Województwo lubuskie

Informatyka

Sprawozdanie z egzaminu maturalnego w roku 2014

Opracowanie
Andrzej Kudelski (Centralna Komisja Egzaminacyjna)
Dorota Jurdzińska (Okręgowa Komisja Egzaminacyjna we Wrocławiu)

Redakcja
dr Marcin Smolik (Centralna Komisja Egzaminacyjna)
dr Wioletta Kozak (Centralna Komisja Egzaminacyjna)

Opracowanie techniczne
Bartosz Kowalewski (Centralna Komisja Egzaminacyjna)

Współpraca
Beata Dobrosielska (Centralna Komisja Egzaminacyjna)
Agata Wiśniewska (Centralna Komisja Egzaminacyjna)
Wydziały Badań i Analiz okręgowych komisji egzaminacyjnych

Opracowanie dla województwa lubuskiego

Okręgowa Komisja Egzaminacyjna w Poznaniu

Izabela Szafrańska
Jerzy Kraczkowski
Michał Pawlak
Jacek Pietrzak

Centralna Komisja Egzaminacyjna
ul. Józefa Lewartowskiego 6, 00-190 Warszawa
tel. 022 536 65 00, fax 022 536 65 04
e-mail: ckesekr@cke.edu.pl
www.cke.edu.pl

Poziom podstawowy

1. Opis arkusza

Egzamin maturalny z informatyki na poziomie podstawowym składał się z dwóch części. Każda z części zawierała trzy zadania. Część pierwsza polegała na rozwiązywaniu zadań bez korzystania z komputera. W części drugiej zdający rozwiązywał zadania przy użyciu komputera i wybranych wcześniej przez siebie narzędzi (język programowania, programy użytkowe).

2. Dane dotyczące populacji zdających

Tabela 1. Zdający rozwiązujący zadania w arkuszu standardowym

Liczba zdających		**71**
Zdający rozwiązujący zadania w arkuszu w wersji standardowej	z liceów ogólnokształcących	29
	z liceów profilowanych	0
	z techników	42
	z liceów uzupełniających	0
	z techników uzupełniających	0
	ze szkół publicznych	71
	ze szkół niepublicznych	0
	ze szkół na wsi	0
	ze szkół w miastach do 20 tys. mieszkańców	25
	ze szkół w miastach od 20 tys. do 100 tys. mieszkańców	11
	ze szkół w miastach powyżej 100 tys. mieszkańców	35
	kobiety	9
	mężczyźni	62

Dane w tabeli dotyczą tegorocznych absolwentów.

Tabela 2. Zdający rozwiązujący zadania w arkuszach dostosowanych

Zdający rozwiązujący zadania w arkuszach w wersji dostosowanej	z autyzmem, w tym z zespołem Aspergera	0
	słabowidzący	0
	niewidomi	0
	słabosłyszący	0
	niesłyszący	0
	Ogółem	**0**

Do egzaminu przystąpili również absolwenci z lat ubiegłych, którzy dotychczas nie uzyskali świadectwa dojrzałości, oraz tacy, którzy uzyskali świadectwo dojrzałości we wcześniejszych latach, a w maju 2014 r. przystąpili ponownie do egzaminu maturalnego w celu podwyższenia wyniku egzaminacyjnego albo uzyskania wyniku z informatyki jako nowego przedmiotu.

3. Przebieg egzaminu

Tabela 3. Informacje dotyczące przebiegu egzaminu

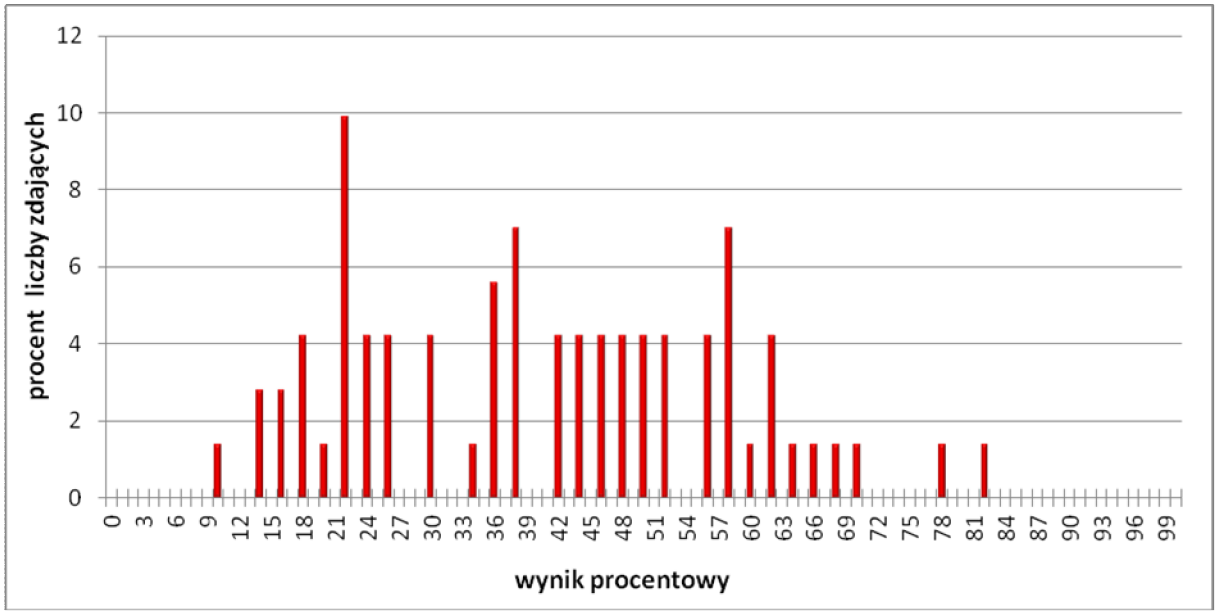
Termin egzaminu			20 maja 2014 r.
Czas trwania egzaminu dla arkusza standardowego			Część I: 75 minut Część II: 120 minut
Liczba szkół			25
Liczba zespołów egzaminatorów*			1
Liczba egzaminatorów*			17
Liczba obserwatorów[1] (§ 143)			1
Liczba unieważnień[1]	w przypadku:		
	§ 99 ust.1	stwierdzenia niesamodzielnego rozwiązywania zadań przez zdającego	0
		wniesienia lub korzystania przez zdającego w sali egzaminacyjnej z urządzenia telekomunikacyjnego	0
		zakłócenia przez zdającego prawidłowego przebiegu części egzaminu w sposób utrudniający pracę pozostałym zdającym	0
	§ 99 ust.2	stwierdzenia podczas sprawdzania pracy niesamodzielnego rozwiązywania zadań przez zdającego	0
	§ 146 ust. 3	stwierdzenia naruszenia przepisów dotyczących przeprowadzenia egzaminu	0
Liczba wglądów[1] (§ 107)			0

*Dane dotyczą obu poziomów egzaminu (podstawowego i rozszerzonego) łącznie w całym Okręgu.

[1] Na podstawie rozporządzenia Ministra Edukacji Narodowej z dnia 30 kwietnia 2007 r. w sprawie warunków i sposobu oceniania, klasyfikowania i promowania uczniów i słuchaczy oraz przeprowadzania sprawdzianów i egzaminów w szkołach publicznych (Dz.U. nr 83, poz. 562, ze zm.)

4. Podstawowe dane statystyczne

Wyniki zdających



Wykres 1. Rozkład wyników zdających

Tabela 4. Wyniki zdających – parametry statystyczne

Liczba zdających	Minimum (%)	Maksimum (%)	Mediana (%)	Modalna (%)	Średnia (%)	Odchylenie standardowe (%)
71	10	82	42	22	40,62	17,22

Dane w tabeli dotyczą tegorocznych absolwentów.

Poziom wykonania zadań

Tabela 5. Poziom wykonania zadań

Nr zad.	Obszar standardów	Sprawdzana umiejętność	Poziom wykonania zadania (%)
1a	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.7)	53
1b	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.7)	49
1c	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.7)	43
2a	Korzystanie z informacji	Zastosowanie podstawowych algorytmów w rozwiązywaniu problemów informatycznych (II.5)	62
2b	Korzystanie z informacji	Zastosowanie podstawowych algorytmów w rozwiązywaniu problemów informatycznych (II.5)	57
2c	Korzystanie z informacji	Zastosowanie podstawowych algorytmów w rozwiązywaniu problemów informatycznych (II.5)	31
3a	Wiadomości i rozumienie	Znajomość podstawowych pojęć związanych z relacyjnymi bazami danych (I.10)	13
3b	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.7)	89
3c	Wiadomości i rozumienie	Znajomość podstawowej terminologii związanej z sieciami komputerowymi (I.3)	90
3d	Wiadomości i rozumienie	Znajomość podstawowej terminologii związanej z sieciami komputerowymi (I.3)	51
3e	Korzystanie z informacji	Posługiwanie się typowymi programami użytkowymi (II.1)	71
3f	Wiadomości i rozumienie	Znajomość reprezentowania wiadomości w komputerze (I.6)	87
3g	Korzystanie z informacji	Wykonywanie obliczeń przy pomocy wbudowanych funkcji i formuł (II.1)	66
4a	Korzystanie z informacji	Dobranie właściwego programu (użytkowego lub własnoręcznie napisanego) do rozwiązania zadania (II.6) Posługiwanie się typowymi programami użytkowymi (II.1)	79
4b	Korzystanie z informacji	Dobranie właściwego programu (użytkowego lub własnoręcznie napisanego) do rozwiązania zadania (II.6) Posługiwanie się typowymi programami użytkowymi (II.1)	66
4c	Korzystanie z informacji	Dobranie właściwego programu (użytkowego lub własnoręcznie napisanego) do rozwiązania zadania (II.6) Posługiwanie się typowymi programami użytkowymi (II.1)	22
5a	Tworzenie informacji Korzystanie z informacji	Sformułowanie informatycznego rozwiązania problemu przez dobór algorytmu i zaimplementowanie w wybranym języku programowania (III.2) Wykorzystanie wybranego środowiska programistycznego do zapisania, uruchomienia i testowania programu (II.2)	7
5b	Tworzenie informacji Korzystanie z informacji	Sformułowanie informatycznego rozwiązania problemu przez dobór algorytmu i zaimplementowanie w wybranym języku programowania (III.2) Wykorzystanie wybranego środowiska programistycznego do zapisania, uruchomienia i testowania programu (II.2)	4

5c	Tworzenie informacji Korzystanie z informacji	Sformułowanie informatycznego rozwiązania problemu przez dobór algorytmu i zaimplementowanie do w wybranym języku programowania (III.2) Wykorzystanie wybranego środowiska programistycznego do zapisania, uruchomienia i testowania programu (II.2)	1
6a	Tworzenie informacji Korzystanie z informacji	Zaprojektowanie bazy danych i wykorzystanie do jej realizacji systemu bazy danych (III.3) Zastosowanie metod wyszukiwania i przetwarzania informacji w relacyjnych bazach danych (II.4)	51
6b	Tworzenie informacji Korzystanie z informacji	Zaprojektowanie bazy danych i wykorzystanie do jej realizacji systemu bazy danych (III.3) Zastosowanie metod wyszukiwania i przetwarzania informacji w relacyjnych bazach danych (II.4)	48
6c	Tworzenie informacji Korzystanie z informacji	Zaprojektowanie bazy danych i wykorzystanie do jej realizacji systemu bazy danych (III.3) Zastosowanie metod wyszukiwania i przetwarzania informacji w relacyjnych bazach danych (II.4)	40
6d	Tworzenie informacji Korzystanie z informacji	Zaprojektowanie bazy danych i wykorzystanie do jej realizacji systemu bazy danych (III.3) Zastosowanie metod wyszukiwania i przetwarzania informacji w relacyjnych bazach danych (II.4)	42

Poziom rozszerzony

1. Opis arkusza

Egzamin maturalny z informatyki na poziomie rozszerzonym składał się z dwóch części. Każda z części zawierała trzy zadania. Część pierwsza polegała na rozwiązywaniu zadań bez korzystania z komputera. W części drugiej zdający rozwiązywał zadania przy użyciu komputera i wybranych wcześniej przez siebie narzędzi (język programowania, programy użytkowe).

2. Dane dotyczące populacji zdających

Tabela 1. Zdający rozwiązujący zadania w arkuszu standardowym

Liczba zdających		**52**
Zdający rozwiązujący zadania w arkuszu w wersji standardowej	z liceów ogólnokształcących	17
	z liceów profilowanych	0
	z techników	35
	z liceów uzupełniających	0
	z techników uzupełniających	0
	ze szkół publicznych	51
	ze szkół niepublicznych	1
	ze szkół na wsi	0
	ze szkół w miastach do 20 tys. mieszkańców	11
	ze szkół w miastach od 20 tys. do 100 tys. mieszkańców	13
	ze szkół w miastach powyżej 100 tys. mieszkańców	28
	kobiety	3
	mężczyźni	49

Dane w tabeli dotyczą tegorocznych absolwentów.

Tabela 2. Zdający rozwiązujący zadania w arkuszach dostosowanych

Zdający rozwiązujący zadania w arkuszach w wersji dostosowanej	z autyzmem, w tym z zespołem Aspergera	0
	słabowidzący	0
	niewidomi	0
	słabosłyszący	0
	niesłyszący	0
	Ogółem	**0**

Do egzaminu przystąpili również absolwenci z lat ubiegłych, którzy dotychczas nie uzyskali świadectwa dojrzałości, oraz tacy, którzy uzyskali świadectwo dojrzałości we wcześniejszych latach, a w maju 2014 r. przystąpili ponownie do egzaminu maturalnego w celu podwyższenia wyniku egzaminacyjnego albo uzyskania wyniku z informatyki jako nowego przedmiotu.

3. Przebieg egzaminu

Tabela 3. Informacje dotyczące przebiegu egzaminu

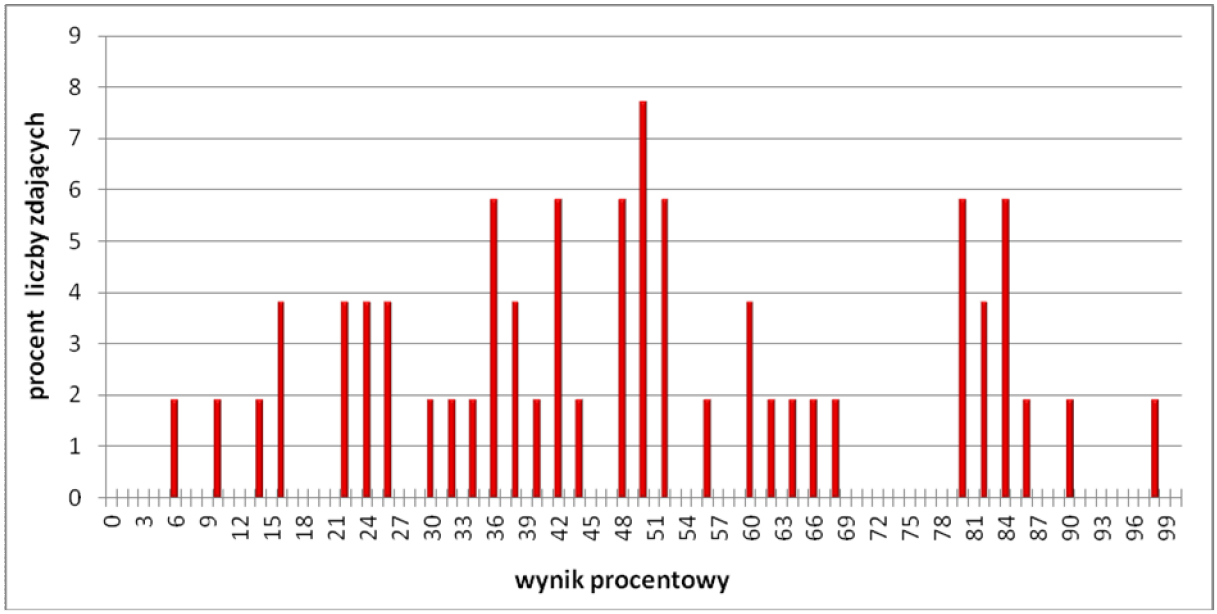
<table>
<tr><td colspan="3">Termin egzaminu</td><td>20 maja 2014 r.</td></tr>
<tr><td colspan="3">Czas trwania egzaminu dla arkusza standardowego</td><td>Część I: 90 minut
Część II: 150 minut</td></tr>
<tr><td colspan="3">Liczba szkół</td><td>17</td></tr>
<tr><td colspan="3">Liczba zespołów egzaminatorów*</td><td>1</td></tr>
<tr><td colspan="3">Liczba egzaminatorów*</td><td>17</td></tr>
<tr><td colspan="3">Liczba obserwatorów[1] (§ 143)</td><td>0</td></tr>
<tr><td rowspan="5">Liczba unieważnień[1]</td><td colspan="2">w przypadku:</td><td></td></tr>
<tr><td rowspan="3">§ 99 ust. 1</td><td>stwierdzenia niesamodzielnego rozwiązywania zadań przez zdającego</td><td>0</td></tr>
<tr><td>wniesienia lub korzystania przez zdającego w sali egzaminacyjnej z urządzenia telekomunikacyjnego</td><td>0</td></tr>
<tr><td>zakłócenia przez zdającego prawidłowego przebiegu części egzaminu w sposób utrudniający pracę pozostałym zdającym</td><td>0</td></tr>
<tr><td>§ 99 ust. 2</td><td>stwierdzenia podczas sprawdzania pracy niesamodzielnego rozwiązywania zadań przez zdającego</td><td>0</td></tr>
<tr><td>§ 146 ust. 3</td><td>stwierdzenia naruszenia przepisów dotyczących przeprowadzenia egzaminu</td><td>0</td></tr>
<tr><td colspan="3">Liczba wglądów[1] (§ 107)</td><td>0</td></tr>
</table>

*Dane dotyczą obu poziomów egzaminu (podstawowego i rozszerzonego) łącznie w całym Okręgu.

