Pierwsza Ławka · zestaw działowy · liceum i technikum

Dział 3

Od problemu do programu – Python

Etapy rozwiązywania problemu, środowisko programistyczne, wejście i wyjście, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, pętle, funkcje i testowanie programów w Pythonie. Osiem kart: teoria z przykładami, ćwiczenia, utrwalenie w formule CKE, zadanie przy komputerze z plikami danych (ZIP), kartkówka i sprawdzian w grupach A i B – z kluczem i zasadami oceniania.



Informatyka | klasa 2 LO i technikum | Dział 3 | zakres podstawowy (ZP)

	Karta	Rodzaj	Poziom	Zadania
1	**Od problemu do programu – Python – najważniejsze wiadomości** Teoria, przykłady krok po kroku, rozgrzewka	Najważniejsze wiadomości	ZP	4 zad. · 11 pkt
2	**Od problemu do programu – Python – ćwiczenia podstawowe** Ćwiczę umiejętności z działu	Ćwiczenia	ZP	7 zad. · 18 pkt
3	**Od problemu do programu – Python – utrwalenie** Zadania w formule CKE: polecenia i punktacja jak w arkuszach maturalnych	Zadania typu maturalnego	ZP	6 zad. · 10 pkt
4	**Od problemu do programu – Python – w praktyce, przy komputerze** Pliki danych w archiwum ZIP · wyniki w plikach wynikiN.txt	Ćwiczenia	ZP	1 zad. · 6 pkt
5	**Kartkówka: wyrażenia, pętle i warunki – grupa A** 15 minut · 8 pkt · bez komputera	Kartkówka	ZP	4 zad. · 8 pkt 15 min
6	**Kartkówka: wyrażenia, pętle i warunki – grupa B** 15 minut · 8 pkt · bez komputera	Kartkówka	ZP	4 zad. · 8 pkt 15 min
7	**Sprawdzian: Od problemu do programu – Python – grupa A** 45 minut · 20 pkt · bez komputera	Sprawdzian	ZP	8 zad. · 20 pkt 45 min
8	**Sprawdzian: Od problemu do programu – Python – grupa B** 45 minut · 20 pkt · bez komputera	Sprawdzian	ZP	8 zad. · 20 pkt 45 min

Jak korzystać z zestawu

- Karta 1 przypomina najważniejsze wiadomości i pokazuje przykłady rozwiązane krok po kroku (z kodem w Pythonie albo opisem kroków w aplikacji) – od niej warto zacząć.
- Karta 2 to ćwiczenia podstawowe, karta 3 – utrwalenie w formule zadań maturalnych CKE (ramka „Jak to sprawdza CKE").
- Karta 4 to praca przy komputerze: pliki danych są w archiwum ZIP dołączonym do zestawu, a odpowiedzi kontrolne dla plików *_przyklad (pliki o nazwie z dopiskiem „_przyklad") podano przy zadaniach. Wyniki zapisuje się w plikach wynikiN.txt – jak na maturze.
- Karty 5–6 to kartkówka (15 minut), karty 7–8 – sprawdzian (45 minut), obie w grupach A i B: te same typy zadań i ta sama punktacja, inne dane. Kartkówka i sprawdzian nie wymagają komputera.
- Klucz zawiera odpowiedzi, przykładowe programy, zasady oceniania w stylu CKE i numery wymagań podstawy. Osobny plik „Powtórka" zbiera teorię, typowe błędy i test „Sprawdź się".

