.

Nr przedziału	Przedział	Liczba wystąpień liczb super B pierwszych w przedziale
1.	<2,1000>	
2.	<100,10000>	
3.	<1000,100000>	

b) Odpowiedz na następujące pytania:
Ile jest liczb w przedziale <100,10000>, których suma cyfr jest liczbą pierwszą?
Czy suma wszystkich liczb super B pierwszych z przedziału <100,10000> jest liczbą pierwszą?

Do oceny oddajesz plik(i) o nazwie(ach) ..... zawierający-(e) komputerową(e) realizację(e) rozwiązania zadania oraz pliki *1.txt*, *2.txt* i *3.txt*.

KOMENTARZ
Przy rozwiązaniu tego zadania przydatny jest algorytm badania, czy dana liczba jest liczbą pierwszą.

10. ALGORYTMIKA, POZIOM PODSTAWOWY (OD 2009)

Zadanie: Algorytm
(poziom podstawowy; Informator o egzaminie maturalnym od 2009 roku. Informatyka, CKE 2007)

Poniżej przedstawiony jest algorytm, działający dla zadanej liczby naturalnej *N* większej od 1.

Krok 1. Zmiennej *M* przypisz wartość *N* – 1.
Krok 2. Sprawdź, czy *M* jest dzielnikiem *N*. Jeśli tak, to wypisz *M* i zakończ wykonywanie algorytmu. W przeciwnym razie przejdź do następnego kroku.
Krok 3. Zmniejsz o 1 wartość zmiennej *M* i przejdź do *Kroku 2.*

a) Co jest wynikiem działania powyższego algorytmu?
b) Czy istnieją takie liczby *N*, dla których wykonywanie algorytmu nigdy się nie zakończy? Odpowiedź: ....
c) Dla jakich liczb *N* wynikiem działania algorytmu jest liczba 1? Odpowiedź uzasadnij.
Ile razy w tym przypadku zostanie wykonany *Krok 2* algorytmu? Odpowiedź: ..... .

Zadanie: Liczby
(poziom podstawowy; Informator o egzaminie maturalnym od 2009 roku. Informatyka, CKE 2007)

W plikach tekstowych o nazwach liczby1.txt oraz liczby2.txt zapisane są liczby naturalne. Każda liczba zapisana jest w oddzielnym wierszu.

Twoim zadaniem jest utworzenie pliku tekstowego o nazwie wynik4.txt, zawierającego odpowiedzi do podpunktów a) – c).

a) Ile jest cyfr w pliku liczby1.txt?
b) Jaka jest najmniejsza liczba w pliku liczby1.txt?
c) Ile liczb występuje jednocześnie w plikach liczby1.txt oraz liczby2.txt?.
d) Załóżmy, że wszystkie liczby z pliku liczby1.txt uporządkowaliśmy od najmniejszej do największej. Jakie liczby znajdują się na pozycjach:
   – 1000 .....
   – 1500 .....

e) Utwórz zestawienie zawierające ilości liczb kończących się odpowiednio cyframi: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Wykonaj wykres ilustrujący otrzymane wyniki. Pamiętaj o czytelnym i pełnym opisie wykresu.

Do oceny oddajesz plik wynik4.txt, plik(i) o nazwie(ach) .....zawierający(e) komputerowe realizacje Twoich obliczeń dla podpunktów a) – d) oraz plik(i) o nazwie(ach) ....zawierający(e) zestawienie i wykres do podpunktu e).

11. ALGORYTMIKA, POZIOM ROZSZERZONY (OD 2009)

Zadanie: Najlepsze sumy
(poziom rozszerzony; Informator o egzaminie maturalnym od 2009 roku. Informatyka, CKE 2007)

Najlepszą sumą ciągu liczb a_1, a_2, ..., a_n nazywamy największą wartość wśród sum złożonych z **kolejnych** elementów tego ciągu. Na przykład dla ciągu: 1, 2, –5, 7 mamy następujące sumy:
1, 1+2 = **3**, 1+2+(–5) = **–2**, 1+2+(–5)+7 = **5**, **2**, 2+(–5) = **–3**, 2+(–5)+7 = **4**, **–5**, –5+7 = **2**, **7**.

