

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-R0-100-2305

DATA: **22 maja 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 21 stron (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 odpowiedzi 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. Nie wpisuj żadnych znaków w tabelkach przeznaczonych dla egzaminatora. Tabelki umieszczone są na marginesie przy każdym zadaniu.
12. Pamiętaj, że zapisy w brudnopisie nie będą oceniane.



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

Zadanie 1. Biblioteczka Adama

Adam przechowuje swoje książki w biblioteczce zbudowanej z półek ponumerowanych kolejno 0, 1, 2, ... (zaczynając od półki położonej najwyżej). Półka o numerze i ma 2^i przegródek, w których umieszczane są książki. W jednej przegródce można umieścić tylko jedną książkę. Przegródki na i -tej półce są ponumerowane od lewej do prawej kolejnymi liczbami 1, 2, 3, ..., 2^i .

Jako $B[i, j]$ oznaczamy j -tą przegródkę na i -tej półce.

Przykład 1.

Szara komórka to przegródka $B[3, 4]$

		Numery przegródek															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Numery półek	0																
	1																
	2																
	3																
	4																

Biblioteczka Adama

Każda książka ma swój numer identyfikacyjny.

Adam ustawia książki na półkach, zawsze zaczynając od przegródki $B[0,1]$. Stosuje przy tym następującą, rekurencyjną regułę:

Adam sprawdza, czy przegródka $B[i, j]$ ($i \geq 0$ oraz $1 \leq j \leq 2^i$) jest pusta. Jeśli tak, umieszcza książkę w tej przegródce. W przeciwnym przypadku porównuje numer wstawianej książki z numerem książki w przegródce. Jeśli numer wstawianej książki jest mniejszy od numeru książki stojącej w przegródce, próbuje umieścić książkę na kolejnej półce w przegródce $B[i + 1, 2j - 1]$. Jeśli numer wstawianej książki jest większy od numeru książki w przegródce, to próbuje umieścić książkę w przegródce $B[i + 1, 2j]$.

Przykład 2.

Poniżej przedstawiono zawartość biblioteczki po wstawieniu do niej książek kolejno o numerach: 10, 2, 15, 13, 1, 5, 25 (zakładamy, że przedtem biblioteczka była pusta).

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Numery półek	0	10															
	1	2	15														
	2	1	5	13	25												
	3																
	4																
	5																

Poniżej przedstawiono zawartość biblioteczki po wstawieniu do niej książek kolejno o numerach: 1, 5, 10, 15, 2, 25, 13 (zakładamy, że przedtem biblioteczka była pusta).

[illegible]

1.1.

0-1-2

Numery książek wpisz we właściwe miejsca na poniższym schemacie.

[illegible]

Miejsce na obliczenia:

A full-page view of a blank sheet of graph paper. The grid consists of small squares formed by thin gray lines. There are 20 columns and 15 rows of squares. The margins are uniform on all sides.



Zadanie 2. Liczby binarne

W tym zadaniu rozważamy binarny zapis liczb całkowitych dodatnich.

Blokiem w zapisie binarnym liczby nazywamy każdy niepusty, maksymalny (nie można go rozszerzyć ani z lewej, ani z prawej strony) ciąg kolejnych takich samych cyfr w tym zapisie.

Przykład:

Liczba binarna 111110000110111 składa się z pięciu *bloków* – trzech *bloków* złożonych z jedynek (11111, 11 i 111) i dwóch *bloków* złożonych z zer (0000 i 0).

Liczba binarna 1111111111111111 składa się z jednego *bloku* złożonego z jedynek.

Zadanie 2.1. (0–3)

Zapisz w pseudokodzie lub w wybranym języku programowania algorytm, który dla danej dodatniej całkowitej liczby n obliczy liczbę *bloków* w jej zapisie binarnym.

Przykład:

Dla liczby **67** wynikiem jest 3, ponieważ 67 w zapisie binarnym to 1000011 (dwa *bloki* jedynek i jeden *blok* zer).

Dla liczby **245** wynikiem jest 5, ponieważ 245 w zapisie binarnym to 11110101 (trzy *bloki* jedynek i dwa bloki zer).

