

WYPEŁNIA ZDAJĄCY

KOD

--	--	--

PESEL

--	--	--	--	--	--	--	--	--	--	--	--

Miejsce na naklejkę.

Sprawdź, czy kod na naklejce to

M-100.

Jeżeli tak – przyklej naklejkę.

Jeżeli nie – zgłoś to nauczycielowi.

Egzamin maturalny

Formuła 2023

INFORMATYKA

Poziom rozszerzony

WYPEŁNIA ZDAJĄCY

WYBRANE:

.....

(system operacyjny)

.....

(program użytkowy)

.....

(środowisko programistyczne)

Symbol arkusza

MINP-R0-100-2306

DATA: **16 czerwca 2023 r.**

GODZINA ROZPOCZĘCIA: **9:00**

CZAS TRWANIA: **210 minut**

LICZBA PUNKTÓW DO UZYSKANIA: **50**

Przed rozpoczęciem pracy z arkuszem egzaminacyjnym

1. Sprawdź, czy nauczyciel przekazał Ci **właściwy arkusz egzaminacyjny**, tj. arkusz we **właściwej formule**, z **właściwego przedmiotu** na **właściwym poziomie**.
2. Jeżeli przekazano Ci **niewłaściwy** arkusz – natychmiast zgłoś to nauczycielowi. Nie rozrywaj banderol.
3. Jeżeli przekazano Ci **właściwy** arkusz – rozerwij banderole po otrzymaniu takiego polecenia od nauczyciela. Zapoznaj się z instrukcją na stronie 2.





Instrukcja dla zdającego

1. Sprawdź, czy arkusz egzaminacyjny zawiera 22 strony (zadania 1–7) i czy dołączony jest do niego nośnik danych – podpisany DANE. Ewentualny brak zgłoś przewodniczącemu zespołu nadzorującego egzamin.
2. Na pierwszej stronie oraz na karcie odpowiedzi wpisz swój numer PESEL i przyklej naklejkę z kodem.
3. Wpisz zadeklarowane (wybrane) przez Ciebie na egzamin: system operacyjny, program użytkowy oraz środowisko programistyczne.
4. Symbol  zamieszczony w nagłówku zadania zwraca uwagę na to, że zadanie nie wymaga użycia komputera i odpowiedź do zadania należy zapisać tylko w miejscu na to przeznaczonym w arkuszu egzaminacyjnym.
5. Jeśli rozwiązaniem zadania lub jego części jest program komputerowy, to umieść w katalogu (folderze) oznaczonym Twoim numerem PESEL wszystkie utworzone przez siebie pliki w wersji źródłowej.
6. Jeśli rozwiązaniem zadania lub jego części jest baza danych utworzona z wykorzystaniem MySQL (MariaDB), to umieść w katalogu (folderze) oznaczonym Twoim numerem PESEL treści zapytań w języku SQL oraz (przed zakończeniem egzaminu) wyeksportowaną całą bazę w formacie *.sql.
7. Pliki oddawane do oceny nazwij dokładnie tak, jak polecono w treści zadań, lub zapisz je pod nazwami (wraz z rozszerzeniem zgodnym z zadeklarowanym oprogramowaniem), jakie podajesz w arkuszu egzaminacyjnym. **Pliki o innych nazwach nie będą sprawdzane przez egzaminatora.**
8. **Przed upływem czasu przeznaczanego na egzamin** zapisz w katalogu (folderze) oznaczonym Twoim numerem PESEL ostateczną wersję plików stanowiących rozwiązania zadań.
9. Pisz czytelnie. Używaj długopisu/pióra tylko z czarnym tuszem/atramentem.
10. Nie używaj korektora, a błędne zapisy wyraźnie przekreśl.
11. Pamiętaj, że zapisy w brudnopisie nie będą oceniane.



