Pierwsza Ławka · zestaw działowy · liceum i technikum

Dział 2

Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci)

Porządkowanie ciągu liczb metodą bąbelkową i przez wstawianie, liczba porównań i zamian, obliczanie wyrazów ciągów metodą iteracyjną, w tym ciągu Fibonacciego – 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 2 | zakres podstawowy (ZP)

	Karta	Rodzaj	Poziom	Zadania
1	**Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – najważniejsze wiadomości** Teoria, przykłady krok po kroku, rozgrzewka	Najważniejsze wiadomości	ZP	4 zad. · 9 pkt
2	**Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – ćwiczenia podstawowe** Ćwiczę umiejętności z działu	Ćwiczenia	ZP	6 zad. · 14 pkt
3	**Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – utrwalenie** Zadania w formule CKE: polecenia i punktacja jak w arkuszach maturalnych	Zadania typu maturalnego	ZP	6 zad. · 11 pkt
4	**Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – w praktyce, przy komputerze** Pliki danych w archiwum ZIP · wyniki w plikach wynikiN.txt	Ćwiczenia	ZP	1 zad. · 7 pkt
5	**Kartkówka: sortowanie i ciąg Fibonacciego – grupa A** 15 minut · 8 pkt · bez komputera	Kartkówka	ZP	4 zad. · 8 pkt 15 min
6	**Kartkówka: sortowanie i ciąg Fibonacciego – grupa B** 15 minut · 8 pkt · bez komputera	Kartkówka	ZP	4 zad. · 8 pkt 15 min
7	**Sprawdzian: Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – grupa A** 45 minut · 20 pkt · bez komputera	Sprawdzian	ZP	8 zad. · 20 pkt 45 min
8	**Sprawdzian: Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – grupa B** 45 minut · 20 pkt · bez komputera	Sprawdzian	ZP	8 zad. · 20 pkt 45 min

PIERWSZA ŁAWKA

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.2c** stosuje przy rozwiązywaniu problemów z różnych dziedzin algorytmy poznane w szkole podstawowej oraz algorytmy: c) porządkowania ciągu liczb przez wstawianie i metodą bąbelkową,
- **I.2d** stosuje przy rozwiązywaniu problemów z różnych dziedzin algorytmy poznane w szkole podstawowej oraz algorytmy: d) obliczania wartości elementów ciągu metodą iteracyjną w tym wartości elementów ciągu Fibonacciego;
- **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.2c, I.2d, 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.

Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – najważniejsze wiadomości

Teoria, przykłady krok po kroku, rozgrzewka

ZP </> KARTA 1/8

Imię i nazwisko: Klasa: Data: Wynik: ..... / 9 pkt

Najważniejsze wiadomości · 9 pkt

Przypomnij sobie

- **Porządkowanie (sortowanie)** ciągu liczb niemalejąco: a[0] ≤ a[1] ≤ … ≤ a[n − 1]. W Pythonie zamiana dwóch elementów: a[i], a[j] = a[j], a[i].
- **Sortowanie bąbelkowe**: porównujemy sąsiadów a[j] i a[j + 1] i zamieniamy, gdy a[j] > a[j + 1]. Po pierwszym przebiegu największy element jest na końcu, po drugim – dwa największe itd. Wystarczy n − 1 przebiegów; w wersji podstawowej (krótszy o 1 przebieg za każdym razem) liczba porównań to (n − 1) + (n − 2) + … + 1 = n(n − 1)/2.
- **Liczba zamian** w sortowaniu bąbelkowym jest równa liczbie **inwersji** – par pozycji i < j, dla których a[i] > a[j] (każda zamiana sąsiadów usuwa dokładnie jedną inwersję). Ulepszenie: gdy w przebiegu nie było zamiany, ciąg jest już uporządkowany i można zakończyć.
- **Sortowanie przez wstawianie**: lewa część a[0..i − 1] jest już uporządkowana; element x = a[i] wstawiamy w odpowiednie miejsce, przesuwając większe od niego elementy o jedną pozycję w prawo. Dla ciągu uporządkowanego wykonamy tylko n − 1 porównań, dla odwrotnie uporządkowanego – n(n − 1)/2.
- Obie metody przy warunku „>” (a nie „≥”) nie zmieniają kolejności równych elementów – są **stabilne**. Funkcja sorted() w Pythonie też jest stabilna (dokumentacja języka); na lekcji i przy „zapisz algorytm” piszemy sortowanie sami.
- **Ciąg liczony iteracyjnie**: kolejny wyraz obliczamy w pętli z poprzednich, np. $a_1 = 3$, $a_{n+1} = 2 \cdot a_n - 1$ daje 3, 5, 9, 17, 33.
- **Ciąg Fibonacciego**: $F_0 = 0$, $F_1 = 1$, $F_n = F_{n-1} + F_{n-2}$ dla n ≥ 2: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, … Iteracyjnie pamiętamy tylko dwa ostatnie wyrazy: a, b = 0, 1, a potem n razy a, b = b, a + b – wtedy a = F_n. (Wersja rekurencyjna jest bardzo wolna – ciekawostka dla rozszerzenia.)
- **Testowanie (I.3)**: sortowanie sprawdzaj na ciągu pustym, jednoelementowym, uporządkowanym, odwrotnie uporządkowanym i z powtórzeniami; Fibonacciego – dla n = 0, 1, 2 i porównaj z tabelą wyrazów.