Uwaga: W zapisie algorytmu możesz korzystać tylko z instrukcji sterujących, operatorów arytmetycznych: dodawania, odejmowania, mnożenia, dzielenia, dzielenia całkowitego i reszty z dzielenia; operatorów logicznych, porównań, instrukcji przypisania lub samodzielnie napisanych funkcji i procedur wykorzystujących powyższe operacje. **Zabronione** jest używanie funkcji wbudowanych oraz operatorów innych niż wymienione, dostępnych w językach programowania, w tym zwłaszcza funkcji zamiany między systemami pozycyjnymi i konwersji między typami danych.

Specyfikacja:

Dane:

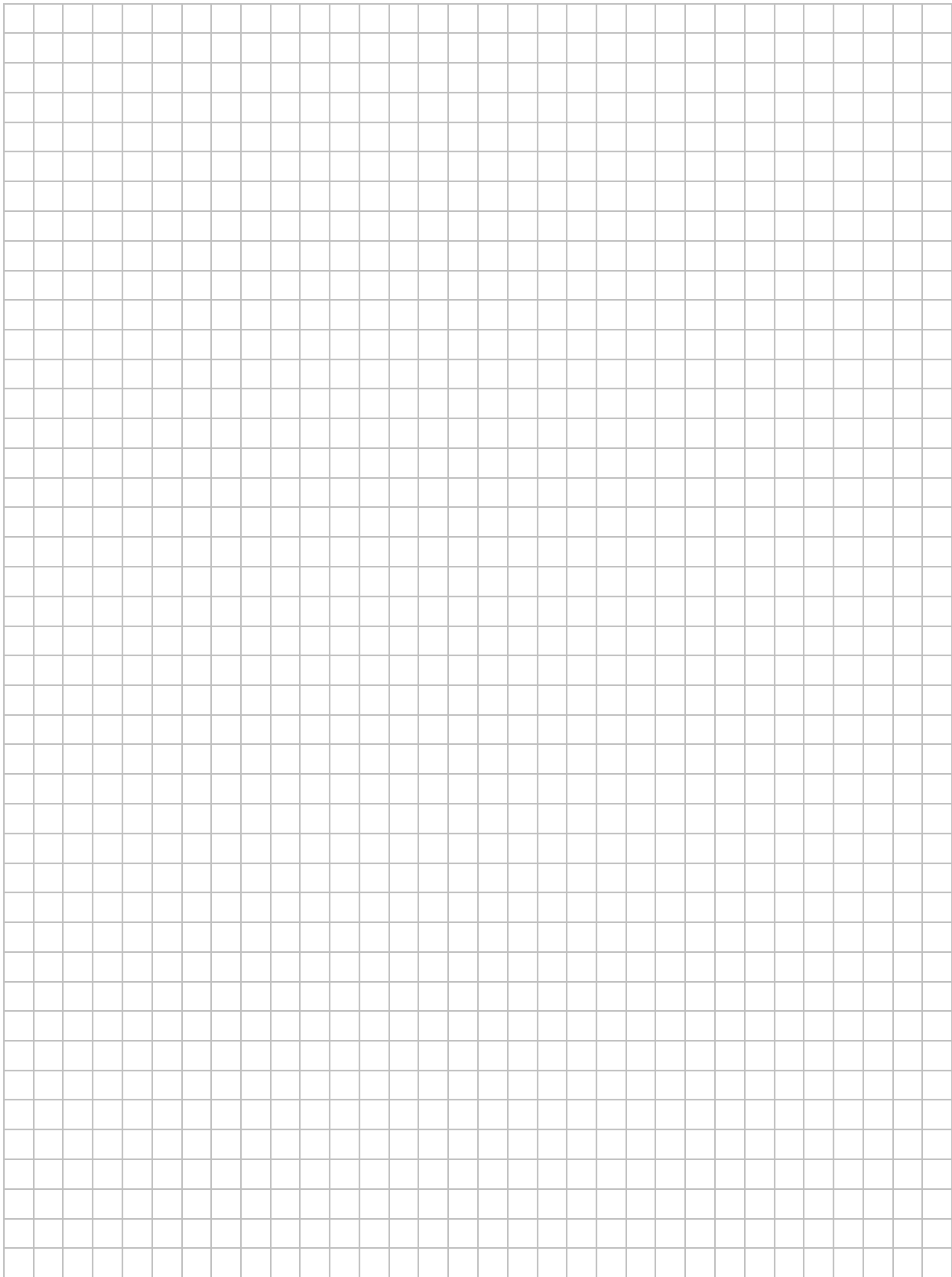
n – dodatnia liczba całkowita

Wynik:

