INFORMATYKA

Poziom rozszerzony

Egzamin maturalny – Formuła 2023 · arkusz próbny nr 4

Imię i nazwisko: .....

WYBRANE:

.....
(system operacyjny)

.....
(program użytkowy)

.....
(środowisko programistyczne)

CZAS TRWANIA:	210 minut
LICZBA PUNKTÓW DO UZYSKANIA:	50
ZAŁĄCZNIK:	katalog DANE (archiwum matura-informatyka-pr-arkusz-4-dane.zip)

Arkusz przygotowany według zasad egzaminu maturalnego z informatyki w Formule 2023 (budowa arkusza, rodzaje zadań i sposób oddawania rozwiązań jak w arkuszach CKE 2023–2026; oprogramowanie zgodnie z komunikatem dyrektora CKE z 20 sierpnia 2026 r.). Zadania sprawdzają wymagania podstawy programowej w brzmieniu z Dz.U. 2024 poz. 1019. To nie jest arkusz CKE; treści, dane i historie są oryginalne. Materiał nie jest związany z CKE ani z producentami oprogramowania – nazwy programów, systemów i instytucji podano wyłącznie informacyjnie.

Instrukcja dla zdającego

1. Zanim zaczniesz, sprawdź, czy arkusz ma 19 stron (zadania 1–8) i czy masz katalog **DANE** z plikami do zadań (archiwum **matura-informatyka-pr-arkusz-4-dane.zip**). Jeśli czegoś brakuje, powiedz o tym nauczycielowi.
2. Na pierwszej stronie wpisz imię i nazwisko oraz wybrane przez siebie (na egzaminie – zadeklarowane) oprogramowanie: system operacyjny, program użytkowy i środowisko programistyczne. Według komunikatu dyrektora CKE z 20 sierpnia 2026 r. w 2027 r. do wyboru są: system **Windows** z pakietem Microsoft Office 2007 lub nowszym (Word, Excel, Access) albo z LibreOffice 7.0 lub nowszym (Writer, Calc, Base) lub OpenOffice/Apache OpenOffice 4.1 lub nowszym, albo system **Linux** z LibreOffice 7.0 lub nowszym lub OpenOffice/Apache OpenOffice 4.1 lub nowszym – zawsze razem z pakietem XAMPP (Apache, MySQL/MariaDB, PHP, phpMyAdmin) – oraz jeden język programowania: C/C++, Java (Java SE 8 lub nowsza) albo Python (3.10 lub nowszy).
3. Zadanie z symbolem ✎ w nagłówku rozwiązujesz bez komputera – odpowiedź wpisz wyłącznie w arkuszu, w miejscu na nią przeznaczonym.
4. Programy pisz w wybranym języku programowania. Wszystkie utworzone przez siebie pliki źródłowe zapisz w katalogu nazwanym Twoim imieniem i nazwiskiem (na egzaminie katalog oznacza się numerem PESEL).
5. Jeżeli algorytm zapisywany w arkuszu przedstawiasz w języku programowania, a nie w pseudokodzie, użyj języka zadeklarowanego na egzamin (C++, Java lub Python).
6. Przy zadaniu z bazą danych w MySQL lub MariaDB zapisz w katalogu treści zapytań SQL, a przed końcem pracy wyeksportuj całą bazę do pliku *.sql. W programie Access lub Base oddajesz plik bazy z kwerendami nazwanymi tak, jak podano w zadaniu.
7. Każdy plik do oceny nazwij dokładnie tak, jak podano w zadaniu (rozszerzenie zależy od używanego programu) – pliki nazwane inaczej nie są sprawdzane. Zadanie praktyczne bez komputerowego rozwiązania otrzymuje 0 punktów.
8. Przed końcem czasu pracy zapisz w katalogu ostateczne wersje plików z rozwiązaniami.
9. Pisz czytelnie, długopisem lub piórem z czarnym tuszem (atramentem). Nie używaj korektora – błędny zapis wyraźnie przekreśl. Zapisów w brudnopisie nikt nie ocenia.
10. Kratki z punktacją przy zadaniach wypełnia egzaminator – nic w nich nie wpisuj.
11. Na egzaminie wtyczki AI generujące kod lub treść (np. Copilot) są w środowiskach programistycznych i w pakiecie biurowym odinstalowane albo wyłączone – rozwiązuj arkusz bez nich.
12. Programy do zadań 3. i 4. możesz dodatkowo sprawdzić na stronie **pierwszalawka.pl/kod-na-maturze** – wpisz kod zadania podany przy poleceniu. Narzędzie sprawdza wynik danego zadania (z numerem zadania albo bez), wypisany na ekran albo zapisany w pliku wyniki*.txt, na danych z arkusza i na dwóch innych zestawach danych.