[1] Na podstawie rozporządzenia Ministra Edukacji Narodowej z dnia 30 kwietnia 2007 r. w sprawie warunków i sposobu oceniania, klasyfikowania i promowania uczniów i słuchaczy oraz przeprowadzania sprawdzianów i egzaminów w szkołach publicznych (Dz.U. nr 83, poz. 562, ze zm.)

4. Podstawowe dane statystyczne

Wyniki zdających



Wykres 1. Rozkład wyników zdających

Tabela 4. Wyniki zdających – parametry statystyczne

Liczba zdających	Minimum (%)	Maksimum (%)	Mediana (%)	Modalna (%)	Średnia (%)	Odchylenie standardowe (%)
52	6	98	48	50	49,27	23,42

Dane w tabeli dotyczą tegorocznych absolwentów.

Poziom wykonania zadań

Tabela 5. Poziom wykonania zadań

Nr zad.	Obszar standardów	Sprawdzana umiejętność	Poziom wykonania zadania (%)
1a	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	72
1b	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	28
1c	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	42
2a	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	49
2b	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	54
2c	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	33
3a	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	75
3b	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	34
3c	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	81
3d	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	49
3e	Wiadomości i rozumienie	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4)	81
4a	Korzystanie z informacji Tworzenie informacji	Modelowanie zjawisk i procesów z różnych dziedzin życia (II.3) Wykorzystanie metod informatyki do rozwiązywania problemów (III.2)	63
4b	Korzystanie z informacji Tworzenie informacji	Modelowanie zjawisk i procesów z różnych dziedzin życia (II.3) Wykorzystanie metod informatyki do rozwiązywania problemów (III.2)	42
4c	Korzystanie z informacji Tworzenie informacji	Modelowanie zjawisk i procesów z różnych dziedzin życia (II.3) Wykorzystanie metod informatyki do rozwiązywania problemów (III.2)	47
4d	Korzystanie z informacji Tworzenie informacji	Modelowanie zjawisk i procesów z różnych dziedzin życia (II.3) Sformułowanie informatycznego rozwiązania problemu przez dobór algorytmu i odpowiednich typów oraz struktur danych i zaimplementowanie go w wybranym języku programowania (III.2)	474
5a	Wiadomości i zrozumienie Korzystanie z informacji	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4) Sformułowanie informatycznego rozwiązania problemu przez dobór algorytmu i zaimplementowanie w wybranym języku programowania (III.2)	28
5b	Wiadomości i zrozumienie Korzystanie z informacji	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4) Sformułowanie informatycznego rozwiązania problemu przez dobór algorytmu i zaimplementowanie w wybranym języku programowania (III.2)	27

5c	Wiadomości i zrozumienie Korzystanie z informacji	Znajomość podstawowych technik algorytmicznych i algorytmów (I.4) Sformułowanie informatycznego rozwiązania problemu przez dobór algorytmu i zaimplementowanie w wybranym języku programowania (III.2)	33
6a	Tworzenie informacji Korzystanie z informacji	Analiza problemu i zbioru danych, którego rozwiązanie wymaga zaprojektowania i utworzenia relacyjnej bazy danych z uwzględnieniem zawartych informacji (III.3) Wyszukiwanie informacji w bazie danych stosując różne techniki oraz zastosowanie metod optymalizujących wyszukiwanie (II.1)	72
6b	Tworzenie informacji Korzystanie z informacji	Analiza problemu i zbioru danych, którego rozwiązanie wymaga zaprojektowania i utworzenia relacyjnej bazy danych z uwzględnieniem zawartych informacji (III.3) Wyszukiwanie informacji w bazie danych stosując różne techniki oraz zastosowanie metod optymalizujących wyszukiwanie (II.1)	71
6c	Tworzenie informacji Korzystanie z informacji	Analiza problemu i zbioru danych, którego rozwiązanie wymaga zaprojektowania i utworzenia relacyjnej bazy danych z uwzględnieniem zawartych informacji (III.3) Wyszukiwanie informacji w bazie danych stosując różne techniki oraz zastosowanie metod optymalizujących wyszukiwanie (II.1)	41
6d	Tworzenie informacji Korzystanie z informacji	Analiza problemu i zbioru danych, którego rozwiązanie wymaga zaprojektowania i utworzenia relacyjnej bazy danych z uwzględnieniem zawartych informacji (III.3) Wyszukiwanie informacji w bazie danych stosując różne techniki oraz zastosowanie metod optymalizujących wyszukiwanie (II.1)	47

Komentarz

1. Analiza jakościowa zadań

Jednym z trzech obszarów obejmujących standardy wymagań egzaminacyjnych są *Wiadomości i rozumienie*, który okazał się najłatwiejszy na poziomie podstawowym (poziom wykonania: 51,4%). Zdający powinien znać i rozumieć podstawowe pojęcia i metody związane z informatyką i technologią informacyjną. Na poziomie podstawowym jest to między innymi znajomość podstawowej terminologii związanej z sieciami komputerowymi i relacyjnymi bazami danych, sposobów reprezentowania informacji w komputerze oraz podstawowych własności algorytmów (pojęcie algorytmu, różne sposoby jego zapisu, zgodność algorytmu ze specyfikacją), a także różnych technik algorytmicznych i algorytmów. Na poziomie rozszerzonym (poziom wykonania: 50,3%) między innymi wymagana jest znajomość systemów liczbowych, zasad programowania obiektowego i wybranych struktura danych (i ich realizacji) oraz lista algorytmów i technik algorytmicznych. Należy zauważyć, że umiejętności z tego obszaru są głównie sprawdzane w pierwszej części egzaminu.

W obszarze dotyczącym *Korzystania z informacji* zdający powinien zastosować posiadaną wiedzę do rozwiązywania zadań zarówno teoretycznych, jak i praktycznych. W zakresie algorytmiki zdający powinien stosować podstawowe algorytmy i struktury danych w rozwiązywaniu problemów informatycznych oraz wykonywać kolejne etapy prowadzące do otrzymania rozwiązania zadania, od sformułowania specyfikacji problemu do zapisu właściwej części algorytmu w jednej z wybranych notacji, dodatkowo w części praktycznej – m.in. posługiwać się typowymi programami użytkowymi, dobierać właściwy program (użytkowy lub napisany własnoręcznie) do rozwiązania zadania oraz stosować metody wyszukiwania i przetwarzania informacji w relacyjnych bazach danych. Ponadto na poziomie rozszerzonym zdający stosuje narzędzia i techniki informatyczne do modelowania i symulacji procesów oraz zjawisk. Absolwenci dużo lepiej poradzili sobie z zadaniami sprawdzającymi umiejętności z tego obszaru w części teoretycznej niż w części praktycznej. Zadania z tego obszaru były umiarkowanie trudne zarówno dla zdających z poziomu podstawowego (poziom wykonania: 51,4%), jak i rozszerzonego (poziom wykonania: 55,4%).

Najtrudniejsze (trudne) okazały się dla zdających z poziomu podstawowego, jak i z poziomu rozszerzonego (poziom wykonania: 43,5%), zadania sprawdzające umiejętności z trzeciego obszaru dotyczącym *Tworzenia informacji*. W obszarze tym zdający powinien stosować metody informatyczne do rozwiązywania problemów, a w szczególności określać sytuacje problemową, definiować problem, tworzyć specyfikację do postawionego zadania oraz proponować i analizować przedstawione rozwiązanie. Sformułowane rozwiązanie powinien realizować przy pomocy różnych narzędzi informatycznych, np. w wybranym języku programowania poprzez dobór odpowiedniego algorytmu. Ponadto zdający projektuje relacyjne bazy danych. Dodatkowo na poziomie rozszerzonym powinien uzasadniać poprawność, efektywność i złożoność rozwiązania problemu.

2. Problem „pod lupą”

Poniżej przedstawione zostały zadania wraz z ich omówieniem. Przykładowe rozwiązania zadań zostały wybrane tak, aby zilustrować zarówno rozwiązania poprawne, jak i takie, które zawierały podstawowe błędy popełniane przez absolwentów.

Poziom podstawowy

Zadanie 1. Doskonała inaczej (6 pkt)

Poniższy algorytm wyznacza wszystkie dzielniki dla zadanej liczby naturalnej n ≥ 1.

Specyfikacja algorytmu:

Dane: *liczba naturalna n ≥ 1*

Wynik: *lista liczb, które są dzielnikami liczby n.*

Algorytm:

1. *d ←1*
2. *dopóki d < n wykonuj*
   - *2.1. jeżeli n mod d = 0, to wypisz d*
   - *2.2. d ← d+1*

UWAGA: „n mod d" – oznacza resztę z dzielenia całkowitego liczby n przez d.

Celem zadania było sprawdzenie, czy zdający zna pojęcie algorytmu obliczania liczby dzielników, czy potrafi go zapisać, wyliczyć liczbę wykonanych operacji oraz czy potrafi zapisać algorytm obliczania liczb doskonałych II rzędu na podstawie definicji takiej liczby (czyli czy stosuje i modyfikuje znane mu algorytmy).

Polecenie a) sprawdzało umiejętność stosowania podanego w treści algorytmu wyznaczania i wypisywania wszystkich dzielników liczby naturalnej dla wskazanych argumentów. Zadanie było rzadko opuszczane, ale wielu zdających nie poradziło sobie z rozwiązaniem tego zadania (poziom wykonania: 53%).

a) Uzupełnij poniższą tabelę – podaj wyniki działania algorytmu dla wskazanych argumentów:

n	**Wynik algorytmu**
6	1 2 3
35	1 5 7
56	1 2 4 7 8 28
81	1 3 9 27

Przykład 1: Poprawne rozwiązanie zadania 1a

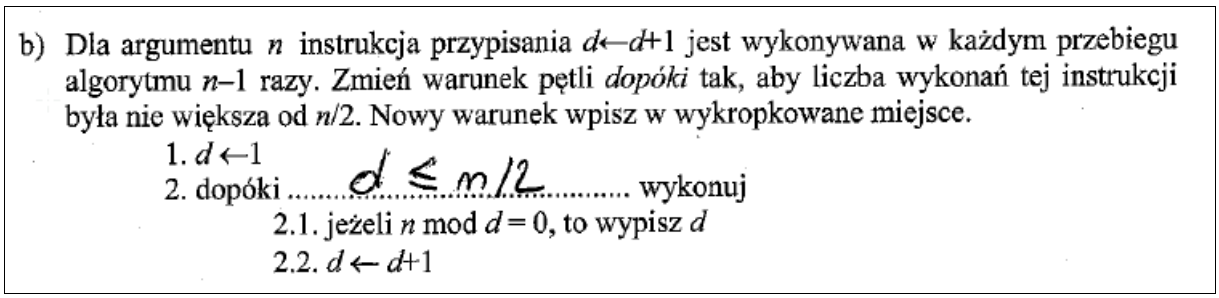
Prawdopodobną przyczyną niepowodzenia była pobieżna analiza działania algorytmu i pospieszne wypisywanie dzielników z pamięci, jednak w kontekście wyników polecenia a w zadaniu drugim (które sprawdzało tę samą umiejętność) przyczyn niepowodzeń można doszukać się tutaj w nierozumieniu lub błędnym posługiwaniem się funkcją *„mod"*.

a) Uzupełnij poniższą tabelę – podaj wyniki działania algorytmu dla wskazanych argumentów:

n	**Wynik algorytmu**
6	1 2 3
35	1 5 7
56	1 7 8
81	1 9

Przykład 2: BŁĘDNE rozwiązanie zadania 1a

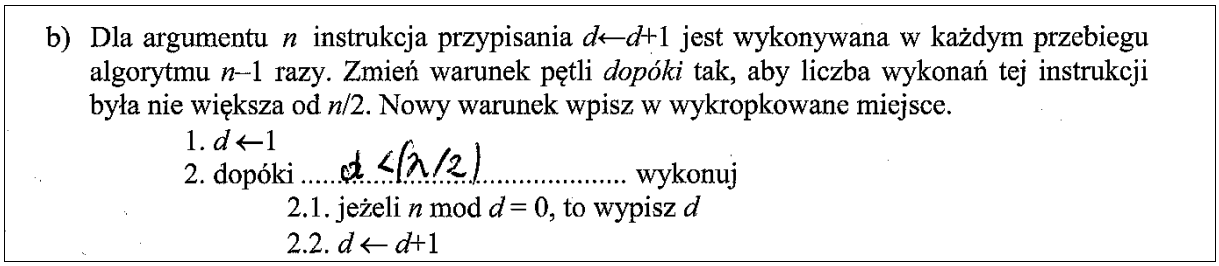
W poleceniu b) należało zoptymalizować działanie algorytmu z treści zadania tak, aby liczba wykonań instrukcji była nie większa od $\frac{n}{2}$. I tu zdający radzili sobie lepiej niż w pierwszym poleceniu i lepiej rozwiązywali to zadanie (poziom wykonania: 49%).


b) Dla argumentu *n* instrukcja przypisania *d*←*d*+1 jest wykonywana w każdym przebiegu algorytmu *n*–1 razy. Zmień warunek pętli *dopóki* tak, aby liczba wykonań tej instrukcji była nie większa od *n*/2. Nowy warunek wpisz w wykropkowane miejsce.

1. *d* ←1
2. dopóki ..... $d \leq n/2$ ..... wykonuj
   2.1. jeżeli *n* mod *d* = 0, to wypisz *d*
   2.2. *d* ← *d*+1

Przykład 3: Poprawnie rozwiązane zadanie 1b

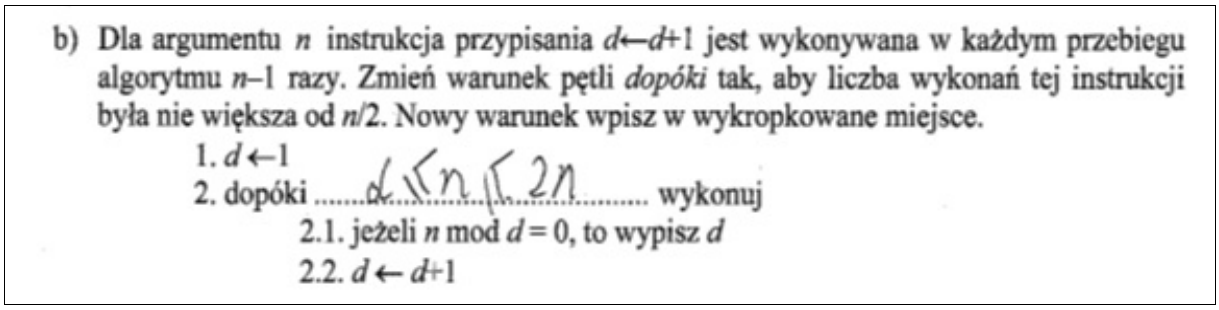
Zdarzały się błędne odpowiedzi, w których pojawiała się ostra nierówność.


b) Dla argumentu *n* instrukcja przypisania *d*←*d*+1 jest wykonywana w każdym przebiegu algorytmu *n*–1 razy. Zmień warunek pętli *dopóki* tak, aby liczba wykonań tej instrukcji była nie większa od *n*/2. Nowy warunek wpisz w wykropkowane miejsce.

1. *d* ←1
2. dopóki ..... $d < (n/2)$ ..... wykonuj
   2.1. jeżeli *n* mod *d* = 0, to wypisz *d*
   2.2. *d* ← *d*+1

Przykład 4: Rozwiązanie zdającego z zastosowaniem ostrej nierówności.

Sporadycznie pojawiały się odpowiedzi całkowicie niepoprawne, wynikające z niezrozumienia zarówno treści zadania, jak i poszczególnych elementów algorytmu.


b) Dla argumentu *n* instrukcja przypisania *d*←*d*+1 jest wykonywana w każdym przebiegu algorytmu *n*–1 razy. Zmień warunek pętli *dopóki* tak, aby liczba wykonań tej instrukcji była nie większa od *n*/2. Nowy warunek wpisz w wykropkowane miejsce.

1. *d* ←1
2. dopóki ..... $d < n < 2n$ ..... wykonuj
   2.1. jeżeli *n* mod *d* = 0, to wypisz *d*
   2.2. *d* ← *d*+1

Przykład 5: BŁĘDNE rozwiązanie zadania 1b

W zadaniu 1c podano definicję liczby doskonałej II rzędu:

c) Liczbą doskonałą II rzędu nazywamy liczbę naturalną n, która jest równa iloczynowi wszystkich swoich dzielników mniejszych od niej samej. Liczba 6 jest taką liczbą, ponieważ 6 = 1•2•3. Podaj algorytm sprawdzający, czy liczba naturalna n > 1 jest liczbą doskonałą II rzędu.

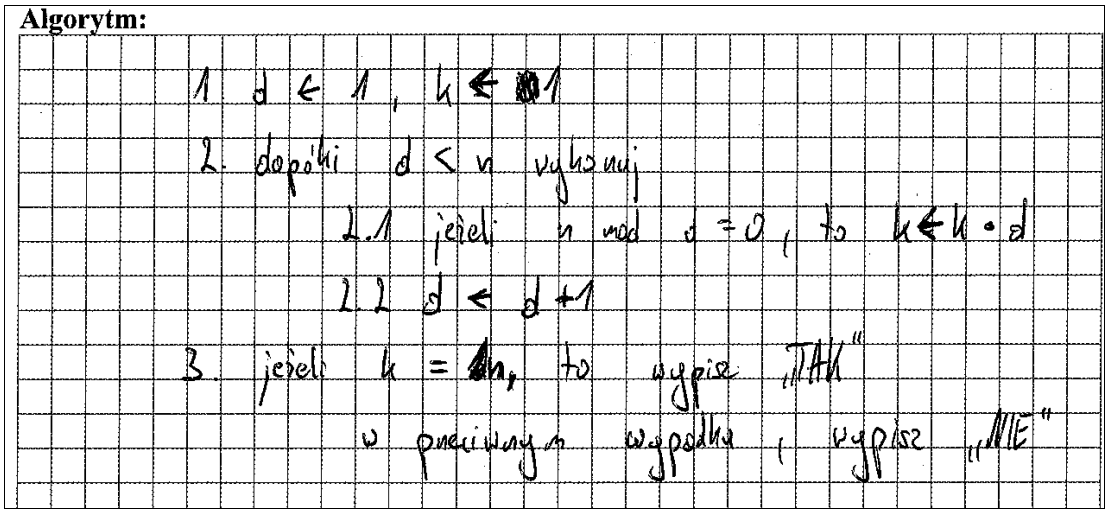
Specyfikacja:

Dane: Liczba naturalna n > 1.

