Pierwsza Ławka · zestaw działowy · liceum i technikum

Dział 1

Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara

Kody znaków, porównywanie tekstów (porządek leksykograficzny), wyszukiwanie wzorca w tekście metodą naiwną, szyfrowanie i odszyfrowywanie metodą Cezara – na papierze i 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 3 LO i technikum | Dział 1 | zakres podstawowy (ZP)

	Karta	Rodzaj	Poziom	Zadania
1	**Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – najważniejsze wiadomości** Teoria, przykłady krok po kroku, rozgrzewka	Najważniejsze wiadomości	ZP	4 zad. · 8 pkt
2	**Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – ćwiczenia podstawowe** Ćwiczę umiejętności z działu	Ćwiczenia	ZP	7 zad. · 16 pkt
3	**Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – utrwalenie** Zadania w formule CKE: polecenia i punktacja jak w arkuszach maturalnych	Zadania typu maturalnego	ZP	6 zad. · 10 pkt
4	**Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – w praktyce, przy komputerze** Pliki danych w archiwum ZIP · wyniki w plikach wynikiN.txt	Ćwiczenia	ZP	1 zad. · 7 pkt
5	**Kartkówka: szyfr Cezara i wyszukiwanie wzorca – grupa A** 15 minut · 8 pkt · bez komputera	Kartkówka	ZP	4 zad. · 8 pkt 15 min
6	**Kartkówka: szyfr Cezara i wyszukiwanie wzorca – grupa B** 15 minut · 8 pkt · bez komputera	Kartkówka	ZP	4 zad. · 8 pkt 15 min
7	**Sprawdzian: Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – grupa A** 45 minut · 20 pkt · bez komputera	Sprawdzian	ZP	8 zad. · 20 pkt 45 min

8 **Sprawdzian: Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – 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.2b** stosuje przy rozwiązywaniu problemów z różnych dziedzin algorytmy poznane w szkole podstawowej oraz algorytmy: b) na tekstach: porównywania tekstów, wyszukiwania wzorca w tekście metodą naiwną, szyfrowania tekstu metodą Cezara,
- **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;

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.2b, I.3, II.1. Miejsce w programach nauczania: „Informatyka na czasie" (Nowa Era) ZP – część III, rozdz. „Algorytmika i programowanie" (nazwy serii podano tylko informacyjnie). Do użytku własnego i pracy z własnymi uczniami.

Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – najważniejsze wiadomości

Teoria, przykłady krok po kroku, rozgrzewka

ZP </> KARTA 1/8

Imię i nazwisko: Klasa: Data: Wynik: ..... / 8 pkt

Najważniejsze wiadomości · 8 pkt

Przypomnij sobie

- **Napis** (ang. *string*) to ciąg znaków. W Pythonie znaki numerujemy od 0: w napisie t = "LAS" mamy t[0] = "L", t[2] = "S", a len(t) = 3. Każdy znak ma **kod**: ord("A") = 65, …, ord("Z") = 90, ord("a") = 97, …, ord("z") = 122, ord("0") = 48, spacja – 32; chr(65) = "A" (funkcja odwrotna).
- **Porównywanie tekstów**: dwa napisy są równe, gdy mają tę samą długość i te same znaki na tych samych pozycjach. Porządek **leksykograficzny** (słownikowy): porównujemy znaki od lewej; decyduje pierwsza różnica – mniejszy jest napis z mniejszym kodem znaku. Jeśli jeden napis jest początkiem drugiego, krótszy jest mniejszy ("KOT" < "KOTEK").
- Uwaga na wielkie litery i cyfry: "Zebra" < "ala", bo 90 < 97, a "10" < "9", bo porównujemy znaki "1" i "9", a nie liczby. Python porównuje napisy właśnie według kodów znaków (Unicode).
- **Wyszukiwanie wzorca metodą naiwną**: wzorzec w (długość m) przykładamy do tekstu t (długość n) kolejno na pozycjach i = 0, 1, …, n − m (razem n − m + 1 pozycji) i porównujemy znak po znaku w[j] z t[i + j]. Przy pierwszej niezgodności przesuwamy wzorzec o jedną pozycję. W najgorszym razie wykonamy około (n − m + 1) · m porównań.
- **Wystąpienia mogą na siebie zachodzić**: wzorzec "ANA" występuje w "ANANAS" na pozycjach 0 i 2. Metoda naiwna znajduje oba, a metoda napisu t.count("ANA") liczy wystąpienia rozłączne i zwraca 1 – czytaj polecenie uważnie.
- **Szyfr Cezara**: każdą literę zastępujemy literą leżącą k pozycji dalej w alfabecie, cyklicznie (po Z wracamy do A). Według Swetoniusza Juliusz Cezar stosował przesunięcie k = 3: A → D, B → E, …, X → A, Y → B, Z → C. W tym zestawie używamy alfabetu łacińskiego (26 liter A–Z, bez polskich znaków); inne znaki (np. spacja) zostają bez zmian.
- **Wzór na kod litery**: nowy = (ord(c) − 65 + k) mod 26 + 65. **Odszyfrowanie** to przesunięcie o −k, czyli to samo co szyfrowanie kluczem 26 − k. Różnych kluczy jest tylko 25 (k = 1, …, 25), więc szyfrogram łatwo złamać, sprawdzając wszystkie klucze (atak siłowy) – szyfr Cezara nie chroni dziś żadnych danych.
- **Testowanie (I.3)**: sprawdzaj wzorzec na początku i na końcu tekstu, wzorzec dłuższy od tekstu (0 pozycji), wystąpienia zachodzące na siebie ("AA" w "AAAA" – 3 razy), klucz 0 i 25, litery X, Y, Z (przejście przez koniec alfabetu).