Zadanie 2. Najmniejszy i największy

Dana jest tablica $A[1..n]$ liczb całkowitych, gdzie n jest liczbą parzystą. Chcemy jednocześnie znaleźć najmniejszy i największy element tablicy. Algorytm „naiwny” szuka osobno minimum ($n - 1$ porównań elementów) i osobno maksimum ($n - 1$ porównań). Algorytm „parami” dzieli tablicę na pary sąsiednich elementów, porównuje elementy w każdej parze, a następnie mniejszy z nich porównuje z bieżącym minimum, a większy – z bieżącym maksimum.

Zadanie 2.1. (0–1)

2.1.
0–1

Uzupełnij tabelę – podaj liczbę porównań elementów tablicy wykonywanych przez oba algorytmy. Algorytm „parami” pierwszą parę porównuje tylko ze sobą (mniejszy element staje się bieżącym minimum, a większy – bieżącym maksimum).

n	algorytm „naiwny”	algorytm „parami”
10	18	13
100		
1000		

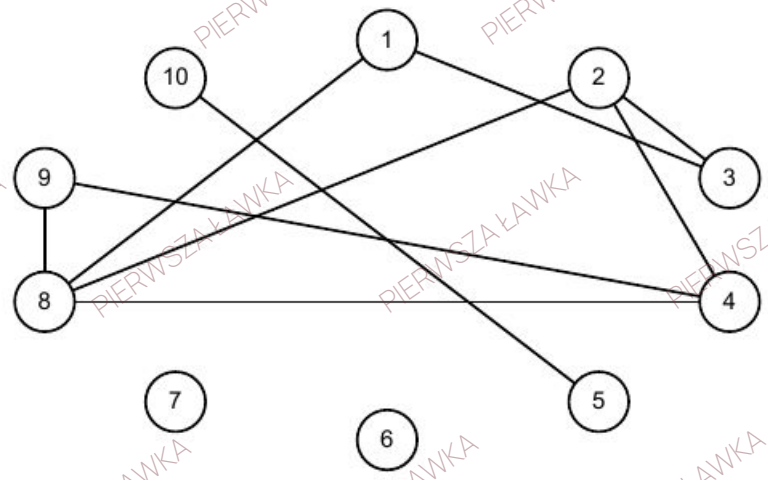
Miejsce na obliczenia (brudnopis)

PIERWSZA ŁAWKA

Zadanie 3. Sieć dróg

W pewnym regionie jest n miast ponumerowanych od 1 do n. Niektóre pary miast są połączone bezpośrednią drogą dwukierunkową. W pierwszym wierszu pliku **polaczenia.txt** zapisano dwie liczby: n – liczbę miast i m – liczbę dróg (w pliku z arkusza n = 200, m = 260). W każdym z kolejnych m wierszy zapisano numery dwóch różnych miast połączonych drogą. Każda para miast występuje w pliku co najwyżej raz. *Obszarem* nazywamy największy zbiór miast, w którym z każdego miasta można dojechać do każdego innego (być może przez inne miasta); miasto, do którego nie prowadzi żadna droga, tworzy osobny obszar.

Na rysunku przedstawiono sieć dróg z pliku **polaczenia_przyklad.txt**:



Rysunek. Sieć dróg z pliku polaczenia_przyklad.txt (koła – miasta, odcinki – drogi)

Pierwsze wiersze pliku polaczenia.txt:

```
200 260
97 66
14 80
158 131
22 50
```

Napisz program (lub kilka programów), który(-e) znajdzie(-ą) i da(dzą) odpowiedzi do podanych zadań. Odpowiedzi do poszczególnych zadań zapisz w pliku **wyniki3.txt**. Każdą odpowiedź poprzedź numerem oznaczającym zadanie.

