INFORMATYKA

Poziom rozszerzony

Egzamin maturalny – Formuła 2023 · arkusz próbny nr 2

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-2-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-2-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.

PIERWSZA ŁAWKA

Zadanie 1. Napisy nawiasowe

Dana jest procedura rekurencyjna P, której argumentem jest nieujemna liczba całkowita *n*. Procedura wypisuje znaki jeden za drugim, bez odstępów; *div* oznacza dzielenie całkowite.

```
procedura P(n)
  jeżeli n = 0, to
    wypisz „x”
  w przeciwnym razie
    wypisz „(”
    P(n - 1)
    P(n div 2)
    wypisz „)”
```

Na przykład wywołanie P(1) wypisuje napis **(xx)**, a procedura P jest przy tym wywoływana 3 razy (P(1), P(0), P(0)).

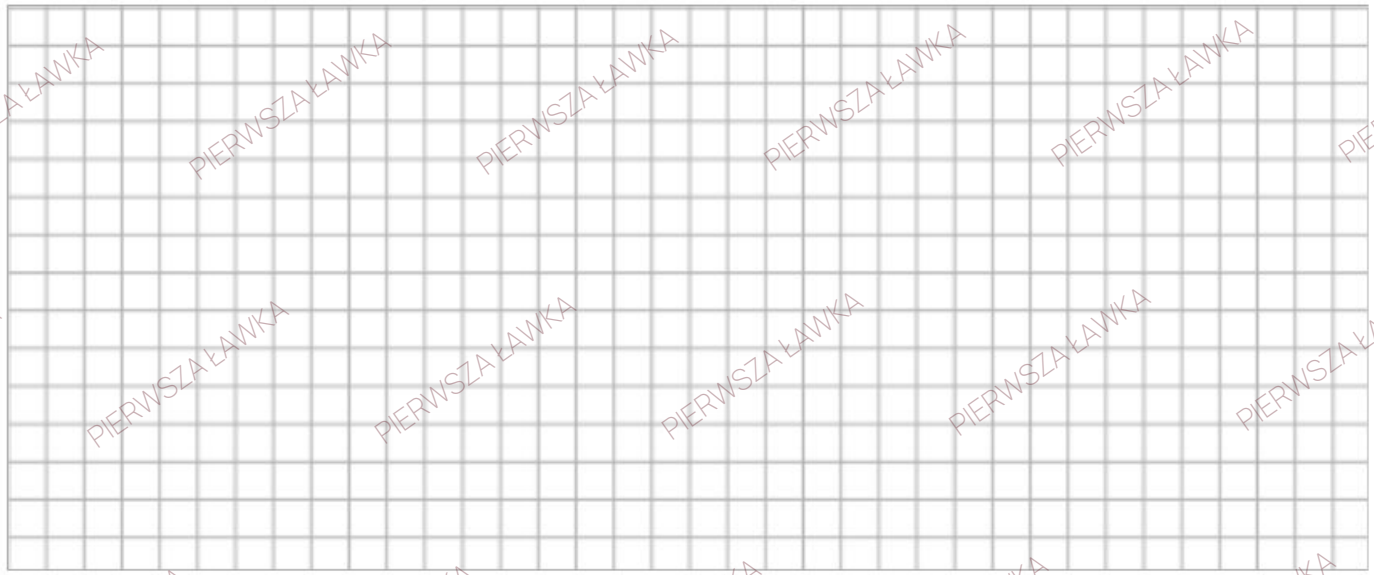
Zadanie 1.1. (0–2)

1.1.
0–1–2

Uzupełnij tabelę. Podaj napis wypisany przez wywołanie P(*n*) oraz liczbę wywołań procedury P (łącznie z pierwszym).

n	napis wypisany przez P(*n*)	liczba wywołań procedury P
1	(xx)	3
2		
3		

Miejsce na obliczenia (brudnopis)



Zadanie 2.2. (0–4)

2.2.
0–1–2–3–4

Zapisz w pseudokodzie lub w wybranym języku programowania algorytm, który dla danych n, $A[1..n]$ i x wyznaczy liczbę elementów tablicy A mniejszych od x.