Zatem najlepszą sumą jest 7 (zwróć uwagę, że jeden element też uznajemy za sumę). Wykonaj poniższe polecenia.

a) Dany jest następujący ciąg liczb całkowitych: 1, –2, 6, –5, 7, –3. Wyznacz najlepszą sumę dla tego ciągu. Czy na podstawie uzyskanego wyniku można podać wartość najlepszej sumy dla ciągu: 1, –2, 2, 2, 2, –5, 3, 3, 1, –3. Odpowiedź uzasadnij.
b) Zaprojektuj jak najszybszy algorytm wyznaczania najlepszej sumy dla dowolnego ciągu liczb całkowitych. Na jego podstawie napisz program do obliczenia najlepszych sum ciągów liczb podanych w plikach *dane5-1.txt*, *dane5-2.txt*, *dane5-3.txt* (znajdujących się na nośniku *DANE*).

Do oceny oddajesz plik tekstowy *wynik5.txt* zawierający odpowiedzi do podpunktów a) i b), opis algorytmu zaimplementowanego w Twoim programie oraz plik o nazwie ....., zawierający kod źródłowy Twojego programu.

12. ZADANIA RÓŻNE

Zamieszczamy tutaj przykładowe zadania „niealgorytmiczne".

Zadanie: Słowniki
(Informator o egzaminie maturalnym. Informatyka, CKE, Warszawa 2002)

Dane: W trzech niepustych plikach tekstowych są wpisane odpowiadające sobie słowa polskie (plik: pol. txt), angielskie (plik ang.txt) i niemieckie (plik: niem.txt). W jednym wierszu pliku jest wpisane dokładnie jedno słowo, począwszy od początku wiersza.

Do oceny oddajesz: Wydrukowany dokument tekstowy – *raport1* – z rozwiązaniami zadań a), b) i c) oraz plik wymieniony w zadaniu a).

a) Utwórz bazę danych, która zawiera słowa z plików *danych* i z której można wygenerować dowolny słowniczek dwujęzyczny. Utwórz słowniczek niemiecko-polski.
Do oceny oddajesz utworzoną bazę danych (plik) o nazwie ....., w *raporcie1*: opis rekordu tej bazy, opis metody generowania na podstawie bazy słowniczków: niemiecko-polskiego i angielsko-niemieckiego i 20 pierwszych par słów ze słowniczka niemiecko-polskiego; jeśli baza danych jest wynikiem programu napisanego przez Ciebie, zamieść tekst tego programu.
b) Utwórz słowniczek angielsko-polski.
Do oceny oddajesz w *raporcie1*: wszystkie hasła rozpoczynające się na literę p. Każdą parę wstaw w osobnym wierszu, tzn. angielskie_słowo polskie_słowo
W osobnym akapicie, dopisz 1-2 zdania, jak utworzyłeś ten fragment słowniczka.
c) Do oceny oddajesz w *raporcie1*: zestaw 40 odpowiadających sobie par słów niemieckie_słowo angielskie_słowo. Każdą parę wstaw w osobnym wierszu.
Dopisz 1-2 zdania, jak utworzyłeś ten zestaw słów.

Zadanie: Firma
(Egzamin maturalny z informatyki, Arkusz II, CKE, Warszawa 2002)

W pliku firma.txt, na dyskietce DANE, znajdują się dane osób zatrudnionych w pewnej firmie. Dane jednej osoby są umieszczone w osobnym wierszu i zawierają: nazwisko, imię, datę urodzenia (dd-mm-rr), miejsce urodzenia, stanowisko zajmowane w firmie. Dane w wierszu są rozdzielone spacjami. Przykład:

Kowal	Michal	02-12-69	Warszawa	sekretarka
Ciosek	Anna	22-08-64	Krakow	informatyk

Do oceny oddajesz: Wydrukowany dokument tekstowy – *RaportF* – z rozwiązaniami zadań z punktów a), b), c). Dodatkowo, umieszczasz na dyskietce WYNIKI: plik o nazwie ....., wymieniony w punkcie b) i plik o nazwie ....., zawierający *RaportF*.