Przykłady z rozwiązaniem

Przykład 1. **Porównaj napisy "KOTEK" i "KOTARA" tak, jak robi to funkcja porownaj(a, b).**

```
def porownaj(a, b):
    i = 0
    while i < len(a) and i < len(b):
        if a[i] != b[i]:
            if ord(a[i]) < ord(b[i]):
                return -1
            return 1
        i = i + 1
    if len(a) < len(b):
        return -1
    if len(a) > len(b):
        return 1
    return 0
```

Znaki na pozycjach 0, 1, 2 są równe (K, O, T). Na pozycji 3: "E" (69) i "A" (65); 65 < 69, więc "KOTARA" < "KOTEK" – funkcja porownaj("KOTEK", "KOTARA") zwraca 1. Długość napisów nie ma tu znaczenia: decyduje pierwsza różnica.

Przykład 2. **Znajdź metodą naiwną wszystkie wystąpienia wzorca "ABA" w tekście "ABABAAB".**

n = 7, m = 3, więc sprawdzamy pozycje i = 0, …, 4: i = 0: ABA – zgodne; i = 1: BAB – niezgodne już na pierwszym znaku; i = 2: ABA – zgodne; i = 3: BAA – niezgodne; i = 4: AAB – niezgodne na drugim znaku. Wynik: pozycje 0 i 2 (wystąpienia zachodzą na siebie – wspólna litera A na pozycji 2).

```
def szukaj(t, w):
    wynik = []
    for i in range(len(t) - len(w) + 1):
        j = 0
        while j < len(w) and t[i + j] == w[j]:
            j = j + 1
        if j == len(w):
            wynik.append(i)
    return wynik

print(szukaj("ABABAAB", "ABA"))   # [0, 2]
```

Przykład 3. **Zaszyfruj słowo ZEGAR szyfrem Cezara z kluczem k = 3, a potem sprawdź odszyfrowanie.**

Z (25) → (25 + 3) mod 26 = 2 → C; E → H; G → J; A → D; R → U. Szyfrogram: **CHJDU**. Odszyfrowanie: przesunięcie o 26 − 3 = 23 daje znowu ZEGAR.

```
def cezar(s, k):
    wynik = ""
    for c in s:
        if "A" <= c <= "Z":
            c = chr((ord(c) - 65 + k) % 26 + 65)
        wynik = wynik + c
    return wynik

print(cezar("ZEGAR", 3))    # CHJDU
print(cezar("CHJDU", 23))   # ZEGAR
```

Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – ćwiczenia podstawowe

ZP </> KARTA 2/8

Ćwiczę umiejętności z działu

Imię i nazwisko: ..... Klasa: ..... Data: ..... Wynik: ..... / 16 pkt

Ćwiczenia · 16 pkt

Zadanie 1. (0–2 pkt) ..... / 2

Metoda naiwna szuka wzorca w = "KAK" w tekście t = "KAKAKO". Uzupełnij tabelę: wpisz fragment tekstu przyłożony do wzorca na pozycji i oraz to, czy jest zgodny ze wzorcem (tak/nie).

i	fragment t[i..i+2]	zgodny?
0		
1		
2		
3		

Zadanie 2. (0–2 pkt) ..... / 2

Szyfr Cezara.

1. SZKOLA zaszyfrowane kluczem k = 5: _____
2. HFJXFW odszyfrowane (klucz szyfrowania k = 5): _____

Zadanie 3. (0–4 pkt) ..... / 4

Oceń prawdziwość zdań o porównywaniu napisów w Pythonie.