Oznaczenia

ZP	zakres podstawowy
ZR	zakres rozszerzony (treści tylko dla ZR)
(0–2 pkt)	liczba punktów możliwych do zdobycia za zadanie
Zadanie 4.1.	wiązka zadań do wspólnego materiału źródłowego
Jak to sprawdza CKE	typ zadania i typowy błąd ze sprawozdań CKE
Na maturze	wymagania z podstawy programowej, których dotyczy temat

Wymagania z podstawy programowej (Dz.U. 2024 poz. 1019)

- **I.1** planuje kolejne kroki rozwiązywania problemu, z uwzględnieniem podstawowych etapów myślenia komputacyjnego (określenie problemu, definicja modeli i pojęć, znalezienie rozwiązania, zaprogramowanie i testowanie rozwiązania);
- **I.3** sprawdza poprawność działania algorytmów dla przykładowych danych.
- **II.1** projektuje i programuje rozwiązania problemów z różnych dziedzin, stosując: instrukcje wejścia/wyjścia, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, instrukcje iteracyjne, funkcje z parametrami i bez parametrów, testuje poprawność programów dla różnych danych, w szczególności programuje algorytmy z punktu I.2;
- **II.2** do realizacji rozwiązań problemów prawidłowo dobiera środowiska informatyczne, aplikacje oraz zasoby, wykorzystuje również elementy robotyki;

Materiał oryginalny – żadne zadanie nie jest kopią zadania CKE ani podręcznika; dane w plikach są fikcyjne. Podstawa programowa LO i technikum w brzmieniu z 2024 r. (Dz.U. 2024 poz. 1019). Wymagania: I.1, I.3, II.1, II.2. Miejsce w programach nauczania: „Informatyka na czasie" (Nowa Era) ZP – część II, rozdz. „Algorytmika i programowanie" (nazwy serii podano tylko informacyjnie). Do użytku własnego i pracy z własnymi uczniami.

Od problemu do programu – Python – najważniejsze wiadomości

Teoria, przykłady krok po kroku, rozgrzewka

ZP </> KARTA 1/8

Imię i nazwisko: Klasa: Data: Wynik: .... / 11 pkt

Najważniejsze wiadomości · 11 pkt

Przypomnij sobie

- **Od problemu do programu (myślenie komputacyjne, I.1)**: 1) *określ problem* – co jest danymi, co wynikiem, jakie są ograniczenia; 2) *zbuduj model* – nazwij wielkości (zmienne) i zależności między nimi (wzór, warunek); 3) *znajdź rozwiązanie* – ułóż kolejne kroki algorytmu (lista kroków, schemat, pseudokod); 4) *zaprogramuj i przetestuj* – napisz program i sprawdź go na przykładzie i na danych brzegowych. Specyfikacja (Dane / Wynik) to zapis kroku 1.
- **Środowisko programistyczne (II.2)**: program w Pythonie zapisujemy w pliku `.py` i uruchamiamy interpreterem. IDLE (Integrated Development and Learning Environment) ma okno edytora i okno powłoki (Shell); polecenie Run → Run Module (F5) uruchamia program, a wynik pojawia się w oknie Shell. Na maturze 2027 z Pythonem dostępny jest Python 3.10 lub nowszy z edytorem IDLE albo PyCharm (komunikat CKE z 20.08.2026); wtyczki AI generujące kod muszą być wyłączone. Języki na maturze: Python, C/C++, Java – rozwiązania w każdym z nich są równoprawne.
- **Wejście i wyjście**: `input()` zawsze zwraca **napis** (str) – liczbę trzeba zamienić: `int(input())` albo `float(input())`. `print(a, b)` rozdziela wartości spacją (domyślnie `sep=" "`) i kończy wiersz (`end="\n"`); `print(1, 2, sep="-")` wypisze 1-2. Wynik do pliku: `print(..., file=w)`, gdzie `w = open("wyniki.txt", "w")`.
- **Wyrażenia arytmetyczne**: `/` – dzielenie zawsze dające liczbę rzeczywistą (7 / 2 = 3.5, 6 / 3 = 2.0); `//` – dzielenie całkowite zaokrąglane w dół (7 // 2 = 3, −7 // 2 = −4); `%` – reszta (7 % 2 = 1, −7 % 2 = 1); `**` – potęga, liczona od prawej (2 ** 3 ** 2 = 2 ** 9 = 512, −2 ** 2 = −4). `round(2.5)` daje 2, bo Python zaokrągla „połówki" do liczby parzystej.
- **Wyrażenia logiczne**: porównania `==` `!=` `<` `<=` `>` `>=` dają True albo False; łączymy je przez `and`, `or`, `not`. Kolejność: najpierw arytmetyka, potem porównania, potem `not`, `and`, na końcu `or`. `x and y` nie oblicza y, gdy x jest fałszem (tzw. skrócone obliczanie) – dlatego `b != 0 and a % b == 0` jest bezpieczne dla b = 0.
- **Instrukcja warunkowa**: `if warunek:` … `elif inny:` … `else:` … – wykonuje się **tylko pierwsza** gałąź z prawdziwym warunkiem. Blok wyznacza wcięcie (zwykle 4 spacje) i dwukropek po warunku.
- **Pętle**: `for i in range(a, b, k)` przyjmuje i = a, a + k, a + 2k, … dopóki i < b – **b nie wchodzi** (range(2, 11, 3) to 2, 5, 8; range(10) to 0, …, 9). `while warunek:` powtarza blok, dopóki warunek jest prawdziwy – w bloku coś musi zmieniać warunek, inaczej pętla nigdy się nie skończy.
- **Funkcje**: `def nazwa(parametry):` – funkcja z parametrami dostaje dane w nawiasie, funkcja bez parametrów ma pusty nawias. `return` kończy funkcję i oddaje wynik (można go użyć dalej); `print` tylko wypisuje na ekran. Funkcja bez `return` zwraca None.
- **Testowanie (I.3)**: sprawdzaj program 1) na przykładzie z treści, 2) na danych brzegowych (0, 1, wartość dokładnie na granicy warunku, np. p = 50 przy „co najmniej 50"), 3) na danych „zwykłych". Gdy wynik się nie zgadza – śledź program w tabeli: zapisuj wartości zmiennych po każdym obrocie pętli.