Do Twojej dyspozycji jest plik **polaczenia_przyklad.txt** w identycznym formacie. Odpowiedzi dla danych z tego pliku podano pod treściami zadań – możesz na nich sprawdzać działanie swojego programu.

Uwaga: Pamiętaj, że Twój program musi ostatecznie działać dla pliku **polaczenia.txt**.

Zadanie 3.1. (0–2)

3.1.
0–1–2

Podaj numer miasta, z którego wychodzi najwięcej dróg, oraz liczbę tych dróg. Jest jedno takie miasto.

Postać odpowiedzi: Dwie liczby w jednym wierszu: numer miasta i liczba dróg.

Dla pliku **polaczenia_przyklad.txt** poprawna odpowiedź to:

```
8 4
```

[kod zadania: inf-4-3a]

Zadanie 3.2. (0–4)

3.2.
0–1–2–3–4

Podaj liczbę obszarów oraz liczbę miast w największym obszarze.

Postać odpowiedzi: Dwie liczby w jednym wierszu: liczba obszarów i liczba miast w największym obszarze.

Dla pliku **polaczenia_przyklad.txt** poprawna odpowiedź to:

```
4 6
```

[kod zadania: inf-4-3b]

Zadanie 3.3. (0–4)

3.3.
0–1–2–3–4

Odległością miasta *x* od miasta 1 nazywamy najmniejszą liczbę dróg, którymi trzeba przejechać, aby dotrzeć z miasta 1 do miasta *x*. Rozważ tylko miasta, do których można dojechać z miasta 1. Podaj największą odległość od miasta 1 oraz liczbę miast, które są w tej odległości od miasta 1.

Postać odpowiedzi: Dwie liczby w jednym wierszu: największa odległość i liczba miast.

Dla pliku **polaczenia_przyklad.txt** poprawna odpowiedź to:

```
2 3
```

[kod zadania: inf-4-3c]

Do oceny oddajesz:

- plik tekstowy **wyniki3.txt** zawierający odpowiedzi do zadań 3.1.–3.3. (odpowiedź do każdego zadania powinna być poprzedzona jego numerem)
- pliki zawierające kody źródłowe Twoich programów o nazwach: **zad3_1** (zadanie 3.1.), **zad3_2** (zadanie 3.2.), **zad3_3** (zadanie 3.3.) – z rozszerzeniem zgodnym z językiem programowania (.py, .cpp lub .java); jeżeli jeden program rozwiązuje kilka zadań, zapisz go pod nazwą zawierającą numery tych zadań, np. **zad3_1_2** (uwaga: brak tych plików jest równoznaczny z brakiem rozwiązania zadania).

PIERWSZA ŁAWKA

Zadanie 4. Łańcuchy DNA

W pliku **dna.txt** zapisano 300 par łańcuchów DNA, każdą parę w osobnym wierszu; łańcuchy w parze są oddzielone spacją. Łańcuch jest napisem złożonym z liter A, C, G, T, o długości od 1 do 50. Wiersze numerujemy od 1.

Pierwsze wiersze pliku dna.txt:

```
CTCGCGGCGAAAGCGGCCCAAGTGATAGAG TTATCAACGAGATCCTATAAGACCCCAGTC
TGACCTGGATGAGGGGA GCGTGAGGCTCATCAGGG
TCCTCGTGCG CGAGTATTCT
GACCCAATA TCCTCTCGTGTCGTACTG
TATGAGATGTAGGGTTACG TGTCCCACGGAGGCGGTGACG
```

Napisz program (lub kilka programów), który(-e) znajdzie(-ą) i da(dzą) odpowiedzi do podanych zadań. Odpowiedzi do poszczególnych zadań zapisz w pliku **wyniki4.txt**. Każdą odpowiedź poprzedź numerem oznaczającym zadanie.

Do Twojej dyspozycji jest plik **dna_przyklad.txt** w identycznym formacie. Odpowiedzi dla danych z tego pliku podano pod treściami zadań – możesz na nich sprawdzać działanie swojego programu.

Uwaga: Pamiętaj, że Twój program musi ostatecznie działać dla pliku **dna.txt**.