Uwaga: Twój algorytm musi korzystać z metody połowienia (wyszukiwania binarnego): liczba porównań elementów tablicy z x może być co najwyżej rzędu $\log_2 n$ – nie wolno przeglądać tablicy element po elemencie. Zabronione jest używanie funkcji wbudowanych wyszukujących i sortujących.

Specyfikacja:

Dane:

n – dodatnia liczba całkowita

$A[1..n]$ – tablica liczb całkowitych uporządkowanych ściśle rosnąco

x – liczba całkowita

Wynik:

k – liczba elementów tablicy A mniejszych od x ($0 \le k \le n$)

Miejsce na zapis algorytmu

PIERWSZA ŁAWKA

Zadanie 3. Hasła

W pliku **hasla.txt** zapisano 800 haseł użytkowników pewnego serwisu, każde w osobnym wierszu. Hasła składają się z małych i wielkich liter alfabetu łacińskiego (a–z, A–Z), cyfr (0–9) oraz znaków specjalnych ze zbioru **! @ # $ % & *** i nie zawierają odstępów. Wielkość liter ma znaczenie (np. Kot1 i kot1 to dwa różne hasła). W pliku **slownik.txt** zapisano 20 popularnych słów złożonych z małych liter a–z, każde w osobnym wierszu.

Pierwsze wiersze pliku hasla.txt:

```
Kasztan681
kawA75
3DLacq&1BD0goH#g
admin8081
cNJIlVeQ4Tb
```

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 są pliki **hasla_przyklad.txt** i **slownik_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 **hasla.txt** (i **slownik.txt**).

Zadanie 3.1. (0–2)

3.1.
0–1–2

Hasło jest *silne*, jeżeli ma co najmniej 12 znaków i zawiera co najmniej jedną małą literę, co najmniej jedną wielką literę, co najmniej jedną cyfrę i co najmniej jeden znak specjalny. Podaj liczbę silnych haseł w pliku **hasla.txt** (hasło występujące w pliku kilka razy liczymy tyle razy, ile razy występuje).

Postać odpowiedzi: Jedna liczba całkowita.

Przykład: hasło **Ab3$efgh1234** jest silne, a hasło **abcd$efgh1234** nie jest silne (nie zawiera wielkiej litery).

Dla pliku **hasla_przyklad.txt i slownik_przyklad.txt** poprawna odpowiedź to:

```
9
```

[kod zadania: inf-2-3a]

Zadanie 3.2. (0–4)

3.2.
0–1–2–3–4

Niektóre hasła występują w pliku więcej niż jeden raz. Podaj, ile jest **różnych** haseł, które występują w pliku więcej niż jeden raz, oraz hasło występujące w pliku najwięcej razy wraz z liczbą jego wystąpień. Jest jedno takie hasło.

Postać odpowiedzi: Jeden wiersz: liczba różnych powtarzających się haseł, najczęstsze hasło, liczba jego wystąpień.

Dla pliku **hasla_przyklad.txt i slownik_przyklad.txt** poprawna odpowiedź to:

```
2 Hw#@eQu1CbBtEAe 5
```

[kod zadania: inf-2-3b]

Zadanie 3.3. (0–4)

3.3.
0–1–2–3–4

Hasło jest *łatwe do odgadnięcia*, jeżeli zawiera jako spójny fragment co najmniej jedno słowo z pliku **slownik.txt**, bez względu na wielkość liter (np. hasło 2KoT!x zawiera słowo kot). Podaj liczbę haseł łatwych do odgadnięcia (hasło występujące w pliku kilka razy liczymy tyle razy, ile razy występuje).

Postać odpowiedzi: Jedna liczba całkowita.

Dla pliku **hasla_przyklad.txt i slownik_przyklad.txt** poprawna odpowiedź to:

```
10
```

[kod zadania: inf-2-3c]

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).