Przykłady z rozwiązaniem

Przykład 1. **Na wycieczkę jedzie n uczniów, autobus ma m miejsc. Ile autobusów trzeba zamówić? Przejdź przez cztery etapy.**

1) Problem: dane n ≥ 0 i m ≥ 1, wynik – najmniejsza liczba autobusów, w których zmieszczą się wszyscy. 2) Model: potrzeba n / m autobusów, zaokrąglone **w górę**. 3) Rozwiązanie: zaokrąglenie w górę bez liczb rzeczywistych to (n + m − 1) // m. 4) Program i testy:

```
def autobusy(n, m):
    # n - liczba uczniów, m - liczba miejsc w autobusie
    return (n + m - 1) // m
```

Testy dla m = 45: n = 0 → 0, n = 1 → 1, n = 45 → 1 (granica), n = 46 → 2, n = 128 → 3. Zwykłe n // m dałoby dla n = 46 wynik 1 – za mało.

Przykład 2. **Prześledź program i podaj, co wypisze.**

```
s = 0
for i in range(1, 8):
    if i % 3 == 1:
        s = s + i * i
print(s)
```

Tabela: i = 1: 1 % 3 = 1 → s = 1; i = 2, 3 – warunek fałszywy; i = 4: 4 % 3 = 1 → s = 1 + 16 = 17; i = 5, 6 – nie; i = 7: 7 % 3 = 1 → s = 17 + 49 = 66. Pętla kończy się po i = 7 (8 nie wchodzi do range(1, 8)). Program wypisze **66**.

Przykład 3. **Kino sprzedaje bilety: dzieci poniżej 7 lat – 0 zł, osoby poniżej 26 lat z legitymacją – 6 zł, pozostali – 12 zł. Napisz funkcję z parametrami i funkcję bez parametrów.**

```
def cena_biletu(wiek, legitymacja):
    if wiek < 7:
        return 0
    elif wiek < 26 and legitymacja:
        return 6
    else:
        return 12

def naglowek():
    print("Kino Pod Lipami - kasa")
```

Kolejność gałęzi ma znaczenie: 5-latek z legitymacją płaci 0, bo pierwszy prawdziwy warunek to wiek < 7. Testy brzegowe: wiek 6 → 0, 7 z legitymacją → 6, 25 → 6, 26 → 12. Funkcja naglowek() nie ma parametrów i niczego nie zwraca – tylko wypisuje tekst.