Zadanie 4.1. (0–2)

4.1.
0–1–2

Podaj liczbę wierszy, w których pierwszy łańcuch występuje w drugim jako spójny fragment (np. ACG występuje w TTACGA; łańcuch jest też fragmentem samego siebie).

Postać odpowiedzi: Jedna liczba całkowita.

Dla pliku **dna_przyklad.txt** poprawna odpowiedź to:

```
1
```

[kod zadania: inf-4-4a]

Zadanie 4.2. (0–3)

4.2.
0–1–2–3

Łańcuch komplementarny odwrócony powstaje z danego łańcucha przez zapisanie go od końca i zastąpienie każdej litery literą komplementarną: A ↔ T, C ↔ G (np. dla AACG jest to CGTT). Podaj liczbę wierszy, w których drugi łańcuch jest łańcuchem komplementarnym odwróconym pierwszego łańcucha.

Postać odpowiedzi: Jedna liczba całkowita.

Dla pliku **dna_przyklad.txt** poprawna odpowiedź to:

```
1
```

[kod zadania: inf-4-4b]

Zadanie 4.3. (0–4)

4.3.
0–1–2–3–4

Podciąg łańcucha powstaje przez usunięcie z niego dowolnych liter (być może żadnej) bez zmiany kolejności pozostałych liter, np. ACT jest podciągiem łańcucha AGCGT. Dla każdego wiersza wyznacz długość najdłuższego wspólnego podciągu obu łańcuchów. Podaj największą z tych długości oraz numer wiersza, w którym ją osiągnięto. Jest jeden taki wiersz.

Postać odpowiedzi: Dwie liczby w jednym wierszu: długość i numer wiersza.

Przykład: dla łańcuchów ACGTTA i CAGTA najdłuższy wspólny podciąg ma długość 4 (np. AGTA).

Dla pliku **dna_przyklad.txt** poprawna odpowiedź to:

```
8 6
```

[kod zadania: inf-4-4c]

Do oceny oddajesz:

- plik tekstowy **wyniki4.txt** zawierający odpowiedzi do zadań 4.1.–4.3. (odpowiedź do każdego zadania powinna być poprzedzona jego numerem)
- pliki zawierające kody źródłowe Twoich programów o nazwach: **zad4_1** (zadanie 4.1.), **zad4_2** (zadanie 4.2.), **zad4_3** (zadanie 4.3.) – z rozszerzeniem zgodnym z językiem programowania (.py, .cpp lub .java); jeżeli jeden program rozwiązuje kilka zadań, zapisz go pod nazwą zawierającą numery tych zadań, np. **zad4_1_2** (uwaga: brak tych plików jest równoznaczny z brakiem rozwiązania zadania).

PIERWSZA ŁAWKA

Zadanie 7. Klub biegacza

W pliku **treningi.txt** zapisano 1676 treningów członków klubu biegacza z okresu od 1 stycznia do 30 czerwca 2026 r. Każdy biegacz trenuje co najwyżej raz dziennie. Pierwszy wiersz pliku jest wierszem nagłówkowym, a dane w wierszach rozdzielono znakami tabulacji. W każdym wierszu zapisano:

- ***Data*** – data treningu (rrrr-mm-dd)
- ***Biegacz*** – pseudonim biegacza
- ***Dystans*** – dystans w km (z dwoma miejscami po przecinku)
- ***Czas*** – czas biegu w formacie g:mm:ss

Przykład:

Data	Biegacz	Dystans	Czas
2026-01-01	bartek77	10,97	1:04:03
2026-01-01	ela_m	4,74	0:23:13
2026-01-01	hubi	7,86	0:45:29
2026-01-01	iga_s	5,12	0:27:31

Z wykorzystaniem danych zawartych w pliku oraz dostępnych narzędzi informatycznych wykonaj podane zadania. Odpowiedzi zapisz w pliku **wyniki7.txt**. Odpowiedź do każdego zadania poprzedź numerem tego zadania.

Zadanie 7.1. (0–2)

7.1.
0–1–2

Podaj pseudonimy trzech biegaczy, którzy w całym okresie przebiegli łącznie najdłuższy dystans (od największego), wraz z tymi dystansami.