Wynik: Odpowiedź „TAK”, gdy liczba n jest liczbą doskonałą II rzędu albo odpowiedź „NIE”, gdy liczba n nie jest liczbą doskonałą II rzędu.

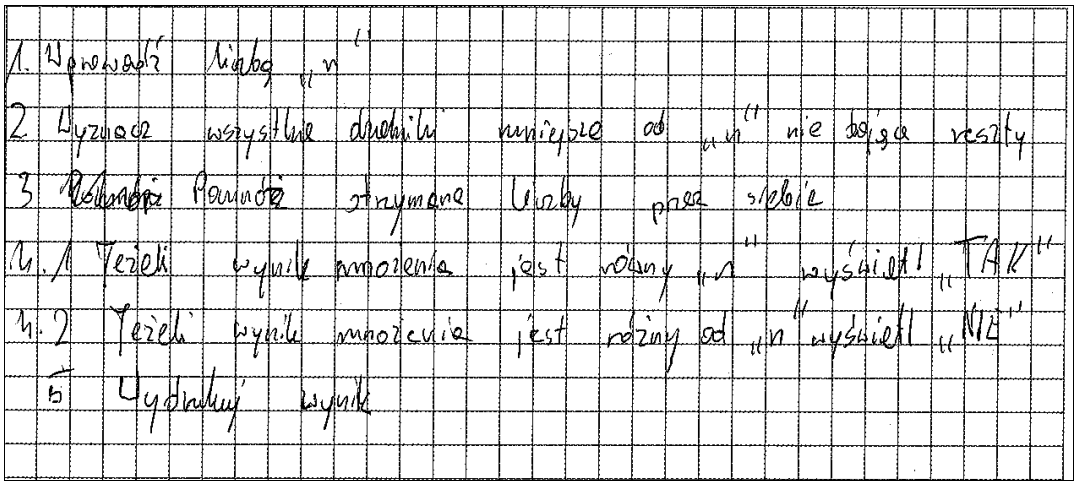
Był to punkt najczęściej opuszczany przez zdających (poziom wykonania: 43%). W dwóch pierwszych podpunktach analizowany był algorytm z treści zadania, który wypisywał dzielniki liczby naturalnej (mniejsze od niej). Zdający mógł zauważyć ścisły związek między podanym algorytmem,

a poleceniem w punkcie c. Wystarczyło zmodyfikować zaprezentowany algorytm, tzn. zamienić wypisywanie dzielników na obliczanie ich iloczynu oraz nadać wartość początkową iloczynowi.

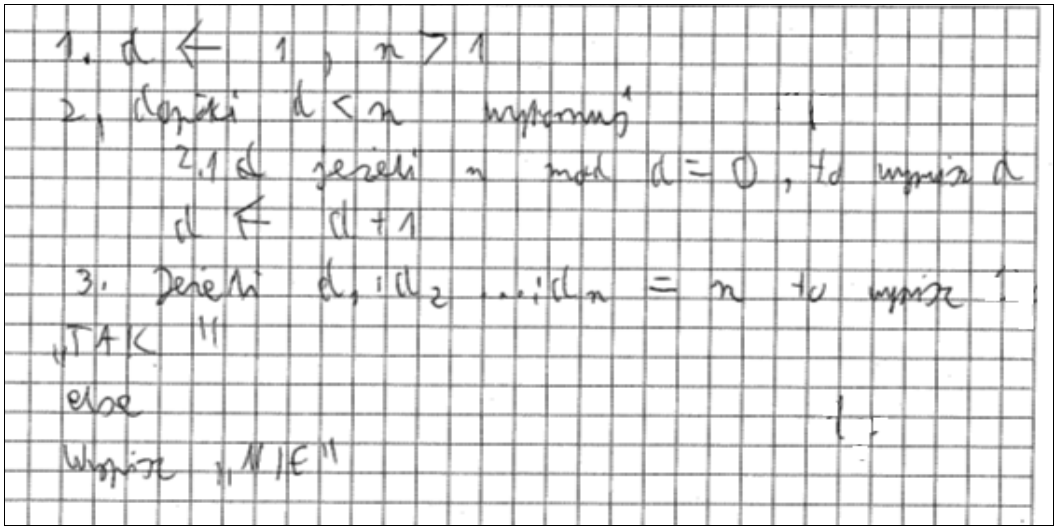


Przykład 6: Poprawne rozwiązanie zadania 1c

Dla tych, którzy podjęli próbę rozwiązania tego punktu, główną trudnością było prawidłowe obliczanie iloczynu dzielników oraz właściwa organizacja pętli.

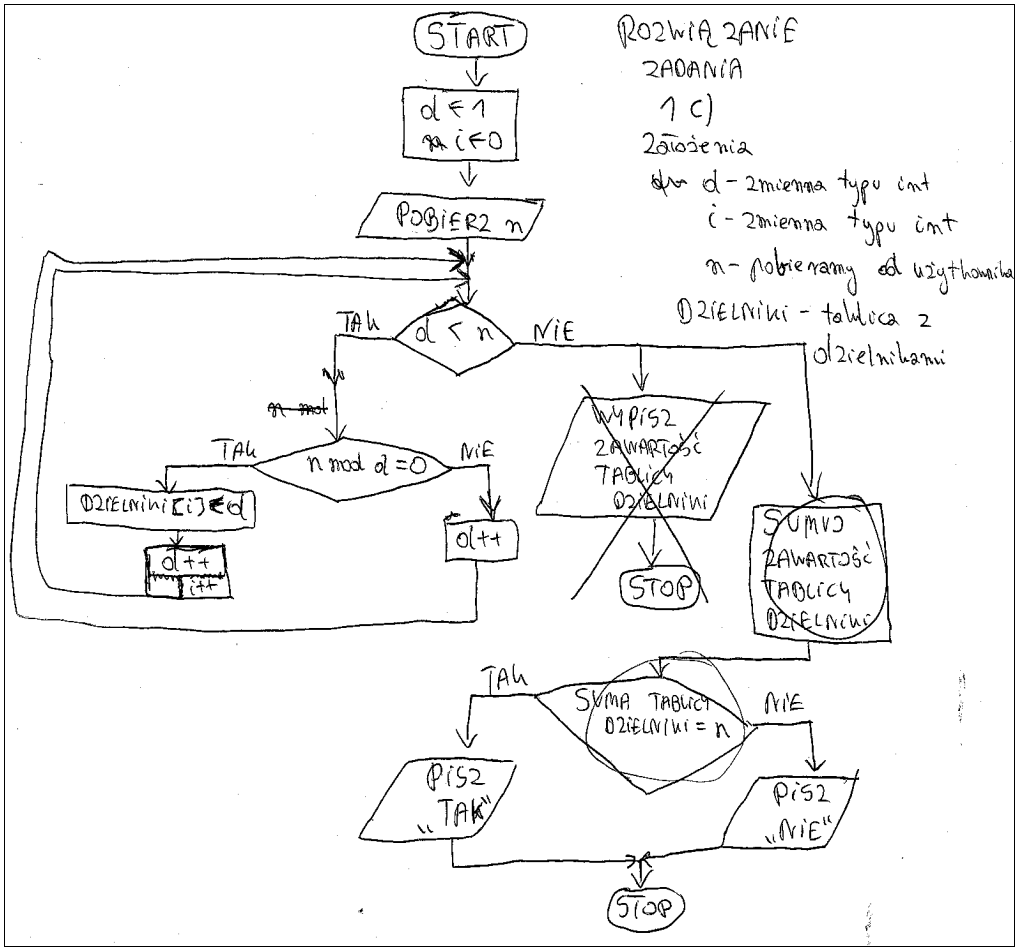


Przykład 7: BŁĘDNE obliczanie iloczynu dzielników



Przykład 8: BŁĘDNE obliczanie iloczynu dzielników

Jednym ze sposobów rozwiązywania tego zadania było umieszczanie kolejnych znalezionych dzielników w tablicy. Nie było to koniecznością, ale taki właśnie sposób rozwiązania proponowała część zdających. Część z nich nie potrafiła tego poprawnie zapisać w algorytmie ani nie była w stanie obliczyć iloczynu liczb zapisanych w tablicy.



Przykład 9: BŁĘDNE rozwiązanie zadania 1c z wykorzystaniem tablicy dzielników

Zadanie 2. Min-Max (6 pkt)

Dana jest parzysta, dodatnia liczba całkowita n oraz n-elementowa tablica a[1..n] liczb całkowitych. Rozważ poniższy algorytm działający na tej tablicy.

Algorytm:

1. *i ← 1*
2. *dopóki i<n wykonuj*
   - *2.1. jeżeli a[i] > a[i+1] to zamień zawartości a[i] oraz a[i+1]*
   - *2.2. i ← i+2*

Zadanie sprawdzało znajomość algorytmu jednoczesnego znajdowania największej i najmniejszej liczby w *n*-elementowym zbiorze.

Polecenia a) zadania sprowadzało się do przeanalizowania działania podanego algorytmu dla zadanych zestawów danych i podania wyniku jego działania. Uzyskany przez zdających wynik (poziom wykonania: 62%) świadczy o tym, że maturzyści dobrze radzą sobie z analizą prostego algorytmu, niewymagającego od nich operacji matematycznych.

a) Przeanalizuj podany algorytm i podaj wynik jego działania dla poniższych danych – wpisz odpowiednie liczby w wykropkowane miejsca.

dla $n = 6$, $a = [\ 45, 12, 7, 39, 20, 1\ ]$:

po wykonaniu algorytmu $a = [$ ..12.., ..45.., ..7.., ..39.., ..1.., ..20.. $]$

dla $n = 8$, $a = [\ 21, 1, 56, 90, 8, 8, 19, 47\ ]$:

po wykonaniu algorytmu $a = [$ ..1.., ..21.., ..56.., ..90.., ..8.., ..8.., ..19.., ..47.. $]$

Przykład 10: Poprawne rozwiązanie zadania 2a

dla $n = 6$, $a = [\ 45, 12, 7, 39, 20, 1\ ]$:

po wykonaniu algorytmu $a = [$ ..3.., ..9.., ..14.., ..22.., ..41.., ..47.. $]$

dla $n = 8$, $a = [\ 21, 1, 56, 90, 8, 8, 19, 47\ ]$:

po wykonaniu algorytmu $a = [$ ..3.., ..10.., ..20.., ..21.., ..23.., ..49.., ..58.., ..92.. $]$

Przykład 11: BŁĘDNE rozwiązanie zadania 2a

Polecenie b) jest konsekwencją polecenia a). Rozwiązując pierwszy podpunkt zadania, zdający właściwie zauważali, że wynikiem działania algorytmu jest tablica, której sąsiednie pary są uporządkowane niemalejąco (poziom wykonania: 57%).

Dla każdego $i = 1, 3, \ldots, n\text{-}1$ mamy $a[i]$ [≤] $a[i+1]$.

Przykład 12: Poprawne rozwiązanie zadania 2b

W poleceniu c) należało uzupełnić luki w algorytmie tak, aby znajdował on minimalną i maksymalną wartość w tablicy:

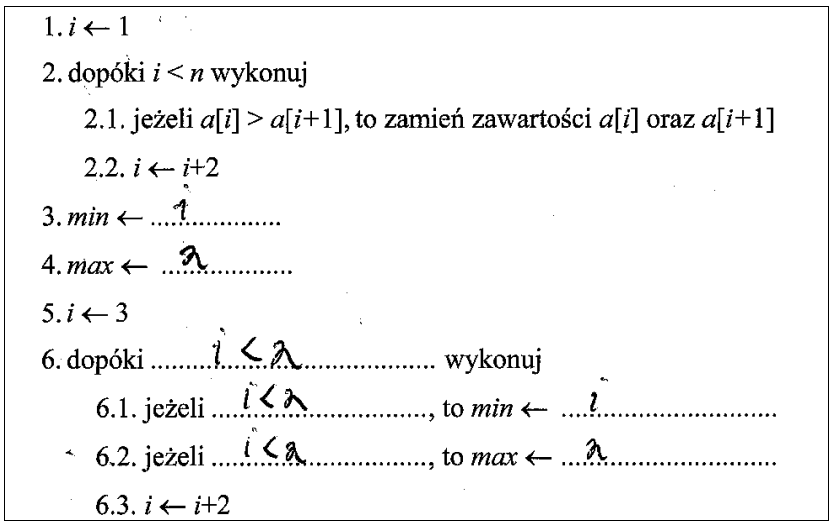
c) W poniższym algorytmie uzupełnij luki tak, aby znajdował on minimalną i maksymalną wartość w tablicy $a[1..n]$ liczb całkowitych, gdzie n to parzysta liczba całkowita dodatnia. Wykorzystaj fakt, że z pary porównywanych ze sobą elementów ciągu tylko jeden warto brać pod uwagę jako kandydata na minimum i tylko jeden jako kandydata na maksimum.

Algorytm:

1. $i \leftarrow 1$
2. dopóki $i < n$ wykonuj
   - 2.1. jeżeli $a[i] > a[i+1]$, to zamień zawartości $a[i]$ oraz $a[i+1]$
   - 2.2. $i \leftarrow i+2$
3. $min \leftarrow$ ..a[i]..
4. $max \leftarrow$ ..a[i+1]..
5. $i \leftarrow 3$
6. dopóki ..i < n.. wykonuj
   - 6.1. jeżeli ..a[i] > a[i+1].., to $min \leftarrow$ ..a[i]..
   - 6.2. jeżeli ..a[i] < a[i+1].., to $max \leftarrow$ ..a[i+1]..
   - 6.3. $i \leftarrow i+2$

Przykład 13: Poprawne rozwiązanie zadania 2c

Mimo podpowiedzi w treści zadania: „*Wykorzystaj fakt, że z pary porównywanych ze sobą elementów ciągu tylko jeden warto brać pod uwagę jako kandydata na minimum i tylko jeden jako kandydata na maksimum.*”, zdający w większości przypadków nie potrafili określić, który element jest właściwym kandydatem na minimum, a który na maksimum (poziom wykonania: 31%).


1. $i \leftarrow 1$
2. dopóki $i < n$ wykonuj
   2.1. jeżeli $a[i] > a[i+1]$, to zamień zawartości $a[i]$ oraz $a[i+1]$
   2.2. $i \leftarrow i+2$
3. $min \leftarrow$ ...i...
4. $max \leftarrow$ ...a...
5. $i \leftarrow 3$
6. dopóki ...i < a... wykonuj
   6.1. jeżeli ...i < a..., to $min \leftarrow$ ...i...
   6.2. jeżeli ...i < a..., to $max \leftarrow$ ...a...
   6.3. $i \leftarrow i+2$

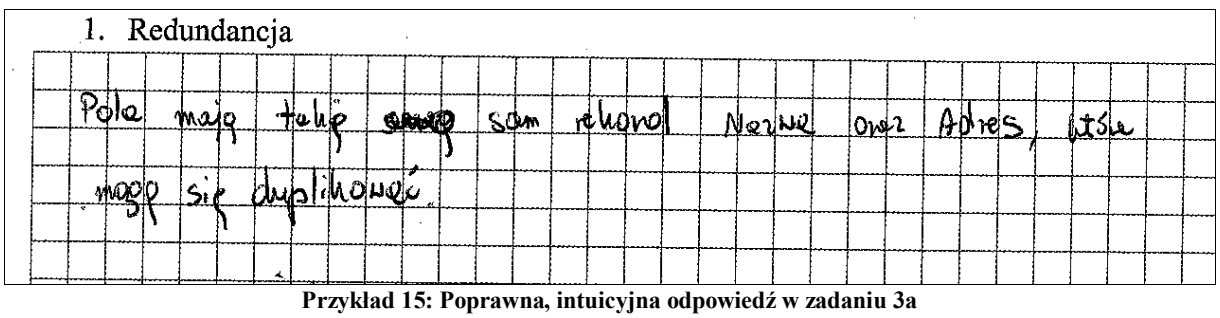
Przykład 14: BŁĘDNE rozwiązanie zadania 2c

Zadanie 3 składało się z jednego polecenia otwartego (podstawowe formy organizacji informacji w bazach danych) oraz sześciu zamkniętych (podstawowa terminologia związana z sieciami komputerowymi, zasady pracy sprzętu komputerowego, znajomość typowych narzędzi informatycznych oraz sposoby reprezentowania informacji w komputerze).

Najwięcej kłopotów w tym zadaniu sprawiło pytanie otwarte (poziom wykonania: 13%), w którym należało wyjaśnić pojęcie redundancji oraz wskazać anomalie przy modyfikacji wynikające z redundancji.

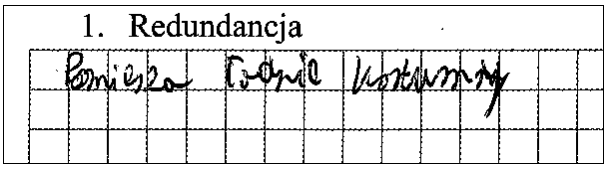
Jako podłoże tak dużych kłopotów zdających z rozwiązaniem tego zadania można wskazać niedostateczne rozumienie pojęć i terminów związanych z projektowaniem baz danych i stosowaniem podstawowych pojęć z zakresu relacyjnych baz danych. A przecież duża część pracy nad bazą wymaga namysłu i zastanowienia się. Na lekcjach prawdopodobnie nie omawia się fazy projektowania baz danych lub się je marginalizuje.