Zadania

Zadanie 1. (0–4 pkt) ..... / 4

Podaj wartości wyrażeń w Pythonie.

1. 17 // 5 = _____ 2. 17 % 5 = _____ 3. 2 ** 3 * 2 = _____ 4. 9 / 3 = _____

Zadanie 2. (0–1 pkt) ..... / 1

Instrukcja for i in range(2, 11, 3) przyjmie kolejno wartości

A. 3, 6, 9 B. 2, 5, 8, 11 C. 2, 5, 8 D. 2, 4, 6, 8, 10

Od problemu do programu – Python – ćwiczenia podstawowe

Ćwiczę umiejętności z działu

ZP </> KARTA 2/8

Imię i nazwisko: Klasa: Data: Wynik: ..... / 18 pkt

Ćwiczenia · 18 pkt

Zadanie 1. (0–3 pkt) ..... / 3

Prześledź program. Uzupełnij tabelę – wartości x i k po każdym obrocie pętli – i podaj, co wypisze program.

```
x = 50
k = 0
while x > 3:
    x = x // 2
    k = k + 1
print(x, k)
```

obrót	x	k
1		
2		
3		
4		

Miejsce na odpowiedź

Zadanie 2. (0–3 pkt) ..... / 3

Zapisz warunki jako wyrażenia logiczne w Pythonie.

1. n jest liczbą parzystą i dodatnią: _____
2. x leży poza przedziałem [1, 10]: _____
3. rok r jest przestępny (dzieli się przez 4 i nie dzieli się przez 100 albo dzieli się przez 400):

_____

Zadanie 3. (0–4 pkt) ..... / 4

Połącz instrukcję z tym, co wypisze. Jeden wynik jest zbędny.

1. print(7 / 2)
2. print(7 // 2)
3. print(7 % 2)
4. print(2 ** -1)

a) 0.5
b) 1
c) 2.0
d) 3
e) 3.5

1 ____ 2 ____ 3 ____ 4 ____

Zadanie 4. (0–2 pkt) ..... / 2

Program miał wypisać sumę dwóch liczb. Dla danych 12 i 5 wypisał 125. Wyjaśnij przyczynę błędu i popraw program.

```
a = input("Pierwsza liczba: ")
b = input("Druga liczba: ")
print(a + b)
```

Miejsce na odpowiedź

Zadanie 5. (0–3 pkt) ..... / 3

Uzupełnij funkcję maks3(a, b, c), która zwraca największą z trzech liczb.

```
def maks3(a, b, c):
    m = a
    if ……(1)……:
        m = b
    if c > m:
        ……(2)……
    ……(3)……
```

1. (1): _____
2. (2): _____
3. (3): _____

Zadanie 6. (0–2 pkt) ..... / 2

Podaj trzy testy (dane n, m i oczekiwany wynik) dla funkcji autobusy(n, m) z karty 1. Co najmniej jeden test ma sprawdzać przypadek, gdy n jest wielokrotnością m.

Miejsce na odpowiedź

Zadanie 7. (0–1 pkt) ..... / 1

Uczeń chce zdawać maturę 2027 z informatyki, pisząc programy w Pythonie. Według komunikatu CKE z 20.08.2026 może korzystać z