**Zadania egzaminacyjne są wydrukowane
na następnych stronach.**

Zadanie 1. Mnożenie

Następujący rekurencyjny algorytm mnożenia dwóch liczb całkowitych dodatnich x , y jest realizowany z użyciem operacji arytmetycznych dodawania i dzielenia całkowitego przez 2.

iloczyn(x , y):

jeżeli $y = 1$
 wynikiem jest x
w przeciwnym razie
 $k \leftarrow y \text{ div } 2$
 $z \leftarrow \text{iloczyn}(x, k)$
 jeżeli $y \bmod 2 = 0$
 wynikiem jest $z + z$
 w przeciwnym razie
 wynikiem jest $x + z + z$

Uwaga: $x \bmod y$ oznacza resztę z dzielenia x przez y , natomiast $x \text{ div } y$ oznacza wynik dzielenia całkowitego x przez y .

Dla danych liczb x , y interesuje nas **liczba wykonywanych operacji dodawania** podczas obliczania wyniku funkcji *iloczyn*(x , y).

Przykład 1.

Dla liczb $x=9$ i $y=11$ algorytm wykonuje 5 dodawań. Działanie funkcji *iloczyn*(9, 11) można zilustrować w następujący sposób (w nawiasach obok wskazano liczbę wykonywanych operacji dodawania):

iloczyn(9, 11) = $9 + z + z$, (dwa dodawania)
gdzie $z = \text{iloczyn}(9, 5)$

iloczyn(9, 5) = $9 + z + z$, (dwa dodawania)
gdzie $z = \text{iloczyn}(9, 2)$

iloczyn(9, 2) = $z + z$, (jedno dodawanie)
gdzie $z = \text{iloczyn}(9, 1)$

iloczyn(9, 1) = 9

Poniższa tabela ilustruje obliczenia wykonywane podczas wywołania *iloczyn*(9, 11)

Numer wywołania	Parametry wywołania		Obliczone k , z		Wynik
	x	y	k	z	
1	9	11	5	45	99 (9+45+45)
2	9	5	2	18	45 (9+18+18)
3	9	2	1	9	18 (9+9)
4	9	1	–	–	9

Zadanie 1.1. (0–2)

Uzupełnij poniższą tabelę tak, aby ilustrowała obliczenia wykonywane podczas wywołania *iloczyn(10, 45)*.

Numer wywołania	Parametry wywołania		Obliczone k, z		Wynik
	x	y	k	z	
1	10	45	22		
2					
3					
4					
5					
6		1	–	–	

Miejsce na obliczenia:

This image shows a full page of blank graph paper. The grid consists of small, uniform squares formed by thin, light gray lines. There are no margins, text, or other markings on the page.

Zadanie 1.2. (0–2)

Dla liczb x , y wymienionych w poniższej tabeli podaj liczbę operacji dodawania, jaka zostanie wykonana podczas obliczania wyniku funkcji $iloczyn(x, y)$.

x	y	Liczba dodawań
9	11	5
8	32	
2	47	
112	112	

Miejsce na obliczenia



Zadanie 1.3. (0–2)

Poniżej zapisano iteracyjny algorytm realizujący funkcję $iloczyn(x, y)$. Uzupełnij trzy luki w algorytmie, tak aby był zgodny z poniższą specyfikacją.

UWAGA: spośród operacji arytmetycznych możesz użyć tylko: dodawania, odejmowania, dzielenia całkowitego i reszty z dzielenia. Nie możesz użyć zwłaszcza operacji mnożenia.

Specyfikacja:

Dane:

x, y – liczby całkowite dodatnie

Wynik:

z – wartość iloczynu $x*y$

Algorytm:

$z \leftarrow$ _____

dopóki _____ wykonuj:

jeżeli $y \bmod 2 = 1$

$z \leftarrow z + x$

$x \leftarrow x + x$

$y \leftarrow$ _____

Zadanie 2. Sufiksy

Słowo definiujemy jako ciąg złożony z małych liter alfabetu angielskiego.

Niech $s[1..n]$ będzie słowem o długości $n > 0$.

Sufiksem słowa s nazywamy każde jego podśłowo kończące na ostatniej pozycji słowa s .