Zadanie 7.2. (0–3)

7.2.
0–1–2–3

Wykonaj zestawienie liczby treningów o dystansie (w km) z przedziałów: [0; 5), [5; 10), [10; 15), [15; 21,1), 21,1 i więcej. Na podstawie zestawienia utwórz wykres kolumnowy. Pamiętaj o czytelnym opisie wykresu: tytuł, opisy osi, przedziały na osi X.

Zadanie 7.3. (0–2)

7.3.
0–1–2

Tempo biegu to czas biegu podzielony przez dystans (w sekundach na kilometr). Spośród treningów o dystansie co najmniej 10 km wybierz trening o najlepszym (najmniejszym) tempie. Podaj pseudonim biegacza, datę treningu oraz tempo w sekundach na kilometr zaokrąglone do liczby całkowitej. *Przykład:* bieg na dystansie 10,50 km w czasie 0:52:30 (3150 s) ma tempo 300 s/km.

Zadanie 7.4. (0–3)

7.4.
0–1–2–3

Serią nazywamy ciąg kolejnych dni kalendarzowych, w każdym z których biegacz odbył trening. Podaj pseudonim biegacza, który miał najdłuższą serię, długość tej serii oraz datę jej pierwszego dnia. Jest jeden taki biegacz. *Przykład:* biegacz, który trenował 3, 4, 5 i 7 marca, ma serię 3-dniową (od 3 marca) i serię 1-dniową (7 marca).

Zadanie 8. Konkurs programistyczny

W czterech plikach tekstowych: **szkoly.txt**, **uczestnicy.txt**, **zadania.txt** i **wyniki.txt** zapisano dane z dwuetapowego konkursu programistycznego dla uczniów szkół ponadpodstawowych (dane są fikcyjne). Pierwszy wiersz każdego pliku jest wierszem nagłówkowym, a dane w wierszach rozdzielono znakami tabulacji.

Plik **szkoly.txt** zawiera 22 wiersze danych. W każdym wierszu zapisano:

- ***IdSzkoly*** – identyfikator szkoły
- ***Nazwa*** – nazwa szkoły
- ***Miasto*** – miejscowość

Przykład (pierwsze wiersze pliku):

IdSzkoly	Nazwa	Miasto
1	LO nr 1	Inowroclaw
2	LO nr 2	Torun
3	LO nr 3	Brodnica

Plik **uczestnicy.txt** zawiera 180 wierszy danych. W każdym wierszu zapisano:

- ***IdUczestnika*** – identyfikator uczestnika
- ***Imie*** – imię
- ***Nazwisko*** – nazwisko
- ***IdSzkoly*** – identyfikator szkoły
- ***Klasa*** – klasa (1–4)

Przykład (pierwsze wiersze pliku):

IdUczestnika	Imie	Nazwisko	IdSzkoly	Klasa
1	Zofia	Duda	5	2
2	Mateusz	Jakubowski	12	3
3	Wiktoria	Lewandowska	17	1

Plik **zadania.txt** zawiera 10 wierszy danych. W każdym wierszu zapisano:

- ***IdZadania*** – identyfikator zadania
- ***Nazwa*** – nazwa zadania
- ***Etap*** – etap konkursu (1 albo 2)
- ***MaksPunkty*** – maksymalna liczba punktów

Przykład (pierwsze wiersze pliku):

IdZadania	Nazwa	Etap	MaksPunkty
1	Mrowki	1	100
2	Kafelki	1	100
3	Plecak	1	100

Plik **wyniki.txt** zawiera 1019 wierszy danych. W każdym wierszu zapisano:

- ***IdUczestnika*** – identyfikator uczestnika
- ***IdZadania*** – identyfikator zadania
- ***Punkty*** – liczba punktów uzyskanych za rozwiązanie (0–100)

Przykład (pierwsze wiersze pliku):

IdUczestnika	IdZadania	Punkty
1	1	16
1	2	19
1	3	4

Plik wyniki.txt zawiera tylko przesłane rozwiązania – jeśli uczestnik nie przesłał rozwiązania zadania, to w pliku nie ma odpowiedniego wiersza. Wynikiem uczestnika jest suma punktów za wszystkie przesłane rozwiązania.

Z wykorzystaniem danych zawartych w plikach oraz dostępnych narzędzi informatycznych podaj odpowiedzi do zadań 8.1.–8.5. Odpowiedzi zapisz w pliku **wyniki8.txt**, a każdą z nich poprzedź numerem odpowiedniego zadania.