a) Utwórz zestawienie, które zawiera wiersze z danymi osób z pliku firma.txt urodzonych w miejscowościach, których nazwa zaczyna się na literę B lub G. W *RaporcieF* opisz sposób generowania tego zestawienia oraz umieść w nim wszystkie wiersze tego zestawienia.
b) Utwórz zestawienie danych wszystkich pracowników firmy z ich kodami.
Kod pracownika składa się z ciągu następujących znaków: pierwszej litery nazwiska, pierwszej litery imienia oraz dwóch ostatnich cyfr z roku urodzenia pracownika. Litery występujące w kodzie pracownika mają być małe.
W zestawieniu dla każdego pracownika, w osobnym wierszu, zamieść jego następujące dane: imię, nazwisko, data urodzenia, kod. Postać wiersza zestawienia odczytaj z poniższego przykładu:
Jan Nowak 12-05-69 nj69
W *RaporcieF* opisz sposób generowania tego zestawienia oraz umieść 40 pierwszych wierszy tego zestawienia. Na dyskietce WYNIKI oddaj plik, w formacie tekstowym, o nazwie ....., zawierający to zestawienie.
c) Utwórz zestawienie osób zatrudnionych w firmie na stanowisku grafik, uporządkowane alfabetycznie ze względu na nazwisko. W zestawieniu dla każdego pracownika, w osobnym wierszu, zamieść jego następujące dane: imię, nazwisko. Postać wiersza zestawienia odczytaj z poniższego przykładu:
Jan Nowak
W *RaporcieF* opisz sposób generowania tego zestawienia oraz umieść wszystkie wiersze tego zestawienia.

Zadanie: Polemika
(Egzamin maturalny z informatyki, Arkusz I, CKE, Warszawa 2002)

Przeczytaj załączony tekst:

„O sabotażu komputerowym wspomniano przy przestępstwach dokonywanych z pobudek ideologicznych. Przedmiotem sabotażu mogą być zarówno obiekty materialne (budynki mieszczące ośrodki obliczeniowe, sprzęt i wyposażenie itp.), jak też programy i zbiory."

[Ryszard Czechowski, Piotr Sienkiewicz, Przestępcze oblicza komputerów, PWN, Warszawa 1993]

a) Na czym współcześnie polega sabotaż komputerowy i jakie są jego konsekwencje? Twoja wypowiedź powinna mieć długość 6 zdań (± 2 zdania).
b) Wymień trzy sposoby zainfekowania komputera wirusem i opisz metody zabezpieczenia się w tych przypadkach.
c) Wymień i scharakteryzuj trzy rodzaje źródeł informacji, dostępnych za pomocą komputera.

13. ZADANIA Z PEŁNĄ DOKUMENTACJĄ

Zamieszczamy tutaj kompletny Arkusz egzaminacyjny I, w nim 3 zadania, wraz z pełną informacją dotyczącą punktacji za poszczególne zadania i ich części, jak również model odpowiedzi i schemat oceniania rozwiązań.

Zadanie: Szyfrowanie tekstu
(Informator 2002, Arkusz I, CKE, Warszawa 2000)

Dany jest ciąg znaków, których dziesiętne kody ASCII są w przedziale od 32 do 127 (zob. tabela). **Szyfrowanie z kluczem *n*** polega na zastąpieniu każdego znaku z ciągu znakiem leżącym o *n* pozycji dalej (w tabeli znaków ASCII) od zastępowanego znaku. Przy szyfrowaniu znaku należy postępować w sposób cykliczny, tzn. po znaku o kodzie 127 przechodzimy do znaku o kodzie 32.

a) Podaj znaki i ich kody dziesiętne ASCII otrzymane po zaszyfrowaniu znaku * (o kodzie 42) dla $n = 121$ i $n = 1000$; dla $n = 1000$ podaj sposób otrzymania wyniku.
b) Podaj algorytm, który dla dowolnej liczby naturalnej *n* szyfruje (powyższą metodą) dowolne słowo złożone z *m* znaków (z tabeli).
c) Kod znaku równa się 7A w układzie szesnastkowym. Podaj ten kod w układzie binarnym. Czy znak 1 i liczba 1 mają jednakową reprezentację w komputerze? Odpowiedź uzasadnij.