b – liczba bloków w zapisie binarnym liczby n

Algorytm:

[illegible]



Informacja do zadań 2.2. i 2.3.

W pliku `bin.txt` znajduje się 100 wierszy. Każdy wiersz zawiera zapis binarny dodatniej liczby całkowitej składający się z co najwyżej dwudziestu cyfr (0 lub 1).

Napisz program(-y), który(-e) da(-dzą) odpowiedzi do poniższych zadań. Odpowiedzi zapisz w pliku `wyniki2.txt`, a każdą z nich poprzedź numerem odpowiedniego zadania.

Plik `bin_przyklad.txt` zawiera 100 wierszy przykładowych danych spełniających warunki zadania. Odpowiedzi dla danych z pliku `bin_przyklad.txt` są podane pod treściami zadań.

Zadanie 2.2. (0–2)

Podaj, ile liczb w pliku `bin.txt` składa się z **co najwyżej dwóch bloków** (zgodnie z definicją *bloku* podaną wcześniej).

Dla danych z pliku `bin_przyklad.txt` poprawna odpowiedź to 3.

Zadanie 2.3. (0–2)

Wypisz największą z liczb zapisanych w pliku `bin.txt`.

Dla danych z pliku `bin_przyklad.txt` poprawna odpowiedź to 10001111110111100000.

Zadanie 2.4. (0–1)

Dla nieujemnych liczb całkowitych a i b wynikiem operacji $a \text{ XOR } b$ jest liczba, której kolejne bity są wyliczane na podstawie poniższej tabelki z odpowiadających sobie bitów w zapisie binarnym liczb a i b . Jeśli jeden zapis jest krótszy od drugiego, to uzupełniamy go zerami z lewej strony (na najbardziej znaczących pozycjach).

p	q	$p \text{ XOR } q$
1	1	0
1	0	1
0	1	1
0	0	0

np.

$$4_{10} \text{ XOR } 7_{10} = 100_2 \text{ XOR } 111_2 = 011_2 = 3_{10}$$

$$6_{10} \text{ XOR } 11_{10} = 0110_2 \text{ XOR } 1011_2 = 1101_2 = 13_{10}$$

Oblicz $(123_{10} \text{ XOR } 101101_2) \text{ XOR } 2D_{16}$. Wynik podaj w systemie **dziesiętnym**.

Odpowiedź: _____

[illegible]

2.5.

0-1-
2-3

[illegible]

Otrzymane wyniki podaj w systemie binarnym. Zapisz je do pliku `wyniki2_5.txt` w kolejności występowania liczb w pliku `bin.txt`, każdy wynik w oddzielnym wierszu.

Odpowiedź dla danych z pliku `bin_przyklad.txt` znajduje się w pliku `odp bin przyklad.txt`.

- plik `wyniki2.txt` – zawierający odpowiedzi do zadań 2.2. i 2.3 (odpowiedź do każdego zadania powinna być poprzedzona jego numerem)
- plik `wyniki2_5.txt` – zawierający odpowiedź do zadania 2.5.
- pliki zawierające kody źródłowe Twoich programów o nazwach odpowiednio:

zadanie 2.2.

zadanie 2.3.

zadanie 2.5.

Zadanie 3. Liczba Pi

Pewien matematyk jest zafascynowany liczbą $\pi \approx 3,14159265\dots$ do tego stopnia, że zapisał jej rozwinięcie dziesiętne z dokładnością do 10 000 cyfr po przecinku. Wszystkie cyfry po przecinku zapisał w pliku tekstowym `pi.txt`.