Sufiks $s[k..n]$ nazywamy k -tym sufiksem.

Przykład 1.

słowo $s[1..10] = \text{mascarpone}$ ma następujące sufiksy:

k	$s[k..n]$
1	mascarpone
2	ascarpone
3	scarpone
4	carpone
5	arpone
6	rpone
7	pone
8	one
9	ne
10	e

Uporządkowanie alfabetyczne wszystkich sufiksów słowa *mascarpone* daje następującą kolejność ich numerów (od najmniejszego): 5, 2, 4, 10, 1, 9, 8, 7, 6, 3:

k	$s[k..n]$
5	arpone
2	ascarpone
4	carpone
10	e
1	mascarpone
9	ne
8	one
7	pone
6	rpone
3	scarpone

Poniżej zapisano funkcję **czy_mniejszy($n, s, k1, k2$)**. Wynikiem funkcji jest wartość PRAWDA, gdy sufiks $s[k1..n]$ jest mniejszy w porządku alfabetycznym od sufiksu $s[k2..n]$ oraz FAŁSZ w przeciwnym przypadku.

Specyfikacja

Dane:

- n – długość słowa,
- $s[1..n]$ – słowo zapisane jako tablica znaków (numerowanych od 1),
- $k1$ – numer pierwszego sufiksu ($1 \leq k1 \leq n$),
- $k2$ – numer drugiego sufiksu ($1 \leq k2 \leq n, k1 \neq k2$).

Wynik:

PRAWDA jeśli sufiks $s[k1..n]$ jest mniejszy w porządku alfabetycznym od $s[k2..n]$, albo FAŁSZ – w przeciwnym wypadku.

czy_mniejszy ($n, s, k1, k2$)

$i \leftarrow k1$

$j \leftarrow k2$

dopóki ($i \leq n$ oraz $j \leq n$) **wykonuj**

jeżeli ($s[i] == s[j]$)

$i \leftarrow i + 1$

$j \leftarrow j + 1$

w przeciwnym razie

jeżeli ($s[i] < s[j]$)

zakończ z wynikiem PRAWDA

w przeciwnym razie

zakończ z wynikiem FAŁSZ

jeżeli ($j \leq n$)

zakończ z wynikiem PRAWDA

w przeciwnym razie

zakończ z wynikiem FAŁSZ

Pierwsze dwie instrukcje *jeżeli* w funkcji `czy_mniejszy` wykonują porównania dwóch znaków słowa `s`.

Przykład:

dla danych $s = mascarpone$, $k1 = 5$, $k2 = 2$ algorytm wykona 3 porównania:

- (pierwsza instrukcja *jeżeli*) – sprawdzenie, czy $s[5] = s[2]$
- (pierwsza instrukcja *jeżeli*) – sprawdzenie, czy $s[6] = s[3]$
- (druga instrukcja *jeżeli*) – sprawdzenie, czy $s[6] < s[3]$

Podaj przykład słowa s , o długości ≤ 10 oraz liczb k_1, k_2 , $k_1 \neq k_2$, dla których funkcja `czy_mniejszy` wykona dokładnie 6 porównań w pierwszej instrukcji *jeżeli*.

$s =$ _____, $k1 =$ _____, $k2 =$ _____

Miejsce na obliczenia:

This image shows a full page of blank graph paper. The grid consists of thin, light gray horizontal and vertical lines that intersect to form small squares across the entire surface. There are no margins, text, or other markings on the paper.

Zadanie 2.2. (0–2)

W plikach `slova1.txt`, `slova2.txt` i `slova3.txt` znajdują się po trzy wiersze:

- w pierwszym wierszu każdego pliku zapisana jest liczba całkowita dodatnia n , oznaczająca długość słowa
- w drugim wierszu zapisane jest n -literowe słowo s , składające się z małych liter alfabetu angielskiego a-z
- w trzecim wierszu zapisane są dwie liczby $k1$ i $k2$, oddzielone spacją.