Informatyka, jak każdy inny przedmiot szkolny, ma swój specyficzny język. W trakcie nauki szkolnej na lekcjach informatyki rzadko wykorzystuje się podręczniki, zarówno przez uczniów, jak i nauczycieli. Stąd wynikają problemy zdających w korzystaniu z języka specyficznego dla przedmiotu. Nawet jeśli rozumieją pojęcie intuicyjnie, to nie potrafią w zwięzły sposób go wyjaśnić czy zdefiniować.

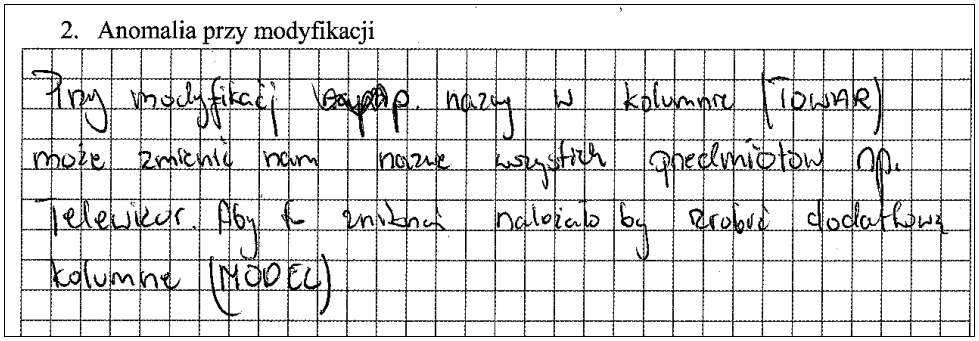

1. Redundancja

Pola mają takie same wartości Nazwę oraz Adres, które mogą się duplikować.

Przykład 15: Poprawna, intuicyjna odpowiedź w zadaniu 3a



Przykład 16: BŁĘDNE odpowiedź w zadaniu 3a



Przykład 17: BŁĘDNE odpowiedź w zadaniu 3a

Pozostałe punkty w zadaniu 3. były zadaniami zamkniętymi i nie sprawiały zdającym większych trudności (poziom wykonania: od 51% do 89%).

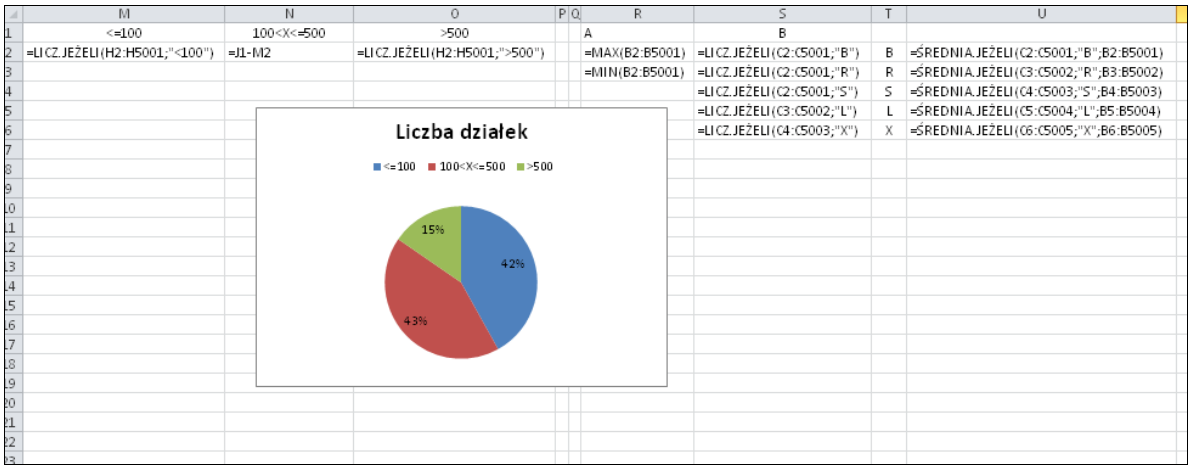
Zadanie 4, którego treść nawiązuje do sytuacji praktycznej, weryfikowało, czy zdający przystępuje do rozwiązania postawionego w treści zadania problemu w sposób planowy, czy poprawnie dobiera właściwy program (użytkowy lub własnoręcznie napisany) do rozwiązywanego zadania. Najprostszym narzędziem, jakim można było posłużyć się przy rozwiązywaniu tego zadania, był arkusz kalkulacyjny. Wykonując obliczenia przy pomocy wbudowanych funkcji lub zaprojektowanych formuł, zdający wykorzystywał zdobytą wiedzę i umiejętności do rozwiązania zadania, obrazując graficznie informacje adekwatnie do ich charakteru.

Polecenia a) i b) nie sprawiały większych kłopotów (poziom wykonania: 79% i 55%). Główną trudnością przy rozwiązaniu zadania było utworzenie, w poleceniu c, zestawienia obrazującego liczbę działek w określonych przedziałach podatkowych i przygotowanie wykresu ilustrującego to zestawienie (poziom wykonania: 22%).

Zadanie to można było rozwiązać na kilka sposobów, np. najprostszym rozwiązaniem (dla osób nieznających zaawansowanych, wbudowanych funkcji) było posortowanie działek według wielkości podatku i zliczenie „ręcznie" liczby działek w określonych przedziałach. Można było również skorzystać z funkcji w arkuszu kalkulacyjnym: licz.jeżeli() lub licz.warunki().

	A	B	C	D	E	F	H	I	J	K	L
1											
2	537/93	502,87	L	C		=B2*0,04*20%					=B2*P3*0,10%
3	536/57	508,22	L	C		=B3*0,04*20%	ilość działek	podatek			
4	555/70	509,29	L	C		=B4*0,04*20%	1419	>=100			
5	555/45	513,39	L	C		=B5*0,04*20%	=5000-H4+H6	100 >=50			
6	557/27	517,35	L	C		=B6*0,04*20%	769	<500			
7	567/49	526,83	L	C		=B7*0,04*20%					
8	524/96	529,8	L	C		=B8*0,04*20%					
9	560/22	531,31	L	C		=B9*0,04*20%					
10	522/65	532,44	L	C		=B10*0,04*20%					
11	561/24	536,19	L	C		=B11*0,04*20%					
12	563/68	545,58	L	C		=B12*0,04*20%					
13	553/83	550,37	L	C		=B13*0,04*20%					a)
14	525/76	557,89	L	C		=B14*0,04*20%					N
15	521/19	565,75	L	C		=B15*0,04*20%					N
16	522/21	568,05	L	C		=B16*0,04*20%					
17	539/66	570,88	L	C		=B17*0,04*20%					b)
18	562/55	572,4	L	C		=B18*0,04*20%					
19	567/32	577,54	L	C		=B19*0,04*20%					
20	542/79	587,8	L	C		=B20*0,04*20%					
21	522/93	589,53	L	C		=B21*0,04*20%					
22	526/76	591,65	L	C		=B22*0,04*20%					
23	521/26	595,06	L	C		=B23*0,04*20%					
24	532/21	605,21	L	C		=B24*0,04*20%		b)			
25	557/31	613,79	L	C		=B25*0,04*20%			Rodzaj Działki	Liczba Działek	Srednia powierzchnia
26	569/69	650,35	L	C		=B26*0,04*20%			Rolna	708	989,582016925248
27	532/54	651,2	L	C		=B27*0,04*20%			Budowlana	1775	=ŚREDNIA(B2:B1776;K27)
28	544/92	657,39	L	C		=B28*0,04*20%			Leśna	559	=ŚREDNIA(B1777:B2335;K28)
29	521/93	657,98	L	C		=B29*0,04*20%			Siedliskowa	996	=ŚREDNIA(B2336:B3879;K29)
30	535/25	660,69	L	C		=B30*0,04*20%			Rekreacyjna	994	989,942482412062

Przykład 18: Poprawne rozwiązanie wykorzystujące „ręczne" obliczanie liczby działek



	M	N	O	R	S	T	U
1	<=100	100<X<=500	>500	A	B		
2	=LICZ.JEŻELI(H2:H5001;"<100")	=J1-M2	=LICZ.JEŻELI(H2:H5001;">500")	=MAX(B2:B5001)	=LICZ.JEŻELI(C2:C5001;"B")	B	=ŚREDNIA.JEŻELI(C2:C5001;"B";B2:B5001)
3				=MIN(B2:B5001)	=LICZ.JEŻELI(C2:C5001;"R")	R	=ŚREDNIA.JEŻELI(C3:C5002;"R";B3:B5002)
4					=LICZ.JEŻELI(C2:C5001;"S")	S	=ŚREDNIA.JEŻELI(C4:C5003;"S";B4:B5003)
5					=LICZ.JEŻELI(C3:C5002;"L")	L	=ŚREDNIA.JEŻELI(C5:C5004;"L";B5:B5004)
6					=LICZ.JEŻELI(C4:C5003;"X")	X	=ŚREDNIA.JEŻELI(C6:C5005;"X";B6:B5005)

Przykład 19: Poprawne rozwiązanie zadania z zastosowaniem wbudowanych funkcji

Jak co roku, najtrudniejsze okazało się zadanie programistyczne, którym było **zadanie 5**. Sprawdzało ono, czy zdający formułuje informatyczne rozwiązanie problemu przez dobór algorytmu oraz odpowiednich struktur danych i realizuje je w wybranym języku programowania.

Zadanie to miało najwyższą frakcję opuszczeń. Nieliczni zdający, którzy podjęli próbę rozwiązania tego zadania, rozwiązywali je poprawnie i radzili sobie z określeniem w podpunkcie a), które liczby są swoją wielokrotnością (poziom wykonania: 7%).

```
7  bool sprawdz_wielokrotnosc(int a, int b)
8  {
9      return (a%b==0 || b%a==0);
10 }
```

Przykład 20: Poprawna funkcja sprawdzająca wielokrotności liczb

W części rozwiązań zdający pomijali fakt, że druga liczba może być wielokrotnością pierwszej, ale też pierwsza liczba może być wielokrotnością drugiej.

```
17     while(inFile.good()){
18                          if (i % 2 == 1)
19                             inFile>>liczba1;
20                              else inFile>>liczba2;
21
22                           if((liczba1 % liczba2)==0)  licznik++;
23
24                          i++;
25                          }
```

Przykład 21: Rozwiązanie nieuwzględniające wzajemnej wielokrotności liczb

Bardzo trudne okazało się polecenie b), w którym należało sprawdzić, które pary liczb są względnie pierwsze. Mimo podpowiedzi w zadaniu (chodziło o pary liczb, których NWD = 1) górowały błędne rozwiązania (poziom wykonania: 4%).

```
11 int nwd(int a, int b)
12 {
13     while(b>0)
14     {
15         int temp = b;
16         b = a%b;
17         a = temp;
18     }
19     return a;
20 }
```

Przykład 22: Poprawna funkcja obliczająca NWD dwóch liczb

```
45 wyjscie<<"podpunkt B)\n";
46 int licznik = 0;
47 for(int i=0;i<1000;i++)
48 {
49     if(nwd(tablica[i][0], tablica[i][1]) == 1)
50         licznik++;
51 }
52 wyjscie<<"wynik: "<<licznik<<"\n";
```

Przykład 23: Fragment poprawnego rozwiązania zadania 5b wykorzystującego funkcję NWD

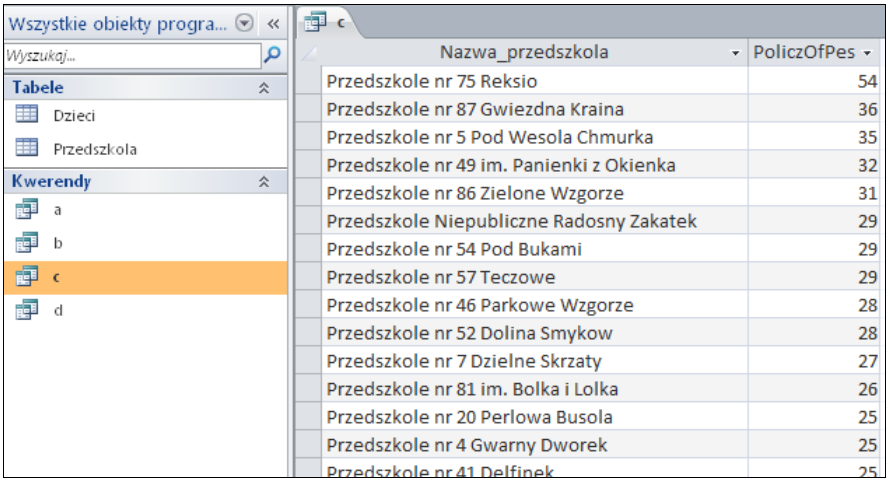
Najtrudniejsze dla zdających okazało się polecenie c), w którym maturzyści nie radzili sobie z obliczaniem sumy cyfr - (poziom wykonania: 1%).

```
21 int suma_cyfr(int a)
22 {
23     int suma = 0;
24     while(a>0)
25     {
26         suma += a%10;
27         a/=10;
28     }
29     return suma;
30 }
```

Przykład 24: Poprawna funkcja obliczająca sumę cyfr.

Zadanie 6 było zadaniem bazodanowym i sprawdzało stosowanie metod wyszukiwania i przetwarzania informacji w relacyjnych bazach danych.

Większość absolwentów podjęła próbę rozwiązania zadania. Jednak zadanie sprawiało wiele trudności. Do wykonania poleceń a) i b) wystarczyły elementarne umiejętności z zakresu baz danych (poziom wykonania: 51% i 48%), a błędy, jakie zdający popełniali w tych punktach, wynikały z niedoczytania treści lub nieprawidłowego określenia kryterium wyszukiwania. Znamienne jest jednak to, że dość często zdający nie potrafili zawęzić wyników wyszukiwania do 1 rekordu, tylko wyświetlali całą listę (brak umiejętności stosowania selekcji poziomej). Kłopot sprawiało także skonstruowanie wyrażenia obliczającego liczbę dzieci przyjętych ponad limit (poziom wykonania: 41%). Zdający, którzy podjęli się rozwiązania punktu d), radzili sobie wyświetlając liczbę miejsc w przedszkolu oraz liczbę przyjętych dzieci. Różnice wyliczali „ręcznie”.



Przykład 25: Rozwiązanie poprawne. Przykład braku zastosowania selekcji poziomej w rozwiązaniu.

Poziom rozszerzony

Zadanie 1: Korale (8 pkt)

Rozważamy następującą rekurencyjną procedurę Korale, której parametrem jest dodatnia liczba całkowita n.

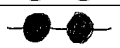
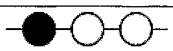
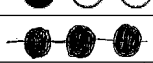
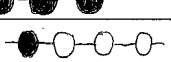
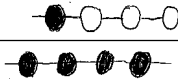
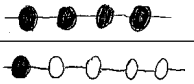
Korale(n)

1. *Jeżeli n = 1, to*
   - *1.1. nawlecz czarny koralik na prawy koniec sznurka,*
   - *1.2. zakończ działanie procedury.*
2. *Jeżeli n jest parzyste, to*
   - *2.1. wykonaj* ***Korale(n/2)****,*
   - *2.2. nawlecz biały koralik na prawy koniec sznurka,*
   - *2.3. zakończ działanie procedury.*
3. *Jeżeli n jest nieparzyste, to*
   - *3.1. wykonaj* ***Korale((n-1)/2)****,*
   - *3.2. nawlecz czarny koralik na prawy koniec sznurka,*
   - *3.3. zakończ działanie procedury.*

Zadanie sprawdzało znajomość pojęcia algorytmu rekurencyjnego i rozmienia w nim mechanizmów przekazywania parametrów.