Zadanie 4. Obraz

Plik **obraz.txt** zawiera obraz w 256 odcieniach szarości o wymiarach 100 wierszy na 120 kolumn. W każdym wierszu pliku zapisano jasności kolejnych pikseli jednego wiersza obrazu – liczby całkowite od 0 do 255 oddzielone pojedynczymi spacjami. Wiersze i kolumny obrazu numerujemy od 1.

Pierwsze wiersze pliku obraz.txt:

```
34 34 34 34 193 193 193 193 240 240 240 240 33 33 141 39 39 39 39 39 39 91 91 150 32 32 32 32 1
112 147 147 147 223 126 126 126 126 126 126 68 68 68 68 54 54 54 54 54 54 115 115 112 112 164 1
1 1 1 1 254 246 246 21 42 42 74 74 74 76 231 231 82 47 47 47 47 47 47 172 172 172 172 172 172 2
237 237 237 237 237 237 69 103 103 22 22 22 22 107 167 167 123 123 123 123 136 136 136 136 137
157 157 157 154 154 154 221 221 221 155 155 155 118 118 118 129 64 64 4 183 183 132 132 132 13
```

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 **obraz_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 **obraz.txt**.

Zadanie 4.1. (0–2)

4.1.
0–1–2

Podaj numer kolumny obrazu, w której suma jasności pikseli jest największa, oraz tę sumę. Jest jedna taka kolumna.

Postać odpowiedzi: Dwie liczby w jednym wierszu: numer kolumny i suma jasności.

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

```
9 1375
```

[kod zadania: inf-2-4a]

Zadanie 4.2. (0–4)

4.2.
0–1–2–3–4

Obraz kompresujemy bezstratnie metodą RLE, osobno dla każdego wiersza: każdy maksymalny ciąg kolejnych pikseli wiersza o tej samej jasności zapisujemy jako jedną parę (jasność, długość ciągu). Na przykład wiersz 7 7 7 0 0 7 zapisujemy za pomocą trzech par: (7, 3) (0, 2) (7, 1). Podaj łączną liczbę par w zapisie całego obrazu oraz numer wiersza, którego zapis wymaga najmniejszej liczby par (jeżeli jest kilka takich wierszy – podaj najmniejszy numer).

Postać odpowiedzi: Dwie liczby w jednym wierszu: łączna liczba par i numer wiersza.

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

```
38 1
```

[kod zadania: inf-2-4b]

Zadanie 4.3. (0–4)

4.3.
0–1–2–3–4

Znajdź największy kwadratowy fragment obrazu, w którym wszystkie piksele mają tę samą jasność. Podaj długość boku tego kwadratu (w pikselach), jasność jego pikseli oraz numer wiersza i numer kolumny jego lewego górnego piksela. Jest jeden taki kwadrat.

Postać odpowiedzi: Cztery liczby w jednym wierszu: bok, jasność, numer wiersza, numer kolumny.

Przykład: w obrazie o wymiarach 4 × 5 przedstawionym poniżej największy jednolity kwadrat ma bok 3 i jasność 7, a jego lewy górny piksel leży w wierszu 1 i kolumnie 3. (Zapis RLE tego obrazu składa się z 2 + 2 + 3 + 3 = 10 par).

5	5	7	7	7
5	5	7	7	7
1	5	7	7	7
1	1	2	2	0

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

```
4 201 2 9
```

[kod zadania: inf-2-4c]

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).