Znak	Kod		Znak	Kod		Znak	Kod
spacja	32		@	64		`	96
!	33		A	65		a	97
„	34		B	66		b	98
#	35		C	67		c	99
$	36		D	68		d	100
%	37		E	69		e	101
&	38		F	70		f	102
‘	39		G	71		g	103
(	40		H	72		h	104
)	41		I	73		i	105
*	42		J	74		j	106
+	43		K	75		k	107
,	44		L	76		l	108
-	45		M	77		m	109
.	46		N	78		n	110
/	47		O	79		o	111
0	48		P	80		p	112
1	49		Q	81		q	113
2	50		R	82		r	114
3	51		S	83		s	115
4	52		T	84		t	116
5	53		U	85		u	117
6	54		V	86		v	118
7	55		W	87		w	119
8	56		X	88		x	120
9	57		Y	89		y	121
:	58		Z	90		z	122
;	59		[	91		{	123
<	60		\	92		|	124
=	61		]	93		}	125
>	62		^	94		~	126
?	63		_	95		Del	127

Punktacja:

Części zadania	Maks.
a (dla $n = 121$) (dla $n = 1000$)	1
	2
b	5
c	2
Razem:	10

KOMENTARZ

Części a) i b) zadania dotyczą uogólnionego szyfru Cezara, w którym alfabet składa się ze znaków o kodach ASCII miedzy 32 a 127 i przesunięcie alfabetu może być dokonane o dowolną liczbę *n*. A zatem znak o kodzie *k* przy szyfrowaniu z kluczem *n* zostaje zastąpiony przez znak o kodzie $(k + n) \bmod (127 - 32 + 1)$, czyli o kodzie $(k + n) \bmod (96)$.

W części c) zadania, pierwsze polecenie dotyczy zmiany reprezentacji znaku z jednego systemu pozycyjnego na inny. Odpowiedź na pytanie jest negatywna – reprezentacja binarna znaku 1 w kodzie ASCII ma postać 00110001 (= 49), a liczba 1 jest reprezentowana w jednym bajcie jako ciąg: 00000001.

[Pełne rozwiązanie tego zadania jest umieszczone w pliku Szyfrowanie.pdf.]

Zadanie: Algorytm
(Informator 2002, Arkusz I, CKE, Warszawa 2000).

Specyfikacja zadania.

Dane: Uczniowie (co najmniej jeden) ustawieni w dowolnej kolejności. (W tym algorytmie, są wykorzystywane następujące dane o uczniu: nazwisko, imię oraz czas, jaki zabiera mu droga z domu do szkoły – uczeń zna ten czas).

Wynik: ..... .

Algorytm

Krok 1. Zapytaj pierwszego ucznia o jego dane, czyli jak się on nazywa (nazwisko i imię) oraz jak długo idzie do szkoły, i zapamiętaj je.

Krok 2. Powtarzaj *Krok 3* dopóty, dopóki w ustawieniu jest uczeń, któremu jeszcze nie zadałeś pytania. Podaj dane ostatnio zapamiętanego ucznia i zakończ wykonywanie algorytmu.

Krok 3. Zapytaj kolejnego ucznia, jak długo idzie do szkoły. Jeśli krócej, niż zapamiętany uczeń, to zapamiętaj dane o nim na miejscu pamiętania danych o poprzednim uczniu.

a) Uzupełnij specyfikację, czyli sformułuj, jaki jest wynik działania tego algorytmu.
b) Przeformułuj ten algorytm tak, aby sprawdzał, czy wśród uczniów jest ktoś, kto idzie do szkoły dokładnie 10 min. Wynikiem są albo dane o uczniu, który idzie do szkoły 10 min., albo informacja, że takiego ucznia nie ma.
c) Podaj algorytm dla specyfikacji:
   Dane: Uczniowie ustawieni według niemalejących czasów dojścia do szkoły, czyli od najkrócej idącego do szkoły do najdłużej idącego.
   Wynik: Dane o uczniu, który idzie do szkoły dokładnie 10 min., albo informacja, że takiego ucznia nie ma.

Punktacja:

Części zadania	Maks.
a	2
b	3
c	5
Razem:	10

KOMENTARZ

Odnośnie części a), algorytm podany w treści zadania służy do znajdowania ucznia, który najkrócej idzie do szkoły, jest to więc algorytm, służący do znajdowania minimum w ciągu, złożonym z czasów dojścia do szkoły.