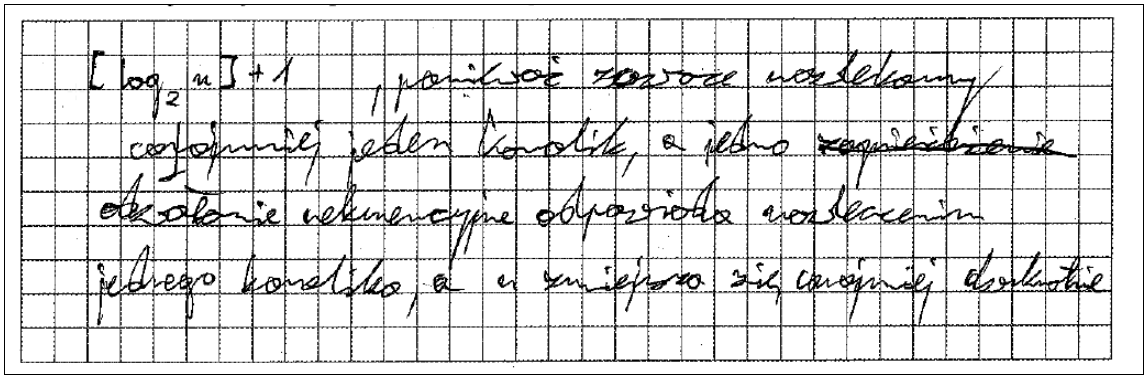
W poleceniu a zadaniem zdających było przeanalizowanie algorytmu rekurencyjnego zapisanego w pseudokodzie dla zapisanych w tabeli danych wejściowych. Większość bez problemu poradziła sobie z tym zadaniem (poziom wykonania: 72%).

a) Uzupełnij tabelę i w ten sposób przedstaw wynik działania powyższego algorytmu dla podanych argumentów *n*:

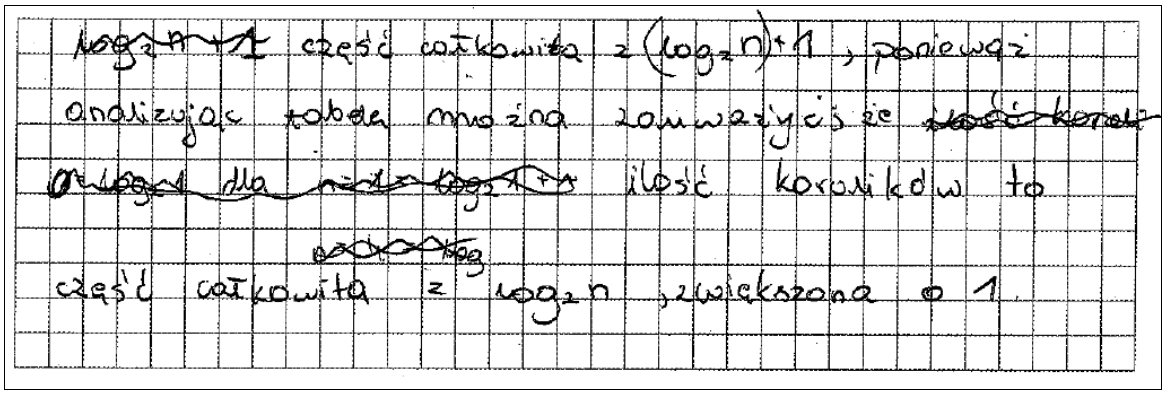
n	wynik działania **Korale(*n*)**
1	
2	
3	
4	
7	
8	
15	
16	

Przykład 26: Poprawne rozwiązanie zadania 1a

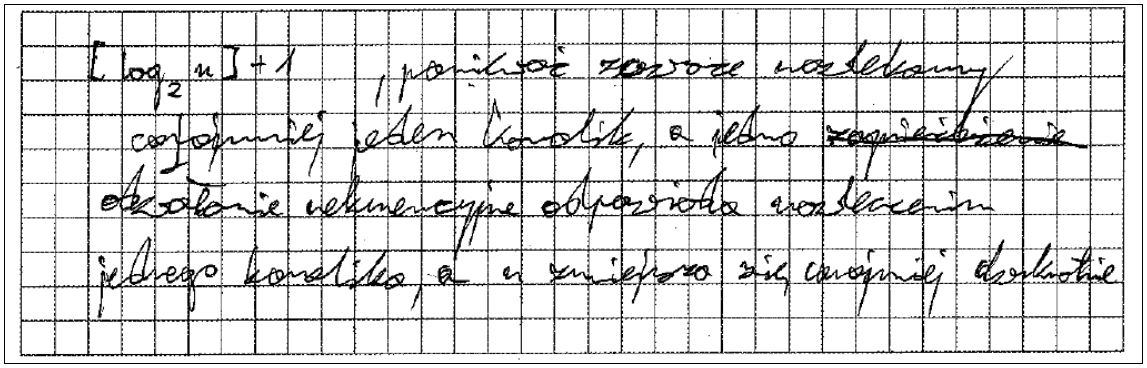
Polecenie b) sprawdzało umiejętność obliczania złożoności zaprezentowanego algorytmu i polegało ono policzeniu przez zdających liczby wywołań rekurencyjnych procedury dla danej liczby *n*. Maturzyści (nawet bez znajomości i używania pojęcia logarytmu) potrafili opisać zależność między liczbą *n* a liczbą wywołań procedury, słabiej poradzili sobie z uzasadnieniem (poziom wykonania: 28%).



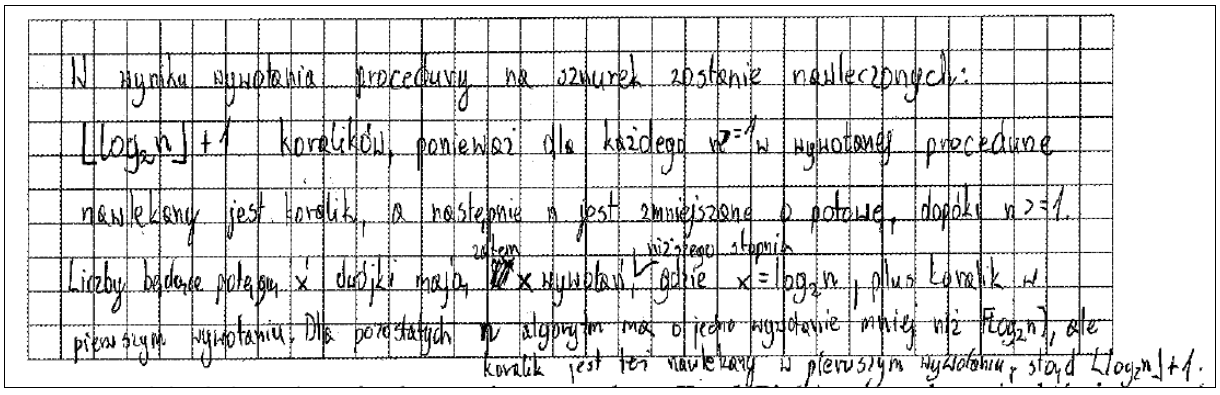
Przykład 27: Rozwiązanie zadania 1b



Przykład 28: Rozwiązanie zadania 1b

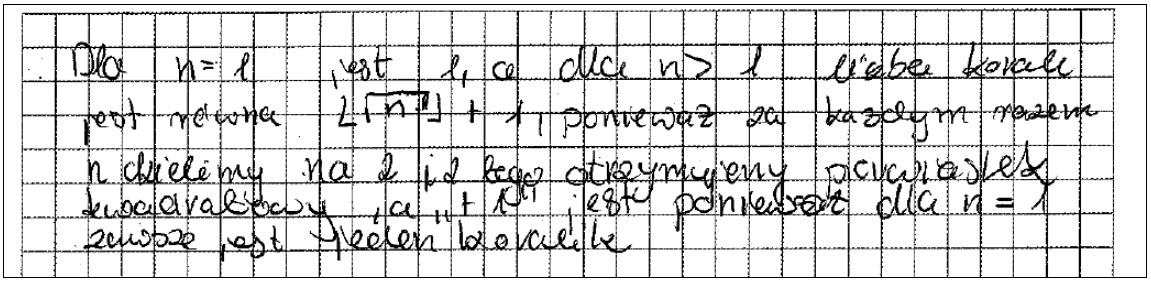


Przykład 29: Rozwiązanie zadania 1b



Przykład 30: Rozwiązanie zadania 1b

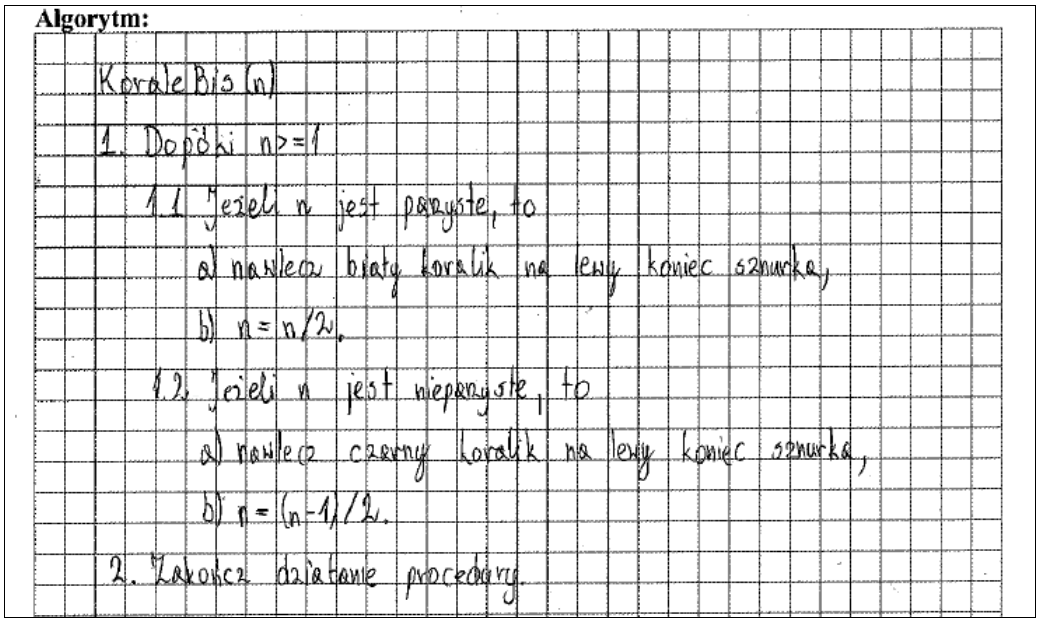
Pojawiały się także błędne odpowiedzi : $\sqrt{n}$ lub $\sqrt{\frac{1}{2}n}$ – odpowiednio zaokrąglone w zależności od parzystości liczby *n*.



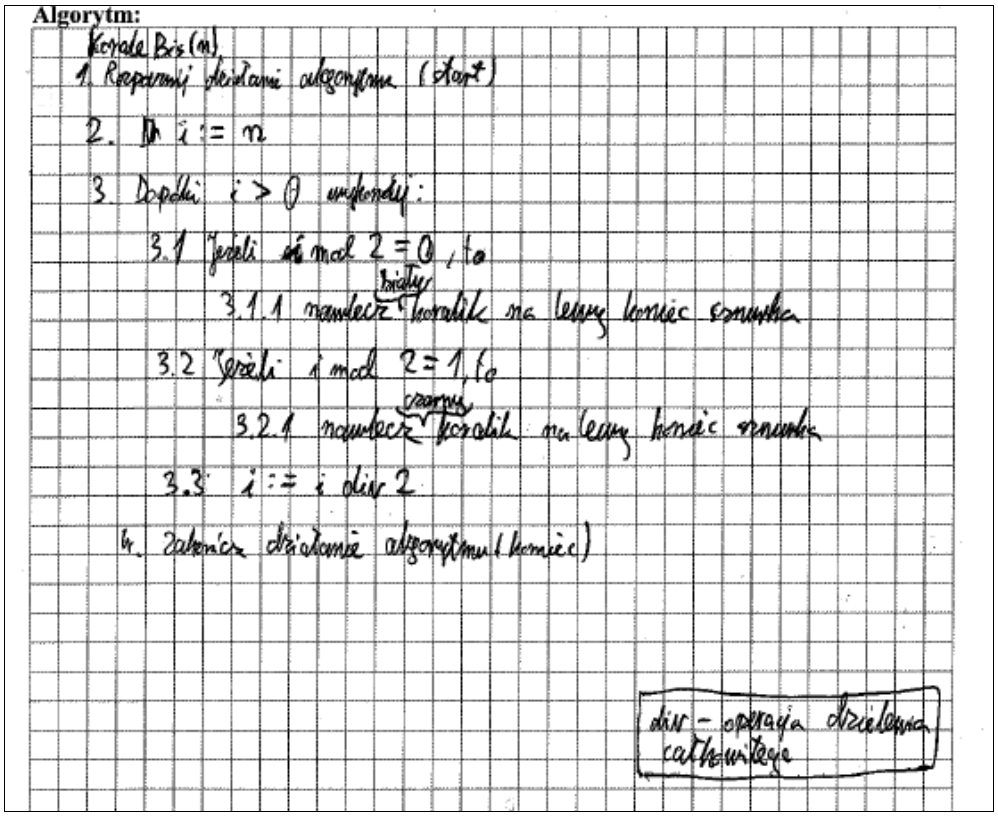
Przykład 31: BŁĘDNE rozwiązanie zadania 1b

Polecenie c) sprawdzało umiejętność zamiany algorytmu rekurencyjnego na iteracyjny (poziom wykonania: 42%). Te osoby, które rozwiązały poprzednie dwa polecenia, nie miały problemu z rozwiązaniem tego punktu. Wśród rozwiązań zdających pojawiały się również ciekawe, trochę zaskakujące rozwiązania (np. pamiętanie kolejnych ciągów koralików jako napisów w tablicy).

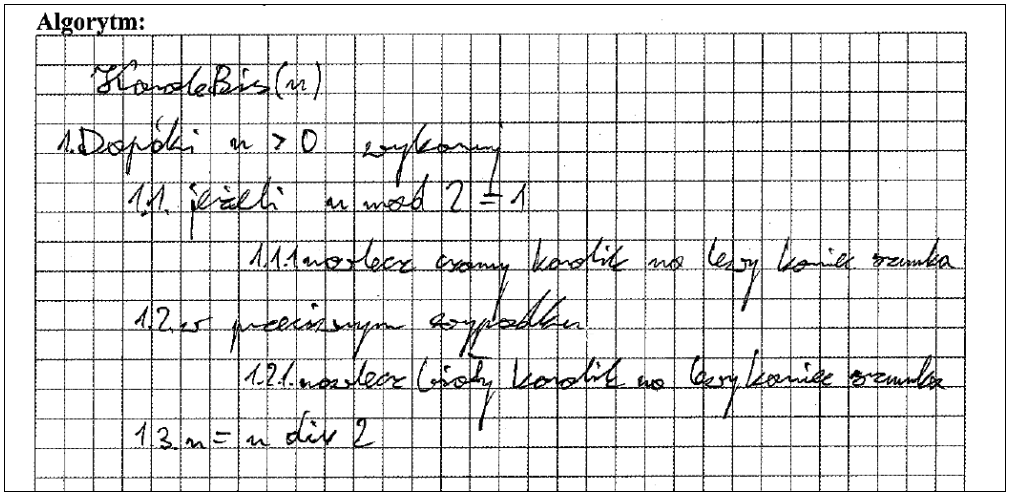
Poniżej przedstawiono kilka przykładowych, poprawnych rozwiązań. Jednym ze sposobów rozwiązania zadania było nawlekanie koralików na lewy koniec sznurka. Jednak przy tym podejściu do rozwiązania należało zwrócić uwagę na warunek zakończenia pętli.



Przykład 32: Poprawne rozwiązanie zadania 1c z wyrażeniem dopóki (n>=1)

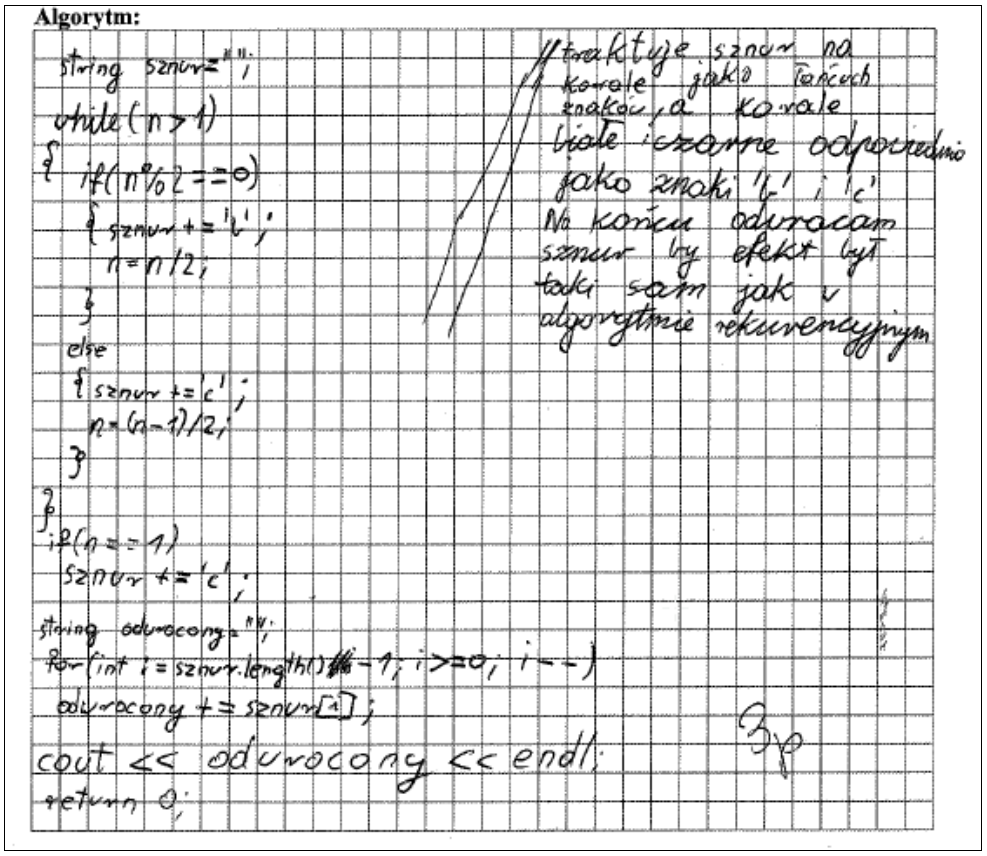


Przykład 33: Poprawne rozwiązanie zadania 1c z wyrażeniem dopóki (i>0)

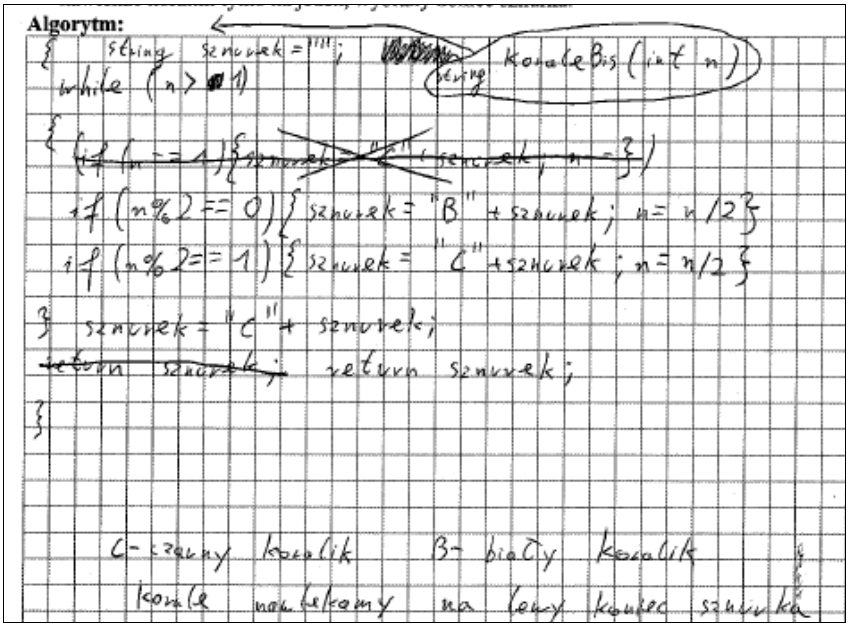


Przykład 34: Poprawne rozwiązanie zadania 1c z wyrażeniem dopóki (n>0)

Ciekawym rozwiązaniem było potraktowanie sznura korali jako napisu. W takim rozwiązaniu bardzo istotna była kolejność sklejania napisów. Część zdających zauważyła, że podczas doklejania kolejnych znaków do napisu z prawej strony końcowy napis należy odwrócić, aby uzyskać wymagany ciąg białych i czarnych korali.