Przykłady z rozwiązaniem

Przykład 1. Wykonaj sortowanie bąbelkowe ciągu 5, 2, 4, 1. Ile będzie porównań i zamian?

Przebieg 1: (5, 2) zamiana → 2, 5, 4, 1; (5, 4) zamiana → 2, 4, 5, 1; (5, 1) zamiana → 2, 4, 1, 5.
Przebieg 2: (2, 4) bez zmiany; (4, 1) zamiana → 2, 1, 4, 5. Przebieg 3: (2, 1) zamiana → 1, 2, 4, 5.
Porównań 3 + 2 + 1 = 6, zamian 5 – tyle, ile inwersji: (5,2), (5,4), (5,1), (2,1), (4,1).

```
def babelkowe(a):
    n = len(a)
    for i in range(n - 1):
        for j in range(n - 1 - i):
            if a[j] > a[j + 1]:
                a[j], a[j + 1] = a[j + 1], a[j]
    return a
```

Przykład 2. **Posortuj przez wstawianie ciąg 4, 1, 3, 2. Zapisz stan po każdym wstawieniu.**

i = 1: x = 1, przesuwamy 4 → 1, 4, 3, 2. i = 2: x = 3, przesuwamy 4 → 1, 3, 4, 2. i = 3: x = 2, przesuwamy 4 i 3 → 1, 2, 3, 4. Razem 4 przesunięcia.

```
def wstawianie(a):
    for i in range(1, len(a)):
        x = a[i]
        j = i - 1
        while j >= 0 and a[j] > x:
            a[j + 1] = a[j]
            j = j - 1
        a[j + 1] = x
    return a
```

Przykład 3. **Oblicz iteracyjnie F_{10}.**

krok	1	2	3	4	5	6	7	8	9	10
a	1	1	2	3	5	8	13	21	34	55
b	1	2	3	5	8	13	21	34	55	89

Start: a = 0, b = 1; po 10 krokach a = F_{10} = 55.

```
def fib(n):
    a, b = 0, 1
    for i in range(n):
        a, b = b, a + b
    return a

print(fib(10))   # 55
```

Zadania

Zadanie 1. (0–3 pkt) ..... / 3

Uzupełnij wyrazy ciągu Fibonacciego.

1. F_7 = _____ 2. F_8 = _____ 3. F_{12} = _____

Zadanie 2. (0–1 pkt) ..... / 1

Po pierwszym przebiegu sortowania bąbelkowego ciągu 6, 3, 8, 2 otrzymamy

A. 3, 2, 6, 8 B. 3, 6, 2, 8 C. 3, 6, 8, 2 D. 2, 3, 6, 8

Zadanie 3. (0–3 pkt) ..... / 3

Ciąg: $a_1 = 4$, $a_{n+1} = 3 \cdot a_n - 2$. Uzupełnij.

1. a_2 = _____ 2. a_3 = _____ 3. a_4 = _____

Zadanie 4. (0–2 pkt) ..... / 2

Oceń prawdziwość zdań.