Napisz program z zaimplementowaną funkcją `czy_mniejszy`. Jako wynik Twój program powinien wypisywać TAK lub NIE, w zależności od wyniku funkcji `czy_mniejszy`. Odpowiedzi dla poszczególnych plików zapisz w pliku `wyniki2_2.txt`.

Dla przykładowego pliku `sufiks_1.txt`, Twój program powinien dać odpowiedź: TAK, a dla przykładowego pliku `sufiks_2.txt` – odpowiedź: NIE.

Do oceny oddajesz:

- plik `wyniki2_2.txt` – zawierający odpowiedzi (odpowiedź dla każdego z plików powinna być poprzedzona jego nazwą)
- plik(i) zawierający(-e) kody źródłowe Twojego programu o nazwie(-ach):

.....

Zadanie 2.3. (0–3)

Dana jest dodatnia liczba całkowita n oraz słowo $s[1..n]$. Naszym celem jest obliczenie wartości elementów tablicy $T[1..n]$ zawierającej numery sufiksów słowa $s[1..n]$ uporządkowanych w porządku alfabetycznym.

Przykład:

dla słowa *mascarpone* wynikowa tablica T to [5, 2, 4, 10, 1, 9, 8, 7, 6, 3],

dla słowa *kalafiorowa* wynikowa tablica T to [11, 4, 2, 5, 6, 1, 3, 7, 9, 8, 10].

Z wykorzystaniem funkcji `czy_mniejszy( $n, s, k1, k2$ )` zapisz w wybranej przez siebie notacji (w postaci pseudokodu lub w wybranym języku programowania) algorytm, który obliczy wartości elementów tablicy T zawierającej numery sufiksów zgodnie z porządkiem alfabetycznym sufiksów słowa s .

Uwaga: w zapisie możesz wykorzystać tylko operacje arytmetyczne (dodawanie, odejmowanie, mnożenie, dzielenie, dzielenie całkowite, reszta z dzielenia), odwoływanie się do pojedynczych elementów tablicy, porównywanie liczb lub znaków, instrukcje sterujące i przypisania lub samodzielnie napisane funkcje zawierające wyżej wymienione operacje.

Specyfikacja

Dane:

n – liczba całkowita dodatnia, długość słowa

$s[1..n]$ – słowo zapisane jako tablica znaków (numerowana od 1)

Wynik:

$T[1..n]$ – tablica T taka, że $T[i]$ -ty sufixs słowa s jest mniejszy w porządku alfabetycznym od $T[i + 1]$ -go sufixsu słowa s dla każdego $1 \leq i < n$.

Algorytm:

This image shows a full page of blank graph paper. The grid consists of thin, light gray horizontal and vertical lines that intersect to form small squares across the entire surface. There are no margins, text, or other markings on the paper.

Zadanie 2.4. (0–3)

W pliku `slova4.txt` znajduje się 10 wierszy. Każdy wiersz zawiera liczbę n ($1 \leq n \leq 100$) oraz n -literowe słowo s składające się z małych liter alfabetu angielskiego. Dane w wierszu są oddzielone znakiem odstępu.

Napisz program, który dla każdego słowa s z pliku wypisze jego sufiks najmniejszy w porządku alfabetycznym.

Przykład:

Sufiksem najmniejszym w porządku alfabetycznym dla słowa *mascarpone* jest *arpone*, a dla słowa *truskawki* sufiksem najmniejszym w porządku alfabetycznym jest *awki*.

Dla przykładowego pliku `sufiks_4.txt`, zawierającego tylko 4 wiersze (ze słowami: *banan*, *mascarpone*, *abcaabbaabbccba*, *maturazinformatyki*), Twój program powinien dać odpowiedź:

```
an
arpone
a
aturazinformatyki
```

Wyniki zapisz w pliku `wyniki2_4.txt`, każdy sufiks w oddzielnym wierszu, zgodnie z kolejnością danych w pliku `slova4.txt`.