A. tylko Pythona 2.7 i Notatnika

B. Pythona w przeglądarce internetowej

C. dowolnego edytora z wtyczką AI generującą kod

D. Pythona 3.10 lub nowszego z edytorem IDLE albo PyCharm

Od problemu do programu – Python – utrwalenie

Zadania w formule CKE: polecenia i punktacja jak w arkuszach maturalnych

ZP </> KARTA 3/8

Imię i nazwisko: ..... Klasa: ..... Data: ..... Wynik: ..... / 10 pkt

Zadania typu maturalnego · 10 pkt

Jak to sprawdza CKE

W arkuszach maturalnych 2023–2026 pierwsza wiązka zadań to analiza algorytmu rozwiązywana w arkuszu (komputer jest dostępny przez cały egzamin, ale odpowiedź wpisuje się w arkusz): śledzenie kodu lub pseudokodu i uzupełnienie tabeli (2024 zad. 1.1 „Plansza" i 2.1 „Cyfry", po 0–2 pkt; 2026 zad. 1.1 „Rekurencja", 0–3 pkt). Zapis algorytmu ze *Specyfikacją* i *Uwagą* o dozwolonych operacjach – np. 2024 zad. 3.1 (0–3). Sprawozdania CKE 2024–2026: do najtrudniejszych należą zadania z zapisem algorytmu i zadania symulacyjne; zdający nie przestrzegają uwag o dozwolonych operacjach, a CKE zaleca sprawdzanie rozwiązania na danych kontrolnych. Zadanie P/F – 1 pkt za komplet poprawnych ocen.

Zadanie 1. (0–1 pkt) ..... / 1

Oceń prawdziwość podanych zdań. Zaznacz P, jeśli zdanie jest prawdziwe, albo F – jeśli jest fałszywe.

1. Wyrażenie 5 > 3 or 1 / 0 > 0 ma wartość True, a program nie zgłasza błędu dzielenia przez zero.	P	F
2. Wyrażenie not 2 + 2 == 5 ma wartość False.	P	F
3. Dla każdej liczby całkowitej n ≥ 0 wartość wyrażenia n // 4 * 4 + n % 4 jest równa n.	P	F

Zadanie 2. (0–1 pkt) ..... / 1

Dokończ zdanie. Zaznacz właściwą odpowiedź spośród podanych.

Pętla for i in range(3, 20, 4) wykona się

A. 17 razy B. 5 razy C. 6 razy D. 4 razy

Zadanie 3. (0–2 pkt) ..... / 2

Przeanalizuj funkcję f(n). Uzupełnij tabelę – podaj wynik funkcji dla podanych n.

```
def f(n):
    w = 0
    while n > 0:
        if n % 3 == 0:
            w = w + 1
        n = n - 2
    return w
```

n	f(n)
7	
12	
20	

Zadanie 4. (0–1 pkt) ..... / 1

Dokończ zdanie. Zaznacz właściwą odpowiedź spośród podanych.
Program zadanie1.py otwiera plik danych instrukcją open("rowery.txt") – bez podawania folderu. Aby program uruchomiony w IDLE poleceniem Run → Run Module (F5) odczytał dane, plik rowery.txt powinien znajdować się

A. w tym samym folderze co plik zadanie1.py

B. w folderze, w którym zainstalowano Pythona

C. na pulpicie, niezależnie od położenia programu

D. w dowolnym folderze – Python sam go odszuka

Zadanie 5. (0–3 pkt) ..... / 3

Zapisz w pseudokodzie lub w wybranym języku programowania algorytm, który dla danej liczby n obliczy, ile liczb całkowitych od 1 do n jest podzielnych przez 3 lub przez 5, ale nie przez 15.

Przykład: dla n = 20 wynikiem jest 8.

Uwaga: Twój algorytm może używać tylko zmiennych przechowujących pojedyncze liczby całkowite, operacji arytmetycznych (dodawanie, odejmowanie, mnożenie, dzielenie całkowite i reszta z dzielenia), porównań, operatorów logicznych, instrukcji warunkowych, pętli, przypisań i własnych funkcji. Nie wolno używać funkcji wbudowanych, tablic ani list ani zamieniać liczby na napis.