1. "KOT" < "KOTEK"	P	F
2. "Kot" < "kot"	P	F
3. "kra" < "kot"	P	F
4. "10" < "9"	P	F

PIERWSZA ŁAWKA

Zadanie 4. (0–2 pkt) ..... / 2

Uzupełnij luki w funkcji szukaj(t, w), która zwraca listę wszystkich pozycji wystąpień wzorca w w tekście t.

```
def szukaj(t, w):
    wynik = []
    for i in range( luka 1 ):
        j = 0
        while j < len(w) and  luka 2 :
            j = j + 1
        if j == len(w):
            wynik.append(i)
    return wynik
```

1. luka 1: _____
2. luka 2: _____

Zadanie 5. (0–2 pkt) ..... / 2

Uczeń napisał funkcję szyfrującą poniżej. Podaj dane, dla których funkcja daje błędny szyfrogram, wyjaśnij przyczynę i zapisz poprawkę.

```
def cezar(s, k):
    wynik = ""
    for c in s:
        wynik = wynik + chr(ord(c) + k)
    return wynik
```

Miejsce na odpowiedź

Zadanie 6. (0–2 pkt) ..... / 2

Ile porównań znaków wykona metoda naiwna (porównuje znaki od lewej i przerywa przy pierwszej niezgodności) przy szukaniu wzorca "AAB" w tekście "AAAAAB"? Rozpisz pozycje.

Miejsce na odpowiedź

Zadanie 7. (0–2 pkt) ..... / 2

Szyfrogram NRWHN powstał szyfrem Cezara. Wiadomo, że odszyfrowane słowo zaczyna się od litery K. Uzupełnij.

1. klucz szyfrowania k = _____
2. odszyfrowane słowo: _____

Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – utrwalenie

ZP </> KARTA 3/8

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

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

Zadania typu maturalnego · 10 pkt

Jak to sprawdza CKE

Na maturze z informatyki (poziom rozszerzony) algorytmy na tekstach pojawiają się w zadaniach przy komputerze: 2026 zad. 3 „Pary słów" – słowa w pliku, kody ASCII znaków (3.1–3.3: 2 + 3 + 4 pkt); 2025 zad. 2 „Zapis symboliczny" – m.in. palindromy i szukanie układów znaków w napisach (2.1: 2 pkt, 2.2: 4 pkt). Odpowiedzi zapisuje się w pliku wynikiN.txt, każdą poprzedzoną numerem zadania. Przy zapisie algorytmu CKE dodaje *Uwagę*, czego nie wolno używać – sprawozdania CKE 2023–2026: zdający często ją pomijają.

Zadanie 1. (0–1 pkt) ..... / 1

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

1. Wzorzec "ANA" występuje w tekście "BANANA" dwa razy, jeśli liczymy wystąpienia zachodzące na siebie.	P	F
2. Wyrażenie "BANANA".find("NA") w Pythonie ma wartość 2.	P	F
3. Metoda naiwna szukająca wzorca o długości 5 w tekście o długości 20 sprawdza 15 pozycji początkowych.	P	F

Zadanie 2. (0–1 pkt) ..... / 1

Dokończ zdanie. Zaznacz właściwą odpowiedź spośród podanych.

Słowo ZIMA zaszyfrowane szyfrem Cezara z kluczem k = 4 to

A. DMQE B. VEIW C. DMQA D. CLPD

Zadanie 3. (0–1 pkt) ..... / 1

Dokończ zdanie. Zaznacz właściwą odpowiedź spośród podanych.

Funkcja porownaj("MAPA", "MAPKA") z karty 1 zwróci

```
def porownaj(a, b):
    i = 0
    while i < len(a) and i < len(b):
        if a[i] != b[i]:
            if ord(a[i]) < ord(b[i]):
                return -1
            return 1
        i = i + 1
    if len(a) < len(b):
        return -1
    if len(a) > len(b):
        return 1
    return 0
```

A. −1 B. 1 C. 0 D. 4

Zadanie 4. (0–2 pkt) ..... / 2

Przeanalizuj funkcję f(t, w). Uzupełnij tabelę – podaj wynik funkcji dla podanych danych.

```
def f(t, w):
    k = 0
    for i in range(len(t) - len(w) + 1):
        if t[i:i + len(w)] == w:
            k = k + 1
    return k
```

t	w	f(t, w)
BANANA	ANA	
MAMAMIA	MAM	
ABCABC	CA	
KAJAK	KAJAKI	

Zadanie 5. (0–3 pkt) ..... / 3

Zapisz w pseudokodzie lub w wybranym języku programowania algorytm, który obliczy, ile razy wzorzec w występuje w tekście t (liczymy także wystąpienia zachodzące na siebie). Zastosuj metodę naiwną.