Do oceny oddajesz:

- plik `wyniki2_4.txt` – zawierający odpowiedź do zadania 2.4.
- plik(i) zawierający(-e) kody źródłowe Twojego programu o nazwie(-ach):

.....

Zadanie 3. Anagram binarny

W pliku `anagram.txt` znajduje się 1000 wierszy. Każdy wiersz zawiera liczbę binarną, składającą się z maksymalnie 14 cyfr: 0 lub 1. Każda liczba zaczyna się jedyneką i żadna z nich się nie powtarza.

Napisz **program(y)**, który(-e) da(-dzą) odpowiedzi do podanych zadań. Odpowiedzi do zadań zapisz w pliku `wyniki3.txt`, a każdą z nich poprzedź numerem odpowiedniego zadania.

Uwaga: plik `przyklad.txt` zawiera 100 wierszy przykładowych danych spełniających warunki zadania. Odpowiedzi dla danych z pliku `przyklad.txt` są podane pod treściami zadań.

Zadanie 3.1. (0–2)

Liczbę binarną nazywamy *zrównoważoną*, gdy zawiera tyle samo zer i jedynek, natomiast *prawie zrównoważoną*, gdy liczba jedynek różni się od liczby zer o 1.

Przykład:

Liczba 101010 jest liczbą *zrównoważoną*.

Liczba 1011010 jest liczbą *prawie zrównoważoną*.

Podaj, ile jest liczb binarnych *zrównoważonych* oraz ile jest liczb binarnych *prawie zrównoważonych* w pliku `anagram.txt`.

Dla danych z pliku `przyklad.txt` prawidłową odpowiedzią jest:

21

15

Zadanie 3.2. (0–3)

Anagramy cyfrowe to liczby utworzone z tego samego zestawu cyfr ustawionych w różnych kolejnościach. Przy tym pierwsza cyfra liczby nie może być równa zero.

Przykład:

Z liczby 209 zapisanej dziesiętnie można utworzyć 4 anagramy: 209, 902, 290, 920.

Z liczby binarnej 11100 można utworzyć 6 różnych anagramów: 10011, 10101, 10110, 11001, 11010, 11100.

Znajdź wszystkie takie liczby binarne 8-cyfrowe w pliku `anagram.txt`, z których można utworzyć największą liczbę anagramów. Wypisz te liczby w kolejności, w jakiej występują w pliku `anagram.txt`.

Dla danych z pliku `przyklad.txt` prawidłową odpowiedzią jest:

10001011

10111000

10100111

11111000



10011100
11100011
10111010
10100011
10011010
10110001
11011010

Zadanie 3.3. (0–2)

Podaj największą wartość bezwzględną różnicy między sąsiednimi liczbami (to jest liczbami zapisanymi w sąsiednich wierszach np. 1 i 2 wierszu, 2 i 3 wierszu itd.) w pliku `anagram.txt`. Tę wartość podaj w zapisie binarnym.

Dla danych z pliku `przyklad.txt` prawidłową odpowiedzią jest:

1110001010

Zadanie 3.4. (0–4)

Zamień wszystkie liczby binarne z pliku `anagram.txt` na ich odpowiedniki w systemie dziesiętnym. Następnie spośród otrzymanych liczb dziesiętnych:

- podaj, ile jest takich, w których nie występuje cyfra zero
- podaj liczbę, która ma największą sumę **różnych** cyfr (jeśli liczb, które mają tę samą, największą sumę różnych cyfr, jest więcej niż jedna – podaj tę, która występuje jako pierwsza w pliku z danymi).

Przykład:

Dla liczby 20462 suma jej różnych cyfr to 12 ($2+0+4+6$), dla liczby 344 suma różnych cyfr to 7.

Dla danych z pliku `przyklad.txt` prawidłową odpowiedzią jest:

81

895

Do oceny oddajesz:

- plik tekstowy `wyniki3.txt`, zawierający odpowiedzi do poszczególnych zadań (odpowiedź do każdego zadania powinna być poprzedzona jego numerem)
- plik(i) zawierający(-e) kody źródłowe Twojego(-ich) programu(-ów) o nazwie(-ach) odpowiednio:

zadanie 3.1.

zadanie 3.2.

zadanie 3.3.

zadanie 3.4.