Przykład 35: Przykład z doklejaniem znaków z prawej strony i obracaniem napisu wynikowego

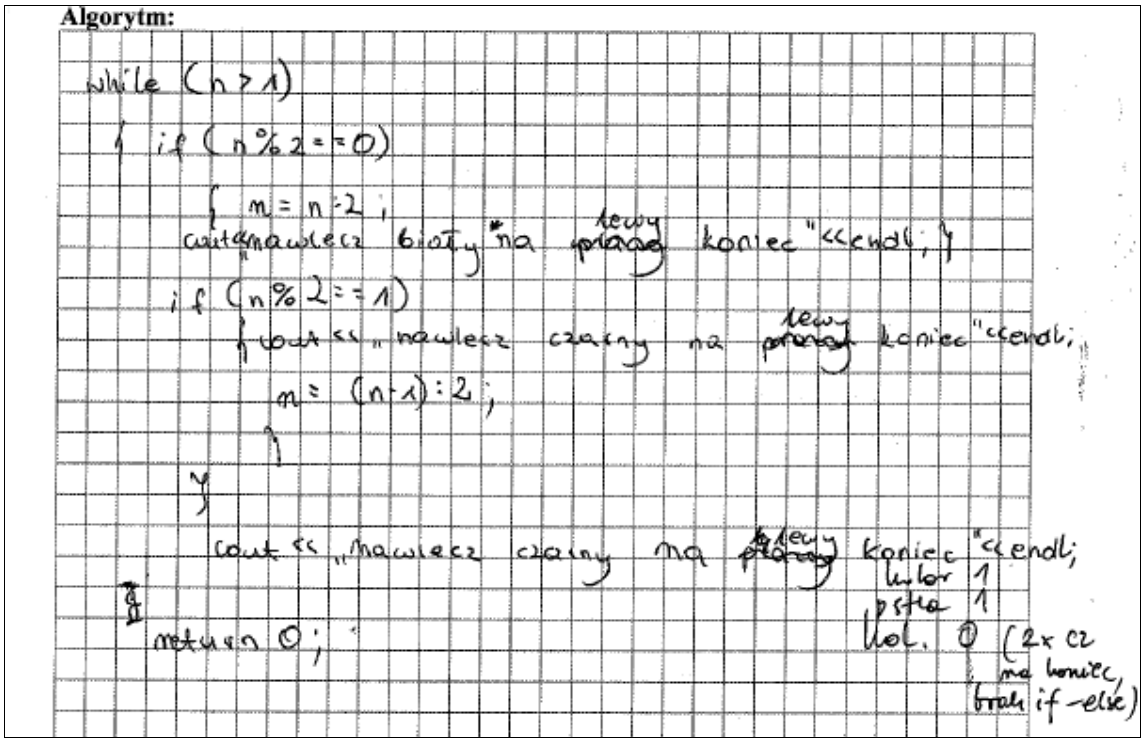


Przykład 36: Przykład doklejania znaku z lewej strony bez potrzeby końcowego odwracania napisu wynikowego.

Dodatkowo w obu powyższych przykładach zdający słusznie zauważyli, że zapis w warunku pętli: while (n>1) powoduje, że po jej zakończeniu należy do sznurka dodać czarny koralik. Jednak zdarzały się rozwiązania, w których część zdających pomijała ten istotny element.

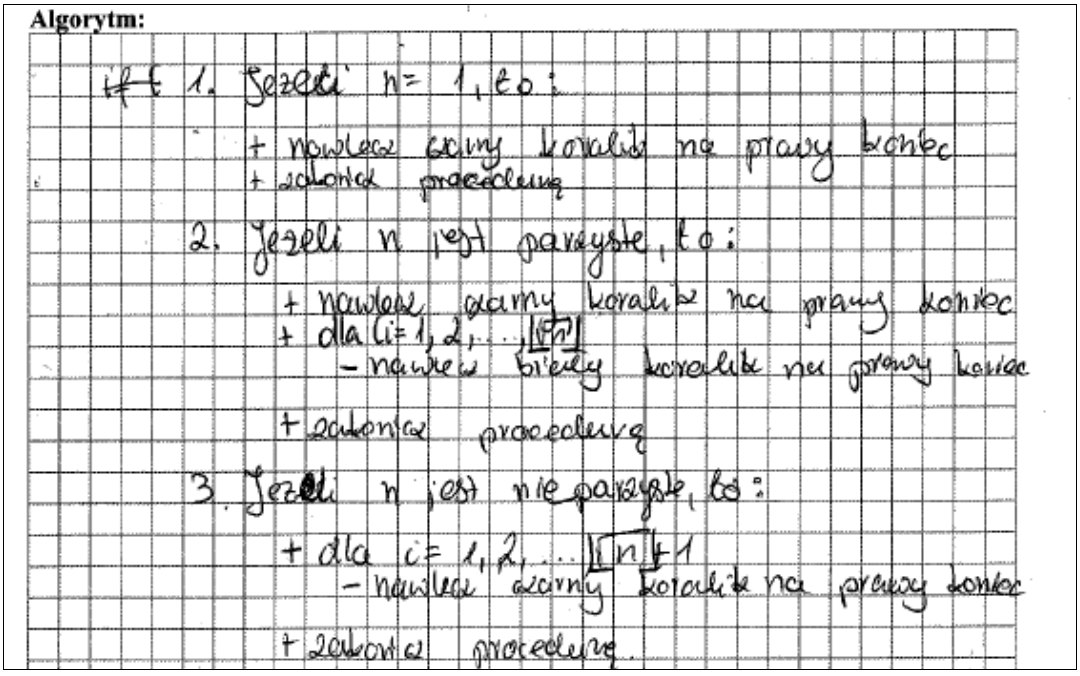
Częstym błędem zdających było zastosowanie w pętli dwóch instrukcji warunkowych „if", zamiast jednej instrukcji if-else, co skutkowało złą kolejnością koralików na końcu sznurka. W poniższym przykładzie po kilku iteracjach gdy *n* będzie już równe 2, wtedy w pętli while zostanie

wykonana pierwsza instrukcja warunkowa i po wyjściu z niej *n* będzie równe 1, zatem zostanie wykonywana druga instrukcja warunkowa, czyli: *nawlekamy czarny koralik*. Po zakończeniu działania pętli while ponownie nawlekany jest czarny koralik (pojawiają się więc dwa czarne korale na końcu).



Przykład 37: Zła kolejność korali (2 czarne na końcu).

Pojawiały się także rozwiązania pokazujące całkowity brak zrozumienia treści zadania i polecenia:



Przykład 38: Błędnie rozwiązane zadanie przez zdającego

Zadanie 2: Bisekcja (6 pkt)

Bisekcja jest jedną z metod szukania przybliżenia miejsca zerowego funkcji rzeczywistej $f(x)$, ciągłej w zadanym przedziale $< a, b >$ i o wartościach mających różne znaki na końcach przedziału.

*Algorytm bisekcji oblicza wartości funkcji na obu końcach przedziału, oraz w jego środku, tj. dla $\boldsymbol{x = \frac{a+b}{2}}$. Jeżeli wartość funkcji w środku przedziału jest zerem, to x jest szukanym miejscem zerowym tej funkcji. W przeciwnym przypadku zawęża się przedział $< a, b >$ do przedziału $< a, x >$ lub $< x, b >$ tak, aby na końcach tego nowego przedziału wartości funkcji znowu miały różne znaki. Wszystkie opisane czynności powtarza się, aż do znalezienia miejsca zerowego lub do zmniejszenia się długości analizowanego przedziału poniżej zadanej **dokładności d** – wówczas wynikiem jest środek ostatniego przedziału.*

Zadanie sprawdzało znajomość technik algorytmicznych, w tym metody dziel i zwyciężaj oraz metody wyznaczania miejsc zerowych funkcji. Zdający podejmowali próby rozwiązywania zadania.

Podpunkt a) polegał na analizie opisanego algorytmu bisekcji dla wybranej funkcji oraz przedziału.

Twoje zadania:

Dla funkcji $f(x) = x^3 - x - 2$ oraz przedziału $<0, 2>$:

a) Wykonaj trzy pierwsze kroki algorytmu bisekcji i uzupełnij tabelkę:

krok	*a*	*b*	*f(a)*	*f(b)*	$x = \frac{a+b}{2}$	*f(x)*	czy *f(a)* i *f(x)* mają te same znaki?
1	0	2	-2	4	1	-2	tak, więc wybieram przedział <x, b>
2	1	2	-2	4	1,5	-0,125	tak, więc wybieram przedział <x, b>
3	1,5	2	-0,125	4	1,75		

Przykład 39: Poprawne rozwiązanie zadania 2a

Zadanie nie sprawiło dużego problemu zdającym, jednak spora część popełniała błędy rachunkowe (poziom wykonania: 49%).

Poprawność rozwiązania polecenia b (poziom wykonania: 54%) była powiązana z właściwym zrozumieniem i rozwiązaniem polecenia a).

b) Podaj, w którym **kroku** algorytmu bisekcji długość analizowanego przedziału $<a, b>$ będzie po raz pierwszy mniejsza niż 0,1.

1 -> 2
2 -> 1
3 -> 0,5
4 -> 0,25
5 -> 0,125
6 -> 0,0625

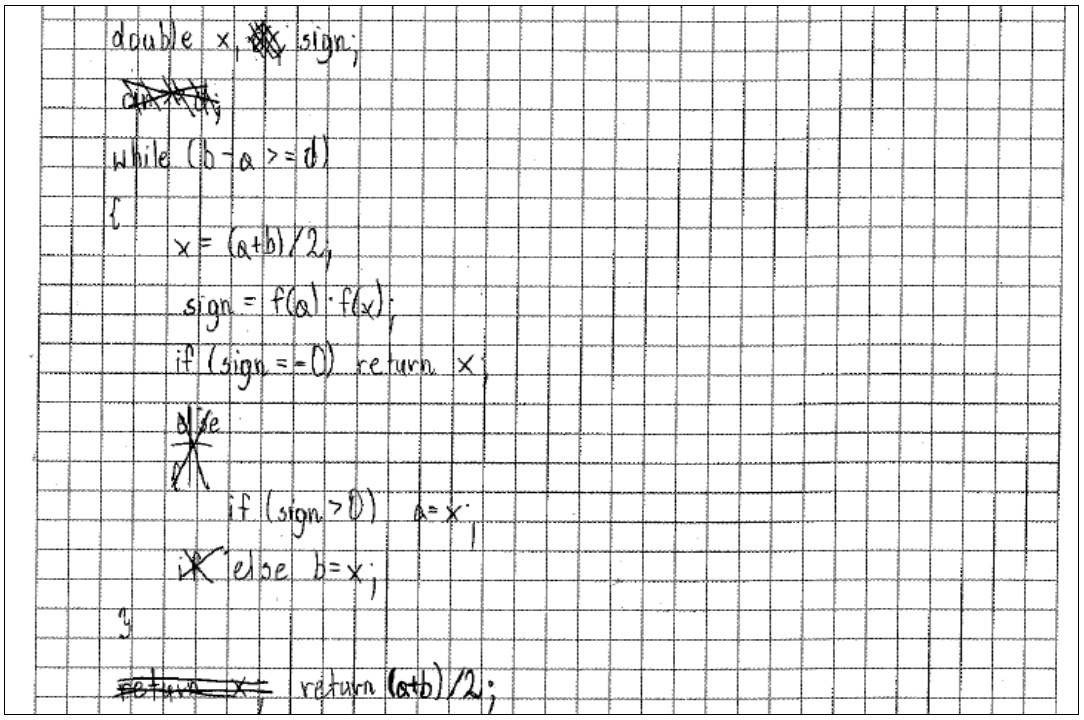
Odp. Długość analizowanego przedziału będzie mniejsza niż 0,1 w 6. kroku po raz pierwszy.

Przykład 40: Poprawne rozwiązanie zadania 2b.

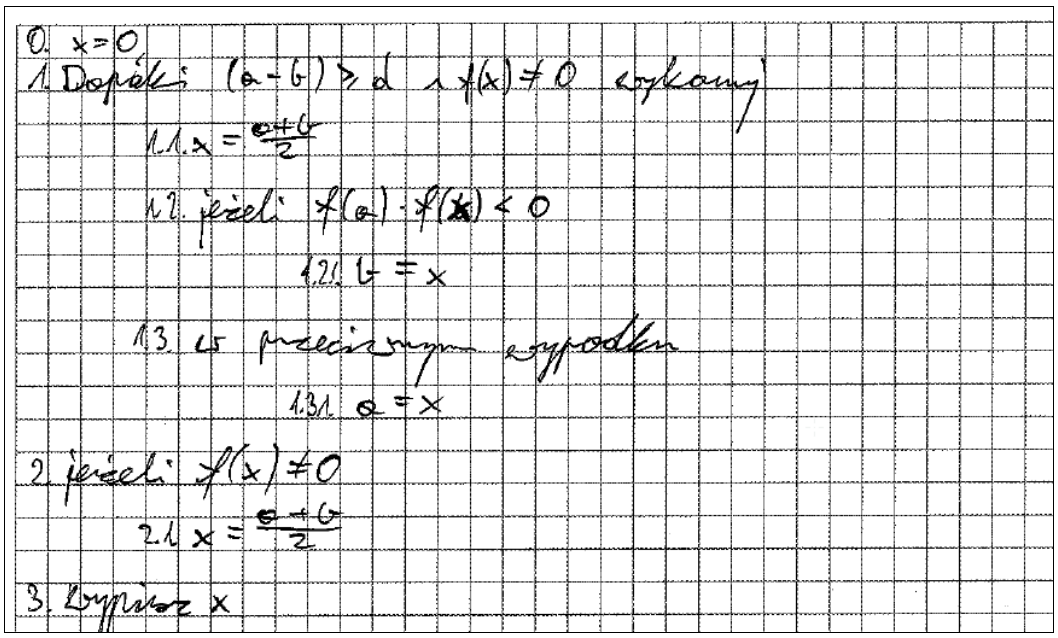
Ostatnie polecenie sprawdzało umiejętność zaprojektowania algorytmu, który poda przybliżenie miejsca zerowego funkcji f w przedziale $< a,b >$, przy zadanej dokładności d. Większość zdających podjęła próbę rozwiązania tego punktu, jednak część zdających popełniała błędy w rozwiązaniu (poziom wykonania: 33%).

Najczęstszym pojawiającym się błędem było niewłaściwe wyrażenie logiczne w pętli. Zamiast porównywania wartości bezwzględnej z wartości funkcji z zadaną dokładnością (while (abs f(x)>d) {...}) porównywano ostatnie przybliżenie miejsca zerowego z zadaną dokładnością (while (x>d) {...}).

Występowały głównie dwa sposoby rozwiązania zadania. Bardzo wiele prac zawierało rozwiązania iteracyjne, w których w pętli dokonywano aktualizacji krańców przedziału.