Plik `pi.txt` zawiera 10 000 wierszy, każdy wiersz zawiera jedną cyfrę. W pierwszych 10 wierszach pliku zapisano zatem cyfry:

1
4
1
5
9
2
6
5
3
5

Matematyk zastanawia się, jakiego rodzaju regularności można zaobserwować w zebranych danych.

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

Plik `pi_przyklad.txt` zawiera 100 pierwszych wierszy pliku `pi.txt`. Odpowiedzi dla danych z tego pliku są podane pod treściami zadań.

Zadanie 3.1. (0–2)

Fragmentem 2-cyfrowym nazywamy dwie następujące po sobie cyfry w pliku `pi.txt`.

Wszystkich fragmentów 2-cyfrowych zapisanych w tym pliku jest 9 999. Ostatni rozpoczyna się w wierszu nr 9 999.

Przykładowe fragmenty 2-cyfrowe podano w poniższej tabeli.

i	Fragment 2-cyfrowy złożony z cyfr na pozycjach $i, i+1$
1	14
2	41
3	15
9	35

Znajdź liczbę wszystkich fragmentów 2-cyfrowych, które są zapisami dziesiętnymi liczb o wartościach **większych** od 90.

Dla danych zapisanych w pliku `pi_przyklad.txt` poprawna odpowiedź to 13.



Zadanie 3.2. (0–3)

Wszystkich możliwych różnych fragmentów 2-cyfrowych jest dokładnie 100. Są nimi fragmenty 00, 01, 02, ..., 99. Można sprawdzić, że np. 2-cyfrowy fragment równy 27 występuje w pliku `pi.txt` dokładnie 101 razy.

Znajdź fragmenty 2-cyfrowe, których liczba wystąpień w pliku `pi.txt` jest najmniejsza, oraz fragmenty 2-cyfrowe, których liczba wystąpień w pliku `pi.txt` jest największa. W wyniku podaj znalezione fragmenty 2-cyfrowe oraz liczby ich wystąpień.

W przypadku, gdy więcej niż jeden fragment występuje tyle samo razy, wypisz ten o mniejszej wartości liczbowej.

Dla danych w pliku `pi_przyklad.txt` poprawna odpowiedź to

00 0

62 4

(minimalna liczba wystąpień: fragment 00, liczba wystąpień 0; maksymalna liczba wystąpień: fragment 62, liczba wystąpień 4)

Informacja do zadań 3.3. i 3.4.

Skończony co najmniej 4-elementowy ciąg liczb $(a_1, a_2, \dots, a_n)$ jest *rosnąco-malejący*, jeśli można podzielić go na dwa ciągi, z których pierwszy jest rosnący, a drugi – malejący, tzn. jeśli istnieje takie $k \in \{2, 3, \dots, n-2\}$, że $a_1 < a_2 < \dots < a_k$ oraz $a_{k+1} > a_{k+2} > \dots > a_n$.

Przykład:

Ciąg (2, 5, 7, 9, 8, 3, 1) jest *rosnąco-malejący*, bo można go podzielić na dwa ciągi: rosnący (2, 5, 7) i malejący (9, 8, 3, 1) lub – odpowiednio – (2, 5, 7, 9) i (8, 3, 1). Ciąg (5, 9, 9, 4, 1) także jest *rosnąco-malejący*.

Przykłady ciągów, które nie są *rosnąco-malejące*, to: (2, 5, 8, 4, 3, 4, 5), (1, 2, 3, 4), (5, 5, 3, 2, 1).

Zadanie 3.3. (0–3)

Podaj, ile jest wszystkich *rosnąco-malejących* ciągów złożonych z dokładnie sześciu kolejnych cyfr zapisanych w pliku `pi.txt`.

Dla pliku `pi_przyklad.txt` poprawna odpowiedź to 3.

(w pliku `pi_przyklad.txt` są trzy ciągi *rosnąco-malejące* złożone z dokładnie sześciu cyfr: 028841, 089986, 899862)

3.2.

0–1–
2–3

--

3.3.