1. W ciągu 3, 1, 2 są dokładnie 2 inwersje.	P	F
2. Sortowanie przez wstawianie ciągu już uporządkowanego nie wykonuje żadnego porównania.	P	F

Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – ćwiczenia podstawowe

ZP </> KARTA 2/8

Ćwiczę umiejętności z działu

Imię i nazwisko: ..... Klasa: ..... Data: ..... Wynik: ..... / 14 pkt

Ćwiczenia · 14 pkt

Zadanie 1. (0–2 pkt) ..... / 2

Wykonaj sortowanie bąbelkowe ciągu 7, 4, 9, 1, 5. Wpisz stan ciągu po każdym przebiegu.

przebieg	stan ciągu
1	
2	
3	
4	

Zadanie 2. (0–2 pkt) ..... / 2

Wykonaj sortowanie przez wstawianie ciągu 8, 3, 6, 2, 7. Wpisz stan ciągu po wstawieniu elementu a[i].

i	stan ciągu
1	
2	
3	
4	

Zadanie 3. (0–3 pkt) ..... / 3

Uzupełnij: liczba porównań i zamian w sortowaniu bąbelkowym (wersja z funkcji babelkowe z karty 1).

1. ciąg 4, 3, 2, 1: porównań _____, zamian _____
2. ciąg 1, 2, 3, 4: porównań _____, zamian _____
3. liczba inwersji w ciągu 2, 5, 1, 4, 3: _____

Podgląd – Pierwsza Ławka

Zadanie 4. (0–2 pkt) ..... / 2

Uzupełnij luki w funkcji wstawianie(a).

```
def wstawianie(a):
    for i in range(1, len(a)):
        x = a[i]
        j = i - 1
        while j >= 0 and  luka 1 :
            a[j + 1] = a[j]
            j = j - 1
        a[ luka 2 ] = x
    return a
```

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

Zadanie 5. (0–2 pkt) ..... / 2

Uczeń napisał funkcję poniżej, która miała zwracać F_n. Podaj, jaką wartość zwraca dla n = 10, wyjaśnij błąd i zapisz poprawkę.

```
def fib(n):
    a, b = 0, 1
    for i in range(n - 1):
        a, b = b, a + b
    return a
```

Miejsce na odpowiedź

Zadanie 6. (0–3 pkt) ..... / 3

Połącz opis z metodą. Jedna metoda jest zbędna.

1. Porównuje i zamienia sąsiednie elementy; największy „wypływa” na koniec.
2. Lewa część jest uporządkowana; kolejny element wkładamy w odpowiednie miejsce.
3. Każdy wyraz jest sumą dwóch poprzednich.

a) sortowanie bąbelkowe

b) sortowanie przez wstawianie

c) ciąg Fibonacciego

d) wyszukiwanie wzorca

1 ____ 2 ____ 3 ____

Zadanie 4. (0–3 pkt) ..... / 3

Przeanalizuj funkcję g(n). Uzupełnij tabelę – podaj wynik funkcji (a) oraz wartość k.

```
def g(n):
    a, b = 1, 2
    k = 1
    while b <= n:
        a, b = b, a + b
        k = k + 1
    return a, k
```

n	zwrócone a	k
10		
30		
100		

Zadanie 5. (0–3 pkt) ..... / 3

Zapisz w pseudokodzie lub w wybranym języku programowania algorytm, który dla danej liczby n obliczy sumę $F_1 + F_2 + \ldots + F_n$ wyrazów ciągu Fibonacciego, gdzie $F_1 = F_2 = 1$, $F_k = F_{k-1} + F_{k-2}$. Zastosuj metodę iteracyjną.

Przykład: dla n = 6 wynikiem jest 20.

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 (dozwolona jest pętla for i in range(…)), tablic ani list ani zamieniać liczby na napis.

Specyfikacja

Dane: n – liczba całkowita, n ≥ 1

Wynik: s – suma n początkowych wyrazów ciągu Fibonacciego

Miejsce na odpowiedź

Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – 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)

Skok w dal

Opis danych

Na szkolnych zawodach każdy z 250 zawodników oddał 8 skoków w dal. Plik **skoki.txt** zawiera 250 wierszy – w każdym 8 liczb całkowitych z przedziału [100, 999] oddzielonych spacjami: długości kolejnych skoków jednego zawodnika w centymetrach. Pierwsze wiersze pliku:

```
182 214 901 796 649 845 903 971
237 282 785 531 407 739 987 144
318 580 830 567 742 117 139 325
```

Plik **skoki_przyklad.txt** zawiera 12 wierszy w tym samym formacie.