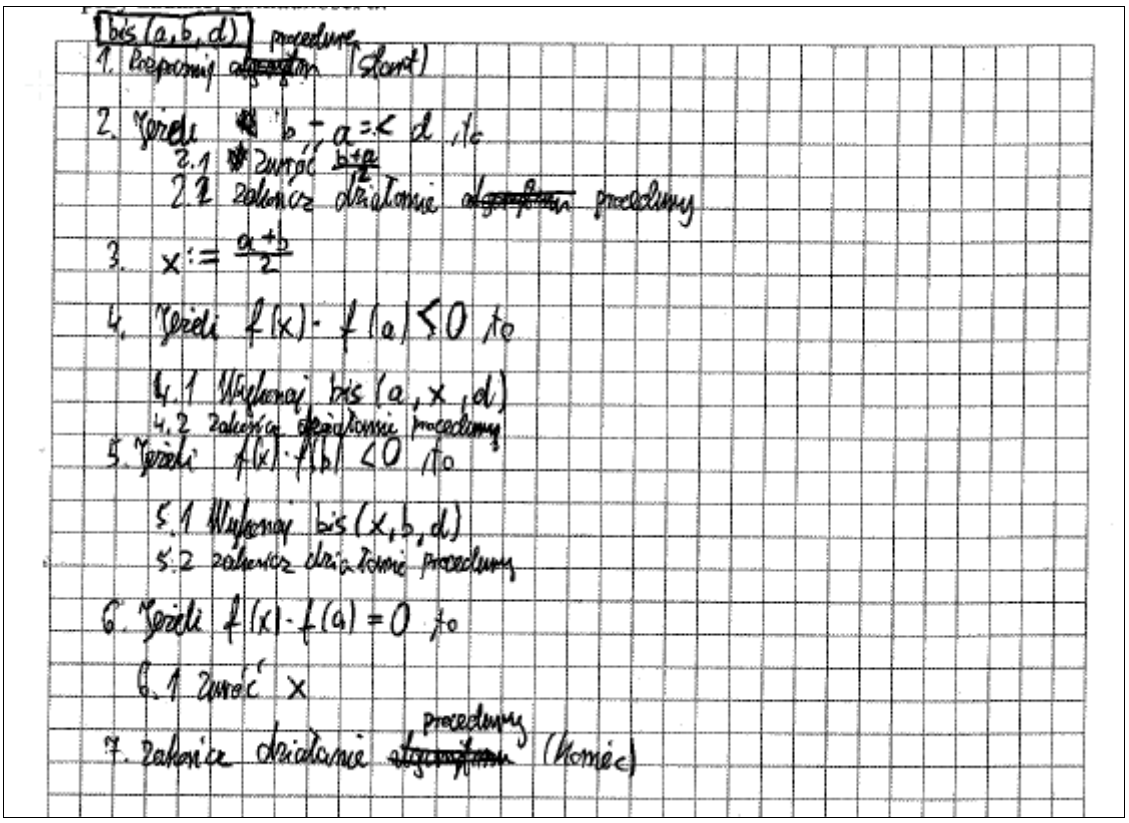


Przykład 41: Rozwiązanie iteracyjne zadania 2c

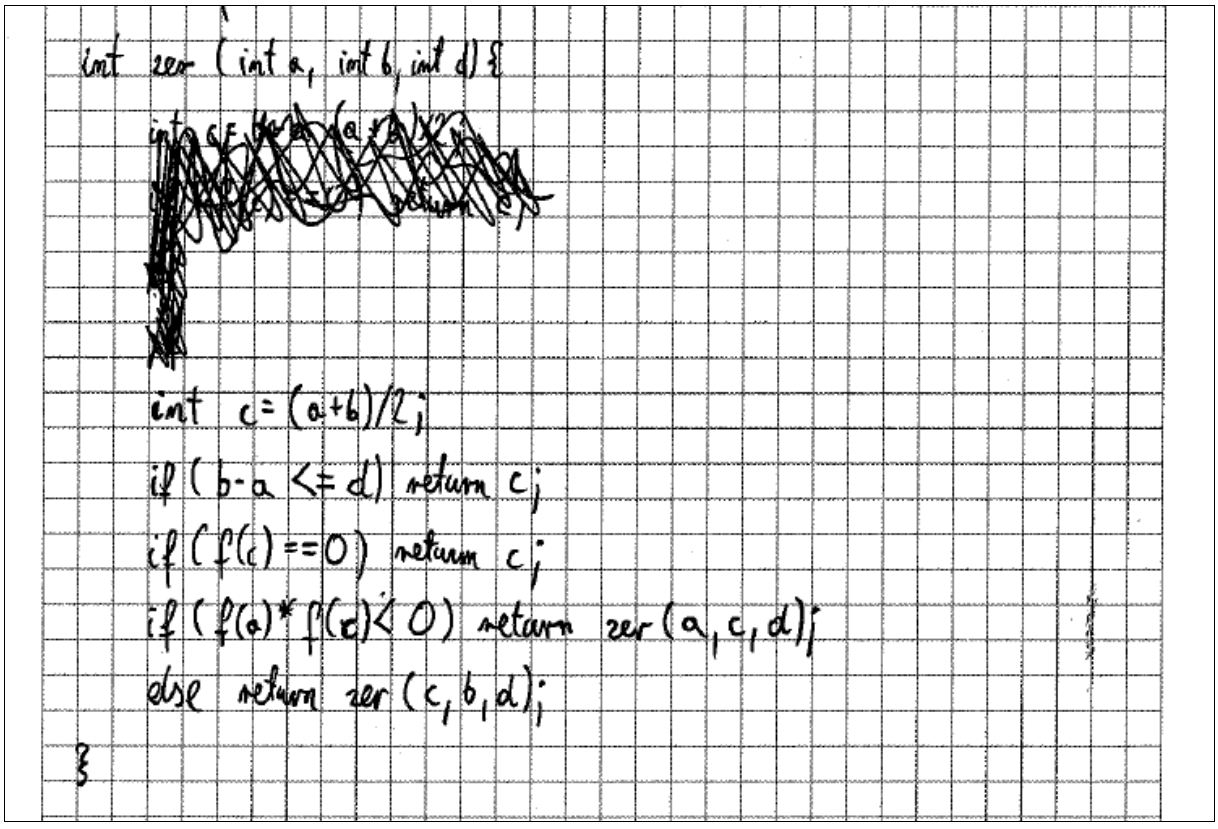


Przykład 42 Rozwiązanie iteracyjne zadania 2c

Część zdających wybrało podejście rekurencyjne, z różnie zapisywanymi warunkami zakończenia wywołań rekruencyjnych.

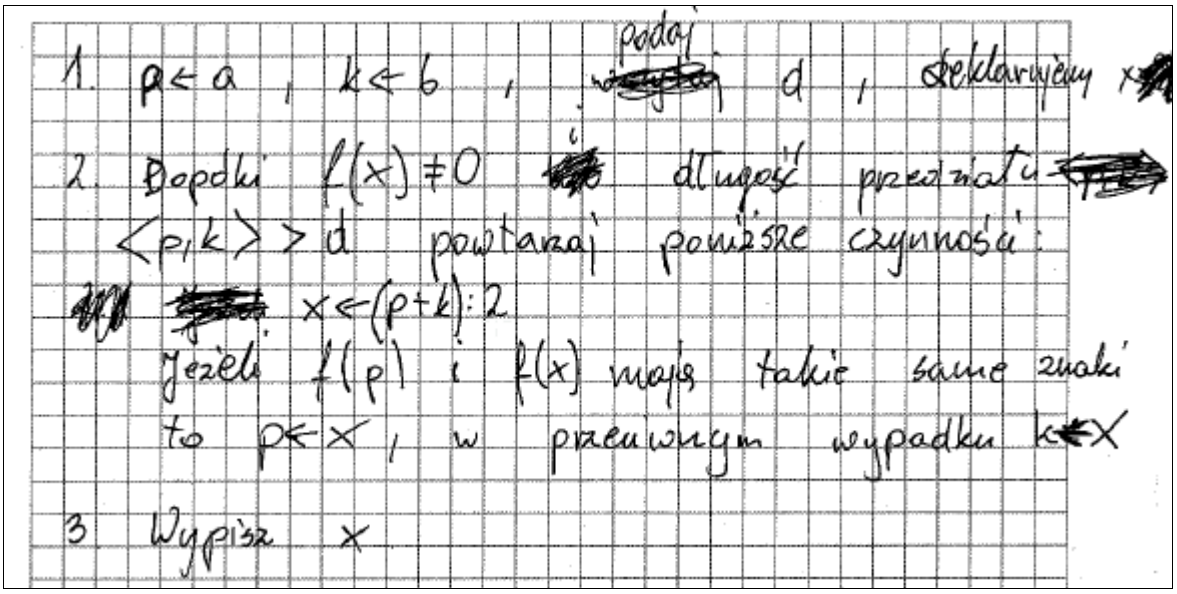


Przykład 43: Rekurencyjne rozwiązanie zadania 2c

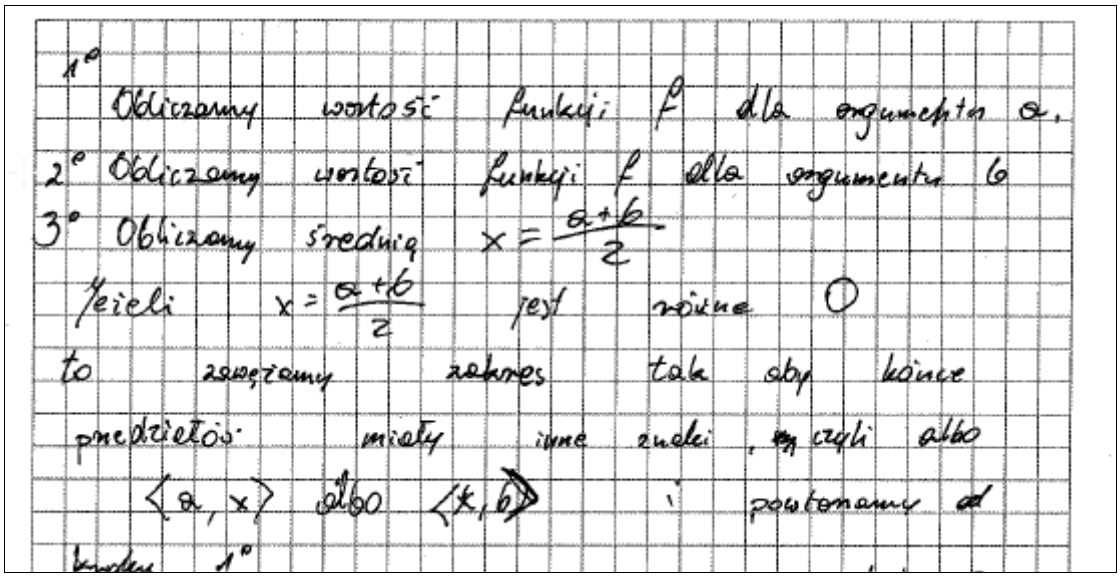


Przykład 44: Rekurencyjne rozwiązanie zadania 2c

Na końcu przedstawione są dwa przykłady „informatycznej prozy” – nadal i takie rozwiązania pojawiają się wśród maturzystów. Pierwsze rozwiązanie jest poprawne, w drugim wyraźnie widać problemy z zapisem algorytmu.



Przykład 45: Rozwiązanie poprawne zadania 2c „prozą”.



Przykład 46: BŁĘDNE rozwiązanie *„prozą”* zadania 2c

Zadanie 4. Ferma (9 pkt)

Pani Binarna została właścicielką kurzej fermy, na której znajduje się 200 kur niosek. Kilogram paszy kosztuje 1,9 zł, a jedna kura zjada przez cały dzień 0,2 kg paszy.

Rozważmy okres hodowli trwający sto osiemdziesiąt dni. Dni są ponumerowane od 1 do 180. Pierwsza niedziela przypada siódmego dnia.

*Codziennie w południe, **z wyjątkiem niedziel**, każda kura znosi 1 jajko i tego samego dnia pani Binarna sprzedaje je w cenie **0,9 zł** za sztukę.*

*W okolicach fermy grasuje lis, który w każdy nieparzysty dzień, po zmroku (po posiłku kur, po zniesieniu i ewentualnym sprzedaniu jajek) zmniejsza liczbę kur o **2 sztuki naraz**.*

*Co **30 dni** (w dniu trzydziestym, sześćdziesiątym itd.) rano, przed posiłkiem kur, pani Binarna powiększa stado kur o **20%,** kupując kolejne kury na giełdzie (liczba kupionych kur zaokrąglamy w dół do liczby całkowitej), po **18 zł** za sztukę.*

Biorąc pod uwagę podane założenia i wykorzystując dostępne narzędzia informatyczne, wykonaj poniższe polecenia. Odpowiedzi do podpunktów a), b), c) zapisz w kolejnych wierszach pliku tekstowego zadanie4.txt. Wykres, o którym mowa w podpunkcie d), utwórz w oddzielnym pliku. Odpowiedź do każdego podpunktu poprzedź literą oznaczającą ten podpunkt.

Zadanie czwarte miało charakter symulacyjny, w którym dla zadanych warunków początkowych oraz podanych zależności należało dokonać symulacji cyklicznie powtarzających się wydarzeń. Zadania tego typu coraz częściej pojawiają się na poziomie rozszerzonym egzaminu maturalnego z informatyki.

Kluczowym krokiem w kierunku rozwiązania wszystkich podpunktów jest zasymulowanie procesu opisanego w zadaniu (rozpisanie szczegółowo wydarzeń z każdego dnia na fermie), co bez większych trudności można zrobić przy pomocy arkusza kalkulacyjnego. Następnie z wykorzystaniem odpowiednik funkcji logicznych i arytmetycznych można było wydobyć informacje wymagane w rozwiązaniach punktów a, b i c oraz wykonać wykres w punkcie d.

Maturzyści, którzy pokusili się w swoich rozwiązaniach o *„skróty myślowe”*, często popełniali błędy w obliczeniach i stąd otrzymywali zero punktów w poleceniu b (poziom wykonania: 42%) i poleceniu c (poziom wykonania: 47%).

	A	B	C	D	E	F	G	H	I	J
1		mod 7	liczba kur	zarobek	wydatki	koszt paszy	lis	po lisie	dzienny zysk	realny zysk
2	1	=MOD(A2;7)	200	=JEŻELI(B2=0;0;C2*$	=JEŻELI(MOD(A2;30	=C2*R1	=JEŻELI(MOD(A2;2)=	=C2-G2	=D2-E2-F2	=I2
3	2	=MOD(A3;7)	=JEŻELI(MOD(A3;30	=JEŻELI(B3=0;0;C3*$	=JEŻELI(MOD(A3;30	=C3*R1	=JEŻELI(MOD(A3;2)=	=C3-G3	=D3-E3-F3	=J2+I3
4	3	=MOD(A4;7)	=JEŻELI(MOD(A4;30	=JEŻELI(B4=0;0;C4*$	=JEŻELI(MOD(A4;30	=C4*R1	=JEŻELI(MOD(A4;2)=	=C4-G4	=D4-E4-F4	=J3+I4
5	4	=MOD(A5;7)	=JEŻELI(MOD(A5;30	=JEŻELI(B5=0;0;C5*$	=JEŻELI(MOD(A5;30	=C5*R1	=JEŻELI(MOD(A5;2)=	=C5-G5	=D5-E5-F5	=J4+I5
6	5	=MOD(A6;7)	=JEŻELI(MOD(A6;30	=JEŻELI(B6=0;0;C6*$	=JEŻELI(MOD(A6;30	=C6*R1	=JEŻELI(MOD(A6;2)=	=C6-G6	=D6-E6-F6	=J5+I6
7	6	=MOD(A7;7)	=JEŻELI(MOD(A7;30	=JEŻELI(B7=0;0;C7*$	=JEŻELI(MOD(A7;30	=C7*R1	=JEŻELI(MOD(A7;2)=	=C7-G7	=D7-E7-F7	=J6+I7
8	7	=MOD(A8;7)	=JEŻELI(MOD(A8;30	=JEŻELI(B8=0;0;C8*$	=JEŻELI(MOD(A8;30	=C8*R1	=JEŻELI(MOD(A8;2)=	=C8-G8	=D8-E8-F8	=J7+I8
9	8	=MOD(A9;7)	=JEŻELI(MOD(A9;30	=JEŻELI(B9=0;0;C9*$	=JEŻELI(MOD(A9;30	=C9*R1	=JEŻELI(MOD(A9;2)=	=C9-G9	=D9-E9-F9	=J8+I9
10	9	=MOD(A10;7)	=JEŻELI(MOD(A10;3	=JEŻELI(B10=0;0;C1	=JEŻELI(MOD(A10;3	=C10*R1	=JEŻELI(MOD(A10;2	=C10-G10	=D10-E10-F10	=J9+I10
11	10	=MOD(A11;7)	=JEŻELI(MOD(A11;3	=JEŻELI(B11=0;0;C1	=JEŻELI(MOD(A11;3	=C11*R1	=JEŻELI(MOD(A11;2	=C11-G11	=D11-E11-F11	=J10+I11
12	11	=MOD(A12;7)	=JEŻELI(MOD(A12;3	=JEŻELI(B12=0;0;C1	=JEŻELI(MOD(A12;3	=C12*R1	=JEŻELI(MOD(A12;2	=C12-G12	=D12-E12-F12	=J11+I12
13	12	=MOD(A13;7)	=JEŻELI(MOD(A13;3	=JEŻELI(B13=0;0;C1	=JEŻELI(MOD(A13;3	=C13*R1	=JEŻELI(MOD(A13;2	=C13-G13	=D13-E13-F13	=J12+I13
14	13	=MOD(A14;7)	=JEŻELI(MOD(A14;3	=JEŻELI(B14=0;0;C1	=JEŻELI(MOD(A14;3	=C14*R1	=JEŻELI(MOD(A14;2	=C14-G14	=D14-E14-F14	=J13+I14
15	14	=MOD(A15;7)	=JEŻELI(MOD(A15;3	=JEŻELI(B15=0;0;C1	=JEŻELI(MOD(A15;3	=C15*R1	=JEŻELI(MOD(A15;2	=C15-G15	=D15-E15-F15	=J14+I15
16	15	=MOD(A16;7)	=JEŻELI(MOD(A16;3	=JEŻELI(B16=0;0;C1	=JEŻELI(MOD(A16;3	=C16*R1	=JEŻELI(MOD(A16;2	=C16-G16	=D16-E16-F16	=J15+I16
17	16	=MOD(A17;7)	=JEŻELI(MOD(A17;3	=JEŻELI(B17=0;0;C1	=JEŻELI(MOD(A17;3	=C17*R1	=JEŻELI(MOD(A17;2	=C17-G17	=D17-E17-F17	=J16+I17
18	17	=MOD(A18;7)	=JEŻELI(MOD(A18;3	=JEŻELI(B18=0;0;C1	=JEŻELI(MOD(A18;3	=C18*R1	=JEŻELI(MOD(A18;2	=C18-G18	=D18-E18-F18	=J17+I18

Przykład 47: Fragment poprawnego zapisu kolejnych kroków symulacji

Jedna błędne zapisy warunków zadania (kolejnych kroków symulacji) nie przekreślały szans zdających na otrzymanie punktów za polecenie a (poziom wykonania: 65%) oraz polecenie d (poziom wykonania: 63%). Część zdających, którzy poprawnie rozwiązali podpunkt pierwszy zadania, wpisywała do pliku tekstowego złą odpowiedź. Błąd polegał na podaniu w odpowiedzi numeru wiersza z arkusza kalkulacyjnego zamiast numeru wiersza ich obliczeń (podawali 34 wiersz, zamiast 33 dnia).