0–1–
2–3

--

Zadanie 3.4. (0–2)

Znajdź najdłuższy ciąg kolejnych cyfr z pliku `pi.txt`, który jest *rosnąco-malejący*, oraz pozycję, na której on się rozpoczyna. W pliku `pi.txt` jest tylko jeden taki ciąg o największej długości.

Wynik zapisz w dwóch wierszach: w pierwszym wierszu zapisz pozycję, od której zaczyna się znaleziony ciąg, a w drugim wypisz znaleziony ciąg. Cyfry ciągu zapisz jedną po drugiej, bez znaku odstępu.

Dla danych w pliku `pi_przyklad.txt` poprawna odpowiedź to

77

0899862

(najdłuższy ciąg *rosnąco-malejący* w pliku `pi_przyklad.txt` to ciąg 0899862 o długości 7 rozpoczynający się w 77 wierszu pliku).

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 6. Konfitury owocowe

W pliku `owoce.txt` zapisano informacje o dostawach owoców do przetwórnicy w okresie od 01.05.2020 do 30.09.2020.

W każdym wierszu podane są: data dostawy (dd.mm.rrrr), liczba kilogramów dostarczonych malin, liczba kilogramów dostarczonych truskawek i liczba kilogramów dostarczonych porzeczek, oddzielone znakiem tabulacji.

Dostawy odbywały się każdego dnia w wymienionym okresie.

Przykład:

data	dostawa_malin	dostawa_truskawek	dostawa_porzeczek
01.05.2020	211	281	88
02.05.2020	393	313	83
03.05.2020	389	315	104
04.05.2020	308	221	119

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. (0–3)

Dla każdego miesiąca pracy przetwórnicy (od maja do września) wykonaj zestawienie liczby dostarczonych kilogramów malin, liczby dostarczonych kilogramów truskawek i liczby dostarczonych kilogramów porzeczek.

Na podstawie wykonanego zestawienia utwórz wykres kolumnowy. Pamiętaj o czytelnym opisie wykresu (tytuł, legenda, opisy osi: na osi X – nazwy miesięcy, na osi Y – liczba kilogramów).

Zadanie 6.2. (0–1)

Podaj liczbę dni, w których dostarczono, spośród trzech rodzajów owoców, najwięcej porzeczek.

Informacja do zadań 6.3. i 6.4.

Przetwórnica produkuje konfitury: malinowo-truskawkowe, malinowo-porzeczkowe oraz truskawkowo-porzeczkowe (zawsze w proporcji owoców 1:1 oraz z wykorzystaniem maksymalnej dostępnej ilości owoców). Decyzja, jaka konfitura w danym dniu będzie produkowana, zależy od ilości owoców w przetwórnicy.

Owoce są dostarczane do przetwórnicy rano, przed rozpoczęciem produkcji. W danym dniu jest produkowany tylko jeden rodzaj konfitur. Do produkcji są brane owoce, których jest najwięcej w przetwórnicy (dla danych w pliku `owoce.txt` nie występuje przypadek, gdy ilość różnych owoców jest taka sama). Owoce niewykorzystane do produkcji są przechowywane w chłodni do następnego dnia. W następnym dniu podejmuje się decyzję o produkcji na ten dzień na podstawie łącznej ilości owoców pozostałych z poprzedniego dnia oraz dostarczonych rano.

Przykład:

Jeżeli 01.05.2020 dostarczono 211 kg malin, 281 kg truskawek i 88 kg porzeczek, to w tym dniu będzie produkowana konfitura malinowo-truskawkowa. Do produkcji wykorzystane zostanie 211 kg malin i 211 kg truskawek. Reszta truskawek i wszystkie porzeczki będą przechowywane w chłodni do następnego dnia.

Po dostawie z 02.05.2020 (393 kg malin, 313 kg truskawek i 83 kg porzeczek) w przetwórni będzie 393 kg malin, 383 kg truskawek i 171 kg porzeczek, czyli znowu będzie produkowana konfitura malinowo-truskawkowa.

Po uwzględnieniu opisanego powyżej cyklu produkcyjnego oraz danych zapisanych w pliku `owoce.txt` podaj odpowiedzi do poniższych zadań.