Zadanie 8.1. (0–2) 8.1. 0–1–2

Podaj imię i nazwisko uczestnika z najwyższym wynikiem oraz ten wynik. Jest jeden taki uczestnik.

Zadanie 8.2. (0–2) 8.2. 0–1–2

Dla każdej miejscowości podaj liczbę uczestników ze szkół z tej miejscowości. Zestawienie uporządkuj malejąco według liczby uczestników, a przy równej liczbie – alfabetycznie według nazwy miejscowości.

Zadanie 8.3. (0–2) 8.3. 0–1–2

Podaj nazwy zadań, za które średnia liczba punktów (liczona z przesłanych rozwiązań) była mniejsza niż 30, wraz z tymi średnimi zaokrąglonymi do 0,01. Zestawienie uporządkuj alfabetycznie według nazw zadań.

Zadanie 8.4. (0–2) 8.4. 0–1–2

Podaj liczbę uczestników, którzy nie przesłali rozwiązania żadnego zadania z etapu 2.

Zadanie 8.5. (0–2) 8.5. 0–1–2

Jury unieważniło zadanie **Kafelki**. Zapisz w języku SQL zapytanie, które usunie z tabeli z danymi z pliku wyniki.txt wszystkie wyniki tego zadania, a następnie podaj imię i nazwisko uczestnika z najwyższym wynikiem po tej zmianie oraz ten wynik.

Do oceny oddajesz:

- plik tekstowy **wyniki8.txt** zawierający odpowiedzi do zadań 8.1.–8.5. (odpowiedź do każdego zadania poprzedzona jego numerem)
- plik **zad8.sql** z treścią zapytań SQL (każde poprzedzone komentarzem z numerem zadania, np. **-- 8.1**) oraz wyeksportowana cała baza **baza8.sql** (MySQL/MariaDB) albo plik bazy **zad8** (.accdb/.odb) z kwerendami zad8_1 … zad8_5
- (uwaga: brak tych plików jest równoznaczny z brakiem rozwiązania zadania)

CO JESZCZE JEST W ŚRODKU

Matura 2027 z informatyki – poziom rozszerzony. Arkusz próbny 4 z kluczem

Podgląd pokazał 10 z 31 stron. Pełna wersja: 31 stron A4, PDF do druku i DOCX do edycji.

- Arkusz próbny: Matura 2027 z informatyki – poziom rozszerzony, zestaw nr 4.
- Poziom rozszerzony · Formuła 2023 · arkusz próbny na maturę 2027 · metoda połowienia, jednoczesne min i max, sieć dróg (graf), łańcuchy DNA, ułamki dwójkowe, adresacja IP, klub bie
- Czas pracy: 210 minut · 8 zadań · 50 pkt · 31 stron A4.
- W pakiecie plików: arkusz w PDF (do druku), ten sam arkusz w DOCX (do edycji w Wordzie) oraz archiwum DANE z plikami .txt do zadań przy komputerze (z plikami *_przyklad.txt do spra
- Na końcu: zasady oceniania, klucz odpowiedzi i przykładowe realizacje zadań otwartych.
- Arkusz jest zbudowany jak matura CKE 2023–2026: 8 zadań, około 20% punktów bez komputera, odpowiedzi zapisywane w plikach wynikiN.txt, wiązka arkusza kalkulacyjnego z wykresem i wi
- Programy z zadań 3 i 4 możesz sprawdzić automatycznie na pierwszalawka.pl/kod-na-maturze – przy każdym poleceniu jest kod zadania.
- „Kod na maturze" uruchamia Twój program w Pythonie, C++ albo Javie na danych z arkusza i na dwóch dodatkowych zestawach.
- To nie jest arkusz CKE ani kopia arkusza z poprzednich lat – wszystkie teksty i zadania są oryginalne, przygotowane zgodnie z wymaganiami egzaminacyjnymi na rok 2027.
- Dzięki temu uczeń nie zna zadań z internetu, a wynik rzetelnie pokazuje stan przygotowania.

Kup pełną wersję – plik od razu na e-mail

pierwszalawka.pl/material/matura-informatyka-pr-2027-arkusz-probny-4