Zadanie 4. (0–2)

Uzupełnij brakujące pola tabeli:

- w wierszu pierwszym dla liczby zapisanej w systemie o podstawie 3 podaj jej zapis w systemie o podstawie 9
- w wierszu drugim dla liczby zapisanej w systemie o podstawie 9 podaj jej zapis w systemie o podstawie 3.

1.	101201 ₃	
2.		2487 ₉

Miejsce na obliczenia:

**Zadanie 5. (0–1)**

Zapis koloru tła w arkuszu CSS został wyrażony w postaci

background-color: #E9967A;

1.	Zapis dziesiętny składowej czerwonej koloru #E9967A to 233.	P	F
2.	Zmiana zapisu #E9967A na rgb(255,255,255) da w efekcie biały kolor tła.	P	F

Zadanie 6. Fotowoltaika

Pan Oszczędny pod koniec roku 2019 zamontował 18 paneli fotowoltaicznych na dachu swojego domu. Od 1 stycznia 2020 do 31 maja 2020 roku zapisywany był co godzinę przez całą dobę pobór prądu od dostawcy i generowany prąd przez panele fotowoltaiczne. W pliku `fotowoltaika.txt` zapisano datę i godzinę, pobór prądu z sieci energetycznej [kWh] oraz liczbę wygenerowanych kilowatogodzin przez panele. Dane w wierszach oddzielone są znakiem tabulacji.

Gospodarstwo domowe Pana Oszczędnego w pierwszej kolejności zużywa prąd wygenerowany przez panele fotowoltaiczne. Jeżeli potrzeby są większe, to pobiera go z sieci energetycznej. Pobór równy 0 oznacza, że ogniwa wytwarzają co najmniej tyle energii, ile wynosiły potrzeby gospodarstwa domowego. Generowanie równe 0 oznacza, że panele nie produkują energii (z powodu braku nasłonecznienia).

Przykładowy fragment pliku:

Data_godzina	Pobor [kWh]	Generowanie [kWh]
01.01.2020 01:00	0,367	0
01.01.2020 02:00	0,485	0
01.01.2020 03:00	0,299	0
01.01.2020 04:00	0,453	0
01.01.2020 05:00	0,409	0
01.01.2020 06:00	0,542	0
01.01.2020 07:00	0,416	0
01.01.2020 08:00	0,453	0
01.01.2020 09:00	0,35	0,001
01.01.2020 10:00	0,22	0,02
01.01.2020 11:00	0,178	0,07
01.01.2020 12:00	0,084	0,158
01.01.2020 13:00	0,25	0,165
01.01.2020 14:00	0,175	0,238
01.01.2020 15:00	2,057	0,129
01.01.2020 16:00	1,051	0
01.01.2020 17:00	2,179	0

Z wykorzystaniem dostępnych narzędzi informatycznych podaj odpowiedzi do poniższych zadań. Odpowiedzi zapisz w pliku `wyniki6.txt`, a każdą z nich poprzedź numerem odpowiedniego zadania.

Zadanie 6.1. (0–2)

Podaj dzień, w którym panele wytworzyły łącznie w ciągu całego dnia najwięcej energii liczonej w kWh. Podaj datę i liczbę wygenerowanych kilowatogodzin.

Zadanie 6.2. (0–2)

O której godzinie najczęściej zdarzała się sytuacja, że energia wytworzona przez ogniwa pokrywa w całości zapotrzebowanie gospodarstwa (pobór z sieci wynosił 0)? Podaj tę godzinę i liczbę dni, w których pobór z sieci wynosił 0 o tej godzinie.

Zadanie 6.3. (0–3)

Utwórz zestawienie średniej liczby wygenerowanych kWh w poszczególnych godzinach. Wartości zapisz z dokładnością do czterech miejsc po przecinku. Dla swojego zestawienia utwórz wykres kolumnowy. Pamiętaj o prawidłowym opisie osi oraz o tytule wykresu.