Zadanie 7. Fotowoltaika

Rodzina Nowaków ma na dachu domu instalację fotowoltaiczną. W pliku **energia.txt** zapisano dane z każdego dnia 2025 roku. Pierwszy wiersz pliku jest wierszem nagłówkowym, a dane w wierszach rozdzielono znakami tabulacji. W każdym wierszu zapisano:

- ***Data*** – data (rrrr-mm-dd)
- ***Produkcja*** – energia wyprodukowana przez instalację w tym dniu, w kWh (z jednym miejscem po przecinku)
- ***Zuzycie*** – energia zużyta w tym dniu przez dom, w kWh (z jednym miejscem po przecinku)

Przykład:

Data	Produkcja	Zuzycie
2025-01-01	1,5	21,0
2025-01-02	1,7	20,0
2025-01-03	1,5	19,8
2025-01-04	1,2	22,4

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. (0–3)

7.1.
0–1–2–3

Wykonaj zestawienie łącznej produkcji i łącznego zużycia energii w poszczególnych miesiącach. Na podstawie zestawienia utwórz wykres kolumnowy porównujący w każdym miesiącu produkcję i zużycie. Pamiętaj o czytelnym opisie wykresu: tytuł, opisy osi, nazwy miesięcy na osi X oraz legenda.

Zadanie 7.2. (0–2)

7.2.
0–1–2

Podaj liczbę dni, w których produkcja energii była większa od jej zużycia, oraz długość najdłuższego ciągu kolejnych takich dni i datę pierwszego dnia tego ciągu. Jest jeden taki najdłuższy ciąg.

Zadanie 7.3. (0–2)

7.3.
0–1–2

Dla każdego miesiąca oblicz stosunek łącznej produkcji do łącznego zużycia energii w tym miesiącu. Podaj miesiąc, w którym ten stosunek był największy, oraz jego wartość zaokrągloną do 0,01.

Zadanie 7.4. (0–3)

7.4.
0–1–2–3

Nowakowie rozważają zakup akumulatora o pojemności 10 kWh. Przyjmij, że na początku dnia 1 stycznia 2025 r. akumulator jest pusty, a w każdym dniu:

- jeżeli produkcja jest większa od zużycia lub mu równa, nadwyżka (produkcja − zużycie) ładuje akumulator, ale tylko do pełnej pojemności 10 kWh – pozostała część nadwyżki jest oddawana do sieci,

- jeżeli zużycie jest większe od produkcji, brakującą energię dom pobiera najpierw z akumulatora (dopóki akumulator nie jest pusty), a pozostałą część – z sieci.

Podaj łączną ilość energii pobranej z sieci w 2025 roku (w kWh, z jednym miejscem po przecinku) oraz liczbę dni, na koniec których akumulator był całkowicie naładowany (10 kWh).

Przykład: jeżeli na początku dnia w akumulatorze jest 8,5 kWh, a produkcja wynosi 12,0 kWh przy zużyciu 9,0 kWh, to akumulator naładuje się do 10 kWh, a 1,5 kWh zostanie oddane do sieci. Jeżeli następnego dnia produkcja wyniesie 2,0 kWh przy zużyciu 14,5 kWh, to z akumulatora zostanie pobrane 10 kWh, a z sieci – 2,5 kWh.

Do oceny oddajesz:

- plik tekstowy **wyniki7.txt** zawierający odpowiedzi do zadań 7.1.–7.4. (odpowiedź do każdego zadania poprzedzona jego numerem)
- plik zawierający wykres do zadania 7.1. o nazwie **wykres7_1** (z rozszerzeniem .xlsx albo .ods)
- plik z komputerową realizacją rozwiązań zadań 7.1.–7.4. o nazwie **zad7** (.xlsx albo .ods); jeżeli zadanie 7.2. lub 7.4. rozwiązujesz programem – także plik(-i) z kodem źródłowym o nazwie **zad7_2** lub **zad7_4** (z rozszerzeniem zgodnym z językiem programowania) (uwaga: brak tych plików jest równoznaczny z brakiem rozwiązania zadania)

CO JESZCZE JEST W ŚRODKU

Matura 2027 z informatyki – poziom rozszerzony. Arkusz próbny 2 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 2.
- Poziom rozszerzony · Formuła 2023 · arkusz próbny na maturę 2027 · procedura rekurencyjna, wyszukiwanie binarne, hasła, obraz i kompresja RLE, kod U2, grafika i kompresja, fotowolt
- 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-2