W części b) należy otrzymać algorytm poszukiwania czasu równego 10 min w ciągu czasów dojścia do szkoły – stosujemy algorytm przeszukiwania ciągu nieuporządkowanego.

Część c) dotyczy poszukiwania 10 min w uporządkowanym ciągu czasów. W rozwiązaniu tej części najwyżej jest punktowany algorytm poszukiwania przez połowienie, niżej – inny algorytm wykorzystujący uporządkowanie, a najniżej algorytm, w którym w ogóle nie bierze się pod uwagę uporządkowania elementów.

[Pełne rozwiązanie tego zadania jest umieszczone w pliku Algorytm.pdf.]

Zadanie: Polemika
(Informator o egzaminie maturalnym. Informatyka, CKE, Warszawa 2002)

Przeczytaj załączony tekst:

„Dzięki konstruktorom sprzętu i autorom oprogramowania możemy łączyć się ze światem i przesyłać informacje, lecz dla większości bezmyślnych internautów liczy się samo żeglowanie po Internecie, a więc nieograniczona możliwość komunikacji. Podobni są do radioamatorów, „zdobywców fal eteru" – cieszą się, widząc zdjęcie kota Billa Clintona, jak niegdyś ich poprzednicy wyrażali entuzjazm z ledwie słyszalnego głosu kogoś z Kuala Lumpur lub Adelajdy. Czytelnikom książek natomiast nie wystarczy radość komunikowania się, ponieważ chcą w Internecie znaleźć coś interesującego".

[Reinhard Kaiser, Literackie spacery po Interneciel]

a) Jakie jest Twoje stanowisko wobec poruszonego w tekście problemu? Wypowiedź powinna zawierać 6 (±2) zdań i nawiązywać jawnie do zamieszczonego tekstu.
b) Wymień i objaśnij trzy pojęcia informatyczne mające związek z przeczytanym tekstem.
c) Następujące terminy wiążą się ze sposobami wymiany informacji w sieci komputerowej: FTP, grupa dyskusyjna, lista dyskusyjna, WWW, IRC. Wybierz dwa z nich i opisz, na czym ta wymiana polega.

Punktacja:

Części zadania	Maks.
a	3
b	3
c	4
Razem:	10

MODEL ODPOWIEDZI I SCHEMAT OCENIANIA ARKUSZA EGZAMINACYJNEGO I (2002)

Zasady oceniania

- Za rozwiązanie zadań z arkusza I można uzyskać maksymalnie 40% całkowitej liczby punktów.
- Model odpowiedzi uwzględnia jej zakres merytoryczny, a nie jest ścisłym wzorcem sformułowania (poza odpowiedziami jednowyrazowymi i do zadań zamkniętych).
- Za odpowiedzi do poszczególnych zadań przyznaje się pełne punkty.
- Za zadania otwarte, za które można przyznać jeden punkt, przyznaje się punkt wyłącznie za odpowiedź w pełni poprawną.
- Za zadania otwarte, za które można przyznać więcej niż jeden punkt, przyznaje się tyle punktów, ile prawidłowych elementów odpowiedzi (zgodnie z wyszczególnieniem w kluczu) przedstawił zdający.