Specyfikacja
Dane: n – liczba całkowita, n ≥ 1
Wynik: k – liczba liczb z przedziału [1, n] podzielnych przez 3 lub przez 5, ale niepodzielnych przez 15

Miejsce na odpowiedź

PIERWSZA ŁAWKA

Od problemu do programu – Python – w praktyce, przy komputerze

Pliki danych w archiwum ZIP · wyniki w plikach wynikiN.txt

ZP </> KARTA 4/8

Imię i nazwisko: ..... Klasa: ..... Data: ..... Wynik: ..... / 6 pkt

Ćwiczenia · 6 pkt

Zadanie 1. (0–6 pkt)

Licznik rowerów

Opis danych

Na ścieżce rowerowej przy moście stoi licznik, który codziennie zapisuje, ile rowerów przejechało rano (6:00–12:00) i po południu (12:00–20:00). Plik **rowery.txt** zawiera 300 wierszy – po jednym na dzień, w kolejności dni. W każdym wierszu są dwie liczby całkowite z przedziału [0, 400] oddzielone spacją: liczba rowerów rano i liczba rowerów po południu. Pierwsze wiersze pliku:

```
286 126
128 269
1 1
187 80
```

Plik **rowery_przyklad.txt** zawiera 15 wierszy w tym samym formacie. Dni numerujemy od 1.

Do dyspozycji masz plik **rowery.txt** oraz mniejszy plik **rowery_przyklad.txt**. Odpowiedzi dla danych przykładowych podano przy zadaniach – najpierw sprawdź na nich swój program.

Napisz program (lub kilka programów), który(-e) znajdzie(-ą) i da(dzą) odpowiedzi do podanych zadań. Odpowiedzi zapisz w pliku **wyniki1.txt**, a każdą z nich poprzedź numerem odpowiedniego zadania.

Do oceny oddajesz: plik tekstowy **wyniki1.txt** z odpowiedziami do zadań 1.1–1.3 (każda poprzedzona numerem zadania) oraz pliki z kodem źródłowym programów, np. **zadanie1.py**.

Dane fikcyjne, przygotowane do zadania.

Zadanie 1.1. (0–1 pkt) ..... / 1

Podaj, w ilu dniach po południu przejechało więcej rowerów niż rano. *Dla danych z pliku przykładowego odpowiedź: 5.*

Odpowiedź zapisz w pliku wyniki1.txt

Zadanie 1.2. (0–2 pkt) ..... / 2

Podaj numer dnia, w którym łącznie (rano i po południu) przejechało najwięcej rowerów, oraz tę łączną liczbę; obie liczby oddziel spacją. *Dla danych z pliku przykładowego odpowiedź: 11 560.*

Odpowiedź zapisz w pliku wyniki1.txt

Zadanie 1.3. (0–3 pkt) ..... / 3

Podaj długość najdłuższego ciągu kolejnych dni, w których łącznie przejechało co najmniej 400 rowerów. *Dla danych z pliku przykładowego odpowiedź: 3.*

Odpowiedź zapisz w pliku wyniki1.txt

CO JESZCZE JEST W ŚRODKU

Od problemu do programu – Python – kartkówka, sprawdzian i karty pracy z powtórką, informatyka, klasa 2 LO i technikum (zakres podstawowy)

Podgląd pokazał 9 z 28 stron. Pełna wersja: 28 stron A4, PDF do druku i DOCX do edycji.

- Karta 1: Od problemu do programu – Python – najważniejsze wiadomości – 4 zadania, 11 pkt
- Karta 2: Od problemu do programu – Python – ćwiczenia podstawowe – 7 zadań, 18 pkt
- Karta 3: Od problemu do programu – Python – utrwalenie – 6 zadań, 10 pkt
- Karta 4: Od problemu do programu – Python – w praktyce, przy komputerze – 1 zadanie, 6 pkt
- Karta 5: Kartkówka: wyrażenia, pętle i warunki – grupa A – 4 zadania, 8 pkt
- Karta 6: Kartkówka: wyrażenia, pętle i warunki – grupa B – 4 zadania, 8 pkt
- Karta 7: Sprawdzian: Od problemu do programu – Python – grupa A – 8 zadań, 20 pkt
- Karta 8: Sprawdzian: Od problemu do programu – Python – grupa B – 8 zadań, 20 pkt Karta 1 przypomina teorię i pokazuje przykłady rozwiązane krok po kroku, dalej ćwiczenia, utrwalen

Kup pełną wersję – plik od razu na e-mail
pierwszalawka.pl/material/informatyka-liceum-klasa-2-dzial-03-od-problemu-do-programu-python