Przykład: dla t = BANANA i w = ANA wynikiem jest 2.

Uwaga: Twój algorytm może odczytywać długość napisu i pojedyncze znaki (np. t[i]), porównywać znaki oraz używać zmiennych liczbowych, instrukcji warunkowych, pętli, przypisań i własnych funkcji. Nie wolno używać wycinków napisów ani funkcji i metod wyszukujących lub zliczających (np. find, count, index, operator in dla napisów).

Specyfikacja

Dane: t, w – napisy złożone z wielkich liter A–Z; n – długość t, m – długość w, $1 \le m \le n$

Wynik: k – liczba wystąpień wzorca w w tekście t

Miejsce na odpowiedź

PIERWSZA ŁAWKA

Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – w praktyce, przy komputerze

ZP </> KARTA 4/8

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

Imię i nazwisko: ..... Klasa: ..... Data: ..... Wynik: ..... / 7 pkt

Ćwiczenia · 7 pkt

Zadanie 1. (0–7 pkt)

Wiadomości z obozu

Opis danych

Uczestnicy obozu wymieniali krótkie zaszyfrowane wiadomości. Plik **wiadomosci.txt** zawiera 300 wierszy. W każdym jest klucz k (liczba całkowita, 1 ≤ k ≤ 25) i – po spacji – jedno słowo złożone z wielkich liter A–Z (od 4 do 15 liter), zaszyfrowane szyfrem Cezara z kluczem k (każda litera przesunięta o k pozycji dalej w alfabecie łacińskim, cyklicznie). Pierwsze wiersze pliku:

```
13 FBNANJVAN
21 KVMJGVNVI
21 VIIVIVIJHD
7 HUHWHGF
```

Plik **wiadomosci_przyklad.txt** zawiera 20 wierszy w tym samym formacie. We wszystkich zadaniach chodzi o słowa **po odszyfrowaniu**.

Do dyspozycji masz plik **wiadomosci.txt** oraz mniejszy plik **wiadomosci_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. (0–2 pkt) ..... / 2

Podaj, w ilu odszyfrowanych słowach występuje (co najmniej raz) fragment LAS. *Dla danych z pliku przykładowego odpowiedź: 4.*

Odpowiedź zapisz w pliku wyniki1.txt

Zadanie 1.2. (0–2 pkt) ..... / 2

Podaj odszyfrowane słowo, które jest pierwsze w porządku alfabetycznym (leksykograficznym). *Dla danych z pliku przykładowego odpowiedź: ANAANLAPA.*

Odpowiedź zapisz w pliku wyniki1.txt

Zadanie 1.3. (0–3 pkt) ..... / 3

Podaj łączną liczbę wystąpień wzorca ANA we wszystkich odszyfrowanych słowach. Liczymy także wystąpienia zachodzące na siebie (w słowie ANANA wzorzec ANA występuje 2 razy). *Dla danych z pliku przykładowego odpowiedź: 6.*

Odpowiedź zapisz w pliku wyniki1.txt

CO JESZCZE JEST W ŚRODKU

Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – kartkówka, sprawdzian i karty pracy z powtórką, informatyka, klasa 3 LO i...

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

- Karta 1: Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – najważniejsze wiadomości – 4 zadania, 8 pkt
- Karta 2: Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – ćwiczenia podstawowe – 7 zadań, 16 pkt
- Karta 3: Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – utrwalenie – 6 zadań, 10 pkt
- Karta 4: Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – w praktyce, przy komputerze – 1 zadanie, 7 pkt
- Karta 5: Kartkówka: szyfr Cezara i wyszukiwanie wzorca – grupa A – 4 zadania, 8 pkt
- Karta 6: Kartkówka: szyfr Cezara i wyszukiwanie wzorca – grupa B – 4 zadania, 8 pkt
- Karta 7: Sprawdzian: Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – grupa A – 8 zadań, 20 pkt
- Karta 8: Sprawdzian: Algorytmy na tekstach: porównywanie, wyszukiwanie wzorca, szyfr Cezara – grupa B – 8 zadań, 20 pkt Karta 1 przypomina teorię i pokazuje przykłady rozwiązane kr

Kup pełną wersję – plik od razu na e-mail

pierwszalawka.pl/material/informatyka-liceum-klasa-3-dzial-01-algorytmy-na-tekstach-porownywanie-wyszukiwanie