Model odpowiedzi i schemat punktowania zadań z arkusza I

Numer zadania	Numer punktu	Oczekiwana odpowiedź	Maksymalna punktacja za część zadania	Maksymalna punktacja za zadanie
1	a	Za poprawną odpowiedź dla $n = 121$ – 1 punkt. Za poprawną odpowiedź dla $n = 1000$ bez uzasadnienia – 1 punkt, za podanie sposobu otrzymania wyniku – 1 punkt.	3	10
	b	Za opisanie metody szyfrowania jednego znaku dla dowolnego *n*, ale nie w postaci algorytmu – 2 punkty. Za to samo z usterką – 1 punkt. Za opisanie metody zaszyfrowania wszystkich znaków, ale nie w postaci algorytmu – 1 punkt. Za podanie algorytmu szyfrowania jednego znaku dla dowolnego *n* – 3 punkty. Za to samo z usterką – 1 lub 2 punkty. Za podanie algorytmu zaszyfrowania wszystkich znaków – 2 punkty.	5	
	c	Za podanie binarnego kodu – 1 punkt. Za poprawną odpowiedź na pytanie i uzasadnienie – 1 punkt.	2	
2	a	Za niezbyt precyzyjną specyfikację – 1 punkt. Za poprawną specyfikację – 2 punkty.	2	10
	b	Za opisanie poprawnej metody (bez wydzielenia kroków) – 2 punkty. Za podanie algorytmu z usterką – 1 punkt. Za poprawny algorytm – 3 punkty.	3	
	c	Za algorytm szukania bez uwzględnienia porządku danych – 1 punkt. Za liniowe szukanie w uporządkowanym ciągu i przerwanie szukania po znalezieniu – 2 punkty. Za opis szukania binarnego z usterką – 3 punkty. Za algorytm szukania binarnego – 4 punkty. Za podanie efektywności algorytmu poszukiwania – 1 punkt.	5	
3	a	Za 6 (±2) zdań – 1 punkt. Za nawiązanie bezpośrednio do tekstu – 1 punkt. Za wypowiedź poprawną merytorycznie i językowo – 1 punkt.	3	10
	b	Za podanie pojęcia i objaśnienia – 1 punkt (3 pojęcia, to 3 punkty).	3	
	c	Za intuicyjne wyjaśnienie jednego pojęcia – 1 punkt. Za jedno pojęcie wyjaśnione poprawnie z punktu widzenia informatycznego – 2 punkty. (2 pojęcia, to od 0 do 4 punktów).	4	

LITERATURA

1. *Algorytmika:* http://www.wsip.com.pl/algorytmika/
2. Centralna Komisja Egzaminacyjna: http://www.cke.edu.pl/
3. Cormen T.H., Leiserson C.E., Rivest R.L., *Wprowadzenie do algorytmów*, WNT, Warszawa 1997

4. Gurbiel E., Hardt-Olejniczak G., Kołczyk E., Krupicka H., Sysło M.M., *Technologia informacyjna. Podręcznik dla liceum ogólnokształcącego, liceum profilowanego i technikum; Technologia informacyjna. Poradnik dla nauczycieli szkół ponadgimnazjalnych.* WSiP S.A., Warszawa 2002.
5. Gurbiel E., Hardt-Olejniczak G., Kołczyk E., Krupicka H., Sysło M.M., *Informatyka. Podręcznik dla liceum ogólnokształcącego*, WSiP S.A., Warszawa 2002 (Część 1), 2003(Część 2).
6. Gurbiel E., Hardt-Olejniczak G., Kołczyk E., Krupicka H., Sysło M.M., *Informatyka. Poradnika dla nauczycieli*, WSiP S.A., Warszawa 2004.
7. Harel D., *Algorytmika. Rzecz o istocie informatyki*, WNT, Warszawa 1992
8. Knuth D.E., *Sztuka programowania*, Tomy 1 – 3, WNT, Warszawa 2003
9. *Matura:* http://www.wsip.com.pl/serwisy/ti/matura/
10. Sysło M.M., *Algorytmy*, WSiP, Warszawa 1997
11. Sysło M.M., *Piramidy, szyszki i inne konstrukcje algorytmiczne*, WSiP, Warszawa 1998. Rozdziały tej książki są zamieszczone na stronie: http://www.wsipnet.pl/kluby/informatyka_ekstra.php?k=69
12. Wirth N., *Algorytmy + struktury danych = programy*, WNT, Warszawa 1980

W projekcie **Informatyka +**, poza wykładami i warsztatami, przewidziano następujące działania:

- 24-godzinne kursy dla uczniów w ramach modułów tematycznych
- 24-godzinne kursy metodyczne dla nauczycieli, przygotowujące do pracy z uczniem zdolnym
- nagrania 60 wykładów informatycznych, prowadzonych przez wybitnych specjalistów i nauczycieli akademickich
- konkursy dla uczniów, trzy w ciągu roku
- udział uczniów w pracach kół naukowych
- udział uczniów w konferencjach naukowych
- obozy wypoczynkowo-naukowe.

Szczegółowe informacje znajdują się na stronie projektu **www.informatykaplus.edu.pl**