Do dyspozycji masz plik **skoki.txt** oraz mniejszy plik **skoki_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, ilu zawodników poprawiało się albo utrzymywało wynik w każdym skoku, tzn. w ilu wierszach liczby są uporządkowane niemalejąco. *Dla danych z pliku przykładowego odpowiedź: 1.*

Odpowiedź zapisz w pliku wyniki1.txt

Zadanie 1.2. (0–3 pkt) ..... / 3

Każdy wiersz porządkujemy niemalejąco sortowaniem bąbelkowym (zamiana sąsiadów, gdy lewy jest większy). Podaj łączną liczbę zamian wykonanych dla wszystkich wierszy. *Dla danych z pliku przykładowego odpowiedź: 158.*

Odpowiedź zapisz w pliku wyniki1.txt

Zadanie 1.3. (0–2 pkt) ..... / 2

Podaj, ile liczb z pliku (łącznie we wszystkich wierszach) jest wyrazami ciągu Fibonacciego. *Dla danych z pliku przykładowego odpowiedź: 8.*

Odpowiedź zapisz w pliku wyniki1.txt

Kartkówka: sortowanie i ciąg Fibonacciego – grupa A

ZP </> KARTA 5/8

15 minut · 8 pkt · bez komputera

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

Kartkówka · 15 minut · 8 pkt

Zadanie 1. (0–2 pkt) ..... / 2

Uzupełnij wyrazy ciągu Fibonacciego ($F_0 = 0$, $F_1 = 1$).

1. F_9 = _____ 2. F_{14} = _____

Zadanie 2. (0–1 pkt) ..... / 1

Dokończ zdanie. Zaznacz właściwą odpowiedź spośród podanych.
Po pierwszym przebiegu sortowania bąbelkowego ciągu 6, 2, 9, 4 otrzymamy

A. 2, 6, 4, 9 B. 2, 4, 6, 9 C. 4, 9, 2, 6 D. 9, 6, 4, 2

Zadanie 3. (0–3 pkt) ..... / 3

Wykonaj sortowanie przez wstawianie ciągu 5, 3, 8, 1. Wpisz stan ciągu po wstawieniu elementu a[i].

i	stan ciągu
1	
2	
3	

Zadanie 4. (0–2 pkt) ..... / 2

Uzupełnij dla ciągu 6, 2, 9, 4 i sortowania bąbelkowego (wersja z karty 1).

1. liczba porównań: _____ 2. liczba zamian: _____

PIERWSZA ŁAWKA

CO JESZCZE JEST W ŚRODKU

Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – kartkówka, sprawdzian i karty pracy z powtórką, informatyka, klasa 3 LO...

Podgląd pokazał 9 z 27 stron. Pełna wersja: 27 stron A4, PDF do druku i DOCX do edycji.

- Karta 1: Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – najważniejsze wiadomości – 4 zadania, 9 pkt
- Karta 2: Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – ćwiczenia podstawowe – 6 zadań, 14 pkt
- Karta 3: Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – utrwalenie – 6 zadań, 11 pkt
- Karta 4: Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – w praktyce, przy komputerze – 1 zadanie, 7 pkt
- Karta 5: Kartkówka: sortowanie i ciąg Fibonacciego – grupa A – 4 zadania, 8 pkt
- Karta 6: Kartkówka: sortowanie i ciąg Fibonacciego – grupa B – 4 zadania, 8 pkt
- Karta 7: Sprawdzian: Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – grupa A – 8 zadań, 20 pkt
- Karta 8: Sprawdzian: Sortowanie (wstawianie, bąbelkowe) i ciągi liczone iteracyjnie (Fibonacci) – grupa B – 8 zadań, 20 pkt Karta 1 przypomina teorię i pokazuje przykłady rozwiązań

Kup pełną wersję – plik od razu na e-mail

pierwszalawka.pl/material/informatyka-liceum-klasa-3-dzial-02-sortowanie-wstawianie-babelkowe-i-ciagi-liczone