Zadanie 5. Ciekawe napisy (10 pkt)

*W pliku NAPIS.TXT w oddzielnych wierszach znajduje się **1 000** napisów o długościach od **2** do **25** znaków. Każdy napis składa się z wielkich liter alfabetu łacińskiego.*

Wykorzystując dostępne narzędzia informatyczne, daj odpowiedzi do poniższych podpunktów. Odpowiedzi zapisz w kolejnych wierszach pliku ZADANIE5.TXT, a każdą poprzedź literą oznaczającą ten podpunkt.

a) Napis pierwszy to taki napis, w którym suma kodów ASCII jest liczbą pierwszą. Przykładowo, suma kodów ASCII w napisie ABB wynosi 197 i jest liczbą pierwszą, co oznacza, że napis ABB jest napisem pierwszym. Podaj, ile jest napisów pierwszych w pliku NAPIS.TXT.

b) Napis rosnący to taki napis, w którym kod ASCII każdej kolejnej litery jest większy od kodu poprzedniej. Podaj wszystkie napisy rosnące występujące w pliku NAPIS.TXT.

c) Wypisz napisy z pliku NAPIS.TXT, które występują w nim więcej niż jeden raz (każde takie słowo wypisz tylko raz).

Zadanie miało charakter programistyczny, ale w treści nie wymagano wprost, aby rozwiązanie było uzyskane przy pomocy samodzielnie napisanego programu komputerowego. Pomimo tego większość zdających pisało program w wybranym przez siebie języku programowania.

Rozwiązanie polecenia a) polegało na zamianie kolejnych liter napisu na kody ASCII, zsumowaniu otrzymanych liczb i zastosowaniu algorytmu sprawdzającego, czy otrzymana suma jest liczbą pierwszą (poziom wykonania: 28%).

Poniżej przedstawiamy przykładowe fragmenty rozwiązania zdającego, który w osobnych funkcjach sumował kody ASCII znaków z zadanego napisu *w*, a następnie drugą funkcją sprawdzał, czy zadawana liczba *n* jest pierwsza. Te dwie funkcje zostały wykorzystane do sprawdzenia napisów znajdujących się w pliku.

```
1  int suma(string w) //suma kodow ASCII
2  {
3   int i,s=0;
4   for (i=0; i<=w.length()-1; i++)
5         s=s+w[i];
6   return s;
7
8  }
9
10 bool is_prime(int n) //czy_pierwsza
11 {
12    int i=3;
13    bool odp=true;
14
15    if ((n==1) || ((n%2==0) && (n!=2))) odp=false;
16    while ((i<=int(sqrt(n))) && odp) {
17         if (n % i == 0)  odp=false;
18         else i++;
19    }
20    return odp;
21
22 }
```

Przykład 48: Dwie funkcje niezbędne do rozwiązania zadania 5a

```
ifstream fin;
ofstream fout;
string wiersz;
fin.open("NAPIS.TXT");
fout.open("ZADANIE5A.TXT");

while (!fin.eof()) {
      getline(fin,wiersz);
      if (is_prime(suma(wiersz))) {
         fout <<wiersz;
         fout<<endl;
      }
}
```

Przykład 49: Fragment poprawnego rozwiązania zadania 5a wykorzystujący powyższe funkcje

Większym wyzwaniem okazał się być podpunkt b) (poziom wykonania: 27%), w którym należało sprawdzić, który napis jest napisem rosnącym, to znaczy czy każdy kolejna litera w napisie jest większa od poprzedniej w sensie leksykograficznym. Do rozwiązania zadania wystarczyła umiejętność porównywania kolejnych znaków w napisie.

```
bool czyrosnacy (string a)
{
    int i=0;
    while(a[i]<a[i+1]&&i<a.length()-1)
        i++;
    if (i==a.length()-1)
        return true;
    return false;
}
```

Przykład 50: Poprawnie skonstruowana funkcja do zadania 5b

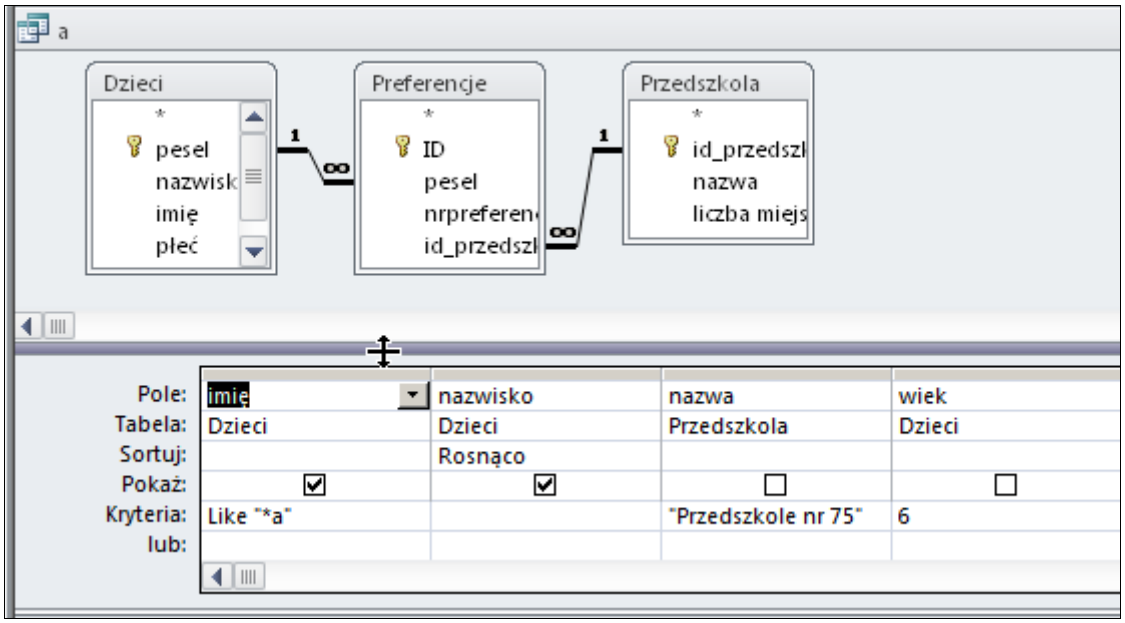
```
int main()
{
    ifstream fin;
    ofstream fout;
    fin.open("napis.txt");
    fout.open("wynik_b.txt");
    string napis;
    int dl=0, i=0;
    bool b=true;
    while(!fin.eof())
    {
          fin>>napis;
          dl=napis.length();
          b=true;
          if(!fin.fail())
          {
                for(i=1; i<dl; i++)
                {
                    if(napis[i]<napis[i-1])
                    {
                        b=false;
                    }
                }
                if(b==true)
                {
                      cout<<napis<<endl;
                      fout<<napis<<endl;
                 }
          }
    }
    fin.close();
    fout.close();
    return 0;
}
```

Przykład 51: Rozwiązanie zadania 5b

Polecenia a) (poziom wykonania: 28%) oraz c) (poziom wykonania: 33%), to typowo szkolne zadania, które często pojawiały się w zadaniach na egzaminach maturalnych w poprzednich latach (zarówno w pierwszej, jak i drugiej części egzaminu). Polecenie b) wymagało zastanowienia się i zaimplementowania (nietrudnego) algorytmu, jednak był to podpunkt najczęściej opuszczany przez zdających.

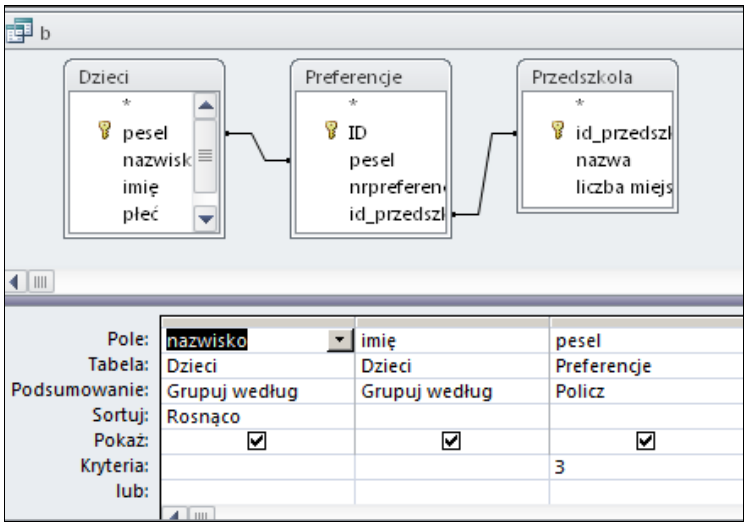
Zadanie 6 było zadaniem bazodanowym. Sprawdzało zastosowanie metod wyszukiwania i przetwarzania informacji w relacyjnych bazach danych z wykorzystaniem różnych technik i narzędzi. Większość zdających rozwiązywała zadanie za pomocą systemu zarządzania bazami danych, ale pojawiały się też rozwiązania w arkuszu kalkulacyjnym.

Najmniej problemów sprawiały zapytania w poleceniu a oraz b. W pierwszym należało wybrać sześcioletnie dziewczynki, które na liście preferencji mają Przedszkole nr 75 (poziom wykonania: 72%).



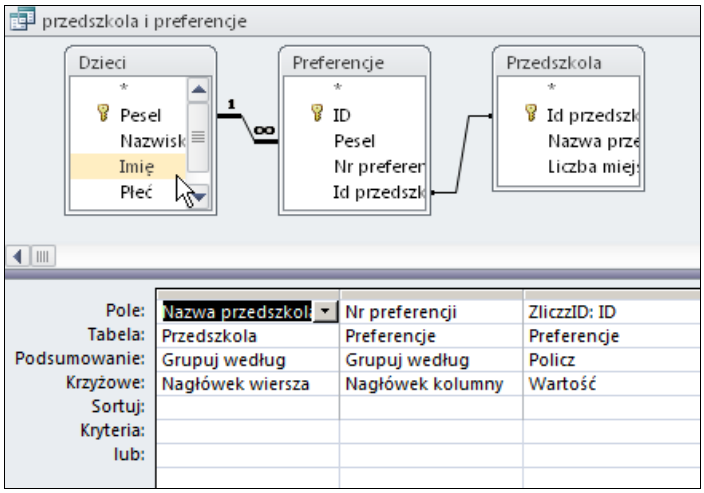
Przykład 52: Przykład poprawnie skonstruowanej kwerendy do zadania 6a

Zaś w drugim poleceniu (poziom wykonania: 71%) należało podać imiona i nazwiska 3 dzieci, które na liście preferencji mają dokładnie po 3 przedszkola:



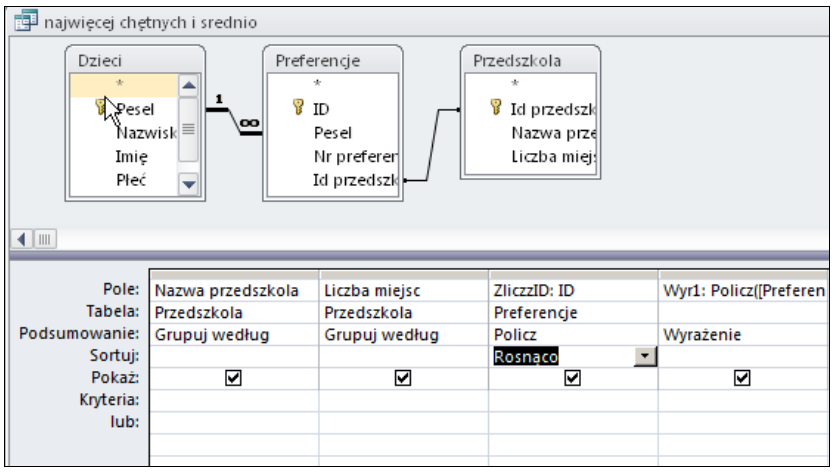
Przykład 53: Przykład poprawnie skonstruowanej kwerendy do zadania 6b

Pozostałe dwa polecenia sprawiły zdającym najwięcej trudności. Polecenie c można było rozwiązać na wiele sposobów. Najprostszym rozwiązaniem było utworzenie kwerendy krzyżowej (nazwy przedszkoli w wierszach, preferencje w kolumnach, liczba dzieci jako wartości), a następnie sprawdzenie, w których przedszkolach na drugiej i trzeciej preferencji nie było chętnych (poziom wykonania: 41%).

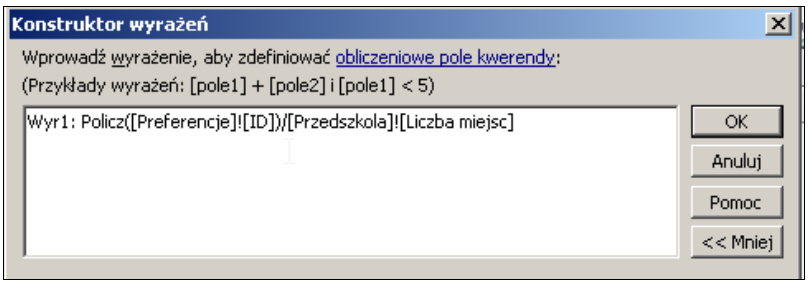


Przykład 54: Przykład kwerendy krzyżowej do zadania 6c.

Nieco mniej trudności sprawiło zdającym polecenie d, w którym było tu zdecydowanie najmniej poprawnych rozwiązań, był to też podpunkt najczęściej opuszczany (poziom wykonania: 47%). W zadaniu należało obliczyć średnią liczbę chętnych dzieci na miejsce oraz podać informacje o 3 przedszkolach, dla których liczba chętnych dzieci jest najmniejsza. Istotną trudność stanowiło zbudowanie odpowiedniego wyrażenia, które wyznaczy średnią liczbę chętnych dzieci na jedno miejsce oraz odpowiednie grupowanie i zliczanie:



Przykład 55: Poprawnie zbudowana kwerenda do zadania 6d ze skonstruowanym wyrażeniem.



Przykład 56: Konstrukcja wyrażenia do powyższego rozwiązania zadania 6d.

3. Wnioski i rekomendacje

Kluczową dziedziną informatyki jako przedmiotu ogólnokształcącego jest algorytmika. Zagadnienia algorytmiczne pojawiają się we wszystkich standardach egzaminacyjnych (standardy mówią zarówno o znajomości pewnych algorytmów, jak i umiejętności ich konstruowania, analizowania i stosowania). We współczesnym świecie obserwujemy zjawisko nazywane czasem „algorytmizacją" nauki. Coraz więcej dziedzin opisuje i charakteryzuje badane procesy i zjawiska w sposób algorytmiczny (dotyczy to nie tylko kierunków ścisłych i technicznych; sztandarowym przykładem może być np. biologia molekularna). Akademickie i korporacyjne ośrodki badawcze różnych branż potrzebują coraz więcej specjalistów potrafiących programować oraz swobodnie posługiwać się terminologią algorytmiczną. Tegoroczny egzamin maturalny (na obu poziomach) sprawdzał umiejętności i wiedzę z zakresu algorytmiki w dwóch pierwszych zadaniach arkusza I oraz umiejętność konstruowania algorytmów wraz z tworzeniem w oparciu o nie programów komputerowych w drugim zadaniu arkusza II. Wyniki tych zadań w części praktycznej są niższe od średnich wyników całego egzaminu (szczególnie na poziomie podstawowym), co po części wynika ze stosunkowo małego nacisku na te zagadnienia w szkołach, a także z faktu, że są to tematy stosunkowo trudne. Innym powodem niepowodzenia było oddawanie do oceny przez zdających wyłącznie plików tekstowych z odpowiedziami (prawidłowymi!), pomijając pliki źródłowe (nie tylko w zadaniach programistycznych). Skutkowało to przyznaniem im przez egzaminatorów 0 punktów za dane zadanie. Zgodnie ze schematem punktowania wszystkie wyniki muszą być odzwierciedleniem komputerowej realizacji obliczeń (w instrukcji dla zdającego na stronie tytułowej oraz w treści każdego zadania znajduje się informacja o konieczności dołączenia plików źródłowych zawierających komputerową realizację rozwiązanych zadań). Poprawiły się średnie rezultaty za te zadania w części teoretycznej egzaminu maturalnego i są wyższe od średnich wyników całego egzaminu, zarówno na poziomie podstawowym, jak i rozszerzonym.

Drugim najważniejszym obszarem zagadnień egzaminu maturalnego z informatyki jest rozwiązywanie konkretnych problemów z zastosowaniem narzędzi technologii informacyjnej (arkusz kalkulacyjny, system baz danych). Zdający muszą się tutaj wykazać nie tylko znajomością tych narzędzi, ale również umiejętnością wyrażenia celów opisanych w zadaniach w precyzyjny matematycznie i „zalgorytmizowany" sposób. W tegorocznym zestawach maturalnym wystąpiły po dwa zadania tego typu, w arkuszu II. Wyniki uzyskane z tych zadań są stosunkowo dobre i porównywalne ze średnim wynikiem z całego egzaminu.

Uzyskane wyniki ujawniają, że w dużej mierze osiągnięcia uczniów zależą od zastosowania przez nauczyciela prawidłowej ścieżki przygotowań do egzaminu. Rozwiązanie zestawu egzaminacyjnego, zarówno na poziomie podstawowym, jak rozszerzonym, wymagało rozumienia algorytmów i umiejętności operowania nimi oraz prowadzenia prostego rozumowania i doboru własnych strategii, które także prowadziły do rozwiązania problemu dla typowych i nietypowych danych. Na lekcjach informatyki warto zwrócić uwagę na ćwiczenia doskonalące rozumienie i stosowanie algorytmów, wykorzystywanie posiadanej w rozwiązywaniu zadań z różnych dziedzin i przy pomocy różnych narzędzi informatycznych. Warto uświadomić uczniom potrzebę uważnego czytania, tworzenia planu rozwiązania zadania i jego wykonania, a także krytycznej oceny rozwiązania. Nauczyciel w czasie zajęć powinien zwracać uwagę i eliminować błędne nawyki uczniów oraz niestaranność w rozwiązaniu zadań (pobieżne czytanie poleceń, podawanie wyników z inną dokładnością niż żądana w zadaniu, posługiwanie się w odpowiedziach językiem potocznym, nieprecyzyjnym).