Zadanie 6.4. (0–2)

Pan Oszczędny rozważa rozbudowę systemu generowania energii za pomocą ogniw fotowoltaicznych. Bazą do obliczeń są wszystkie dane z **kwietnia** 2020 roku.

Podaj minimalną liczbę paneli, o którą należałoby powiększyć system, aby przy kwietniowym zużyciu prądu i kwietniowym nasłonecznieniu, w godzinach od 10 do 15 system nie pobierał prądu z zakładu energetycznego.

Do oceny oddajesz:

- plik tekstowy `wyniki6.txt`, zawierający odpowiedzi do poszczególnych zadań (odpowiedź do każdego zadania powinna być poprzedzona jego numerem)
- plik(i) zawierający(e) komputerową realizację Twoich obliczeń o nazwie(-ach):

.....
.....

Zadanie 7. Instalacje

W bazie danych firmy X zawarte są informacje o instalacjach pewnej aplikacji, o urządzeniach, na których ta aplikacja została zainstalowana, oraz o krajach, w których przeprowadzono instalację.

Dane zgromadzono w plikach tekstowych: `kraje.txt`, `instalacje.txt` oraz `urządzenia.txt`. Pierwszy wiersz każdego z plików jest wierszem nagłówkowym, a dane w wierszach rozdzielone są znakami tabulacji.

Plik o nazwie `kraje.txt` zawiera informacje o krajach, w których instalowano aplikację.

W każdym wierszu pliku znajdują się następujące dane:

`kod_k` – kod kraju (napis dwuznakowy)
`nazwa_k` – nazwa kraju (napis do 50 znaków)
`ludnosc_k` – ludność kraju (liczba całkowita do 10 cyfr określająca liczbę ludności).

Przykład:

<code>kod_k</code>	<code>nazwa_k</code>	<code>ludnosc_k</code>
AN	NETHERLANDS ANTILES	227049
CR	COSTA RICA	5003393
DZ	ALGERIA	42545964

Plik o nazwie `urzadzenia.txt` zawiera informacje o urządzeniach, na których może być instalowana aplikacja. W każdym wierszu pliku znajdują się następujące informacje:

`kod_u` – unikatowy kod (liczba całkowita co najwyżej 5-cyfrowa)
`nazwa_u` – nazwa urządzenia (napis do 80 znaków)
`producent_u` – producent urządzenia (napis do 35 znaków)
`typ_u` – typ urządzenia (napis: *Tablet*, *Phone* lub *PC*).

Uwaga: nazwa urządzenia nie jest unikatowa – w tabeli mogą występować dwa lub więcej urządzeń o tej samej nazwie.

Przykład:

<code>kod_u</code>	<code>nazwa_u</code>	<code>producent_u</code>	<code>typ_u</code>
12410	PLATINUM_E5	Sky Devices	Phone
6549	Ilium L1120	Lanix	Phone

Plik o nazwie `instalacje.txt` zawiera informacje o instalacjach aplikacji. W każdym wierszu pliku znajdują się następujące informacje:

`data_i` – data instalacji (w formacie dd.mm.rrrr)
`kod_u` – kod urządzenia, na którym była wykonana instalacja (liczba całkowita co najwyżej 5-cyfrowa)
`kod_k` – kod kraju, w którym znajdowało się to urządzenie (napis dwuznakowy).

Uwaga: `kod_u` nie oznacza pojedynczego egzemplarza urządzenia, a tylko jego rodzaj – to znaczy na urządzeniach o tym samym kodzie może być wykonanych wiele instalacji.

Przykład:

<code>data_i</code>	<code>kod_k</code>	<code>kod_u</code>
01.03.2019	AM	145
01.03.2019	AR	804
01.03.2019	AT	12632

Z wykorzystaniem danych zawartych w podanych plikach oraz dostępnych narzędzi informatycznych, podaj odpowiedzi do zadań 7.1.–7.4. Odpowiedzi zapisz w pliku `wyniki7.txt`, a każdą z nich poprzedź numerem odpowiedniego zadania.

Zadanie 7.1. (0–2)

Dla każdego typu urządzenia podaj liczbę instalacji aplikacji na tym typie urządzenia.

Zadanie 7.2. (0–2)

Podaj nazwę producenta urządzeń, dla którego w lutym 2019 wykonano najwięcej instalacji. Podaj liczbę tych instalacji.

Zadanie 7.3. (0–3)

Podaj nazwy pięciu krajów, w których przeprowadzono najwięcej instalacji w przeliczeniu na 1 000 000 mieszkańców, oraz podaj liczby tych instalacji.

Dla każdego z tych pięciu krajów podaj liczbę instalacji na 1 000 000 mieszkańców z dokładnością do dwóch miejsc po przecinku.

Uwaga: pomiń kraje, w których jest mniej niż milion mieszkańców.

Zadanie 7.4. (0–2)

Podaj kod oraz nazwę urządzenia typu tablet („Tablet”), na którym zainstalowano aplikację w największej liczbie krajów. Podaj także liczbę krajów, w których instalowano aplikację na tym urządzeniu.

Do oceny oddajesz:

- plik tekstowy `wyniki7.txt` zawierający odpowiedzi do zadań 7.1.–7.4. (odpowiedź do każdego zadania powinna być poprzedzona jego numerem)
- plik(i) zawierający(-e) komputerową realizację Twoich obliczeń o nazwie(-ach):

.....

.....

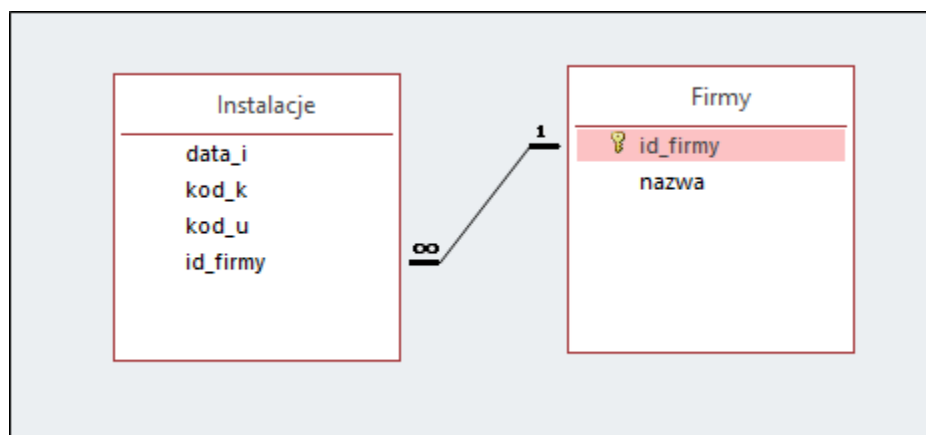
Zadanie 7.5. (0–2)

Do istniejących już tabel bazy danych dołączono tabelę *firmy* zawierającą dane firm, w których wykonywano instalacje aplikacji.

Tabela *firmy* zawiera pola *id_firmy* (identyfikator firmy – klucz podstawowy) oraz *nazwa* – nazwa firmy.

Do tabeli *instalacje* (zawierającej dane z pliku `instalacje.txt`) dodano pole *id_firmy* wskazujące, w której firmie na należących do niej urządzeniach wykonano instalację.

Tabele *firmy* i *instalacje* połączone są relacją jeden do wielu.



Miejsce na zapis zapytania:

A full-page view of a blank sheet of graph paper. The grid consists of small, uniform squares formed by thin, light gray lines. There are no margins, text, or other markings on the page.

BRUDNOPIS (*nie podlega ocenie*)

INFORMATYKA

Poziom rozszerzony

Formuła 2023



INFORMATYKA

Poziom rozszerzony

Formuła 2023



INFORMATYKA

Poziom rozszerzony

Formuła 2023