Zadanie 6.3. (0–3)

Podaj, ile razy, w okresie od 01.05.2020 do 30.09.2020, produkowano konfitury poszczególnych rodzajów.

6.3.0–1–
2–3**Zadanie 6.4. (0–3)**

Na wyprodukowanie 1 kg konfitur dwuowocowych potrzeba **po 1 kg** każdego owocu.

Podaj, ile kilogramów konfitur każdego rodzaju wyprodukowano w okresie od 01.05.2020 do 30.09.2020.

6.4.0–1–
2–3**Do oceny oddajesz:**

- plik tekstowy `wyniki6.txt`, zawierający odpowiedzi do poszczególnych zadań (odpowiedź do każdego zadania powinna być poprzedzona jego numerem)
- plik zawierający wykres do zadania 6.1. o nazwie
- plik(i) zawierający(e) komputerową realizację Twoich obliczeń o nazwie(-ach):

.....

.....

Zadanie 7. Gry planszowe

Pewien serwis internetowy prowadzi ranking gier planszowych. Baza serwisu została zapisana w trzech plikach.

Plik `gry.txt` zawiera informacje o grach planszowych. W każdym wierszu zapisano:

`id_gry` – unikatowy numer gry planszowej
`nazwa` – tytuł gry planszowej
`kategoria` – kategorię, do jakiej została zakwalifikowana gra planszowa; każda gra należy tylko do jednej kategorii.

Przykład:

<code>id_gry</code>	<code>nazwa</code>	<code>kategoria</code>
1	Wsiasc do Pociagu: Europa	familijna
2	Pandemia	kooperacyjna
3	Splendor	familijna
4	Dixit	familijna
5	Dobble	familijna

Plik `gracze.txt` zawiera informacje o graczach. W każdym wierszu zapisano:

`id_gracza` – unikatowy numer gracza
`imie` – imię gracza
`nazwisko` – nazwisko gracza
`wiek` – wiek gracza.

Przykład:

<code>id_gracza</code>	<code>imie</code>	<code>nazwisko</code>	<code>wiek</code>
1	Jozef	Gorecki	29
2	Przemyslaw	Mazurek	68
3	Cezary	Kaczmarczyk	41
4	Kornel	Wysocki	72
5	Eustachy	Gorecki	74

Plik `oceny.txt` zawiera oceny wystawione grom przez poszczególnych graczy. W każdym wierszu pliku zapisano:

`id_gry` – numer gry planszowej
`id_gracza` – numer gracza
`stan` – zawiera jedną z możliwych wartości: **posiada, chce kupic, sprzedal**, opisującą, czy użytkownik posiada daną grę, czy ją sprzedał lub czy zamierza ją zakupić
`ocena` – zawiera ocenę gry przez gracza, wyrażoną liczbą całkowitą w zakresie od 0 do 10.

Przykład:

<code>id_gry</code>	<code>id_gracza</code>	<code>stan</code>	<code>ocena</code>
66	1	posiada	8
72	1	chce kupic	3
79	1	sprzedal	8
43	2	posiada	9



We wszystkich plikach dane w wierszach są rozdzielone znakami tabulacji, a pierwszy wiersz w każdym pliku jest wierszem nagłówkowym.

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. (0–1)

Podaj tytuł gry, która otrzymała najwięcej ocen.

7.1.

0–1

Zadanie 7.2. (0–2)

Dla każdej gry z kategorii „imprezowa” podaj średnią jej ocen z dokładnością do dwóch miejsc po przecinku.

7.2.

0–1–2

Zadanie 7.3. (0–2)

Podaj liczbę graczy, którzy nie posiadają żadnej z ocenianych przez siebie gier (nie mają żadnej gry ze stanem „posiada”), a wystawili co najmniej jedną ocenę.

7.3.

0–1–2

Zadanie 7.4. (0–3)

W ocenianiu gier planszowych uczestniczą osoby w wieku od 10 do 99 lat. Osoby oceniające gry podzielono na trzy kategorie wiekowe: juniorzy (do 19 lat), seniorzy (od 20 do 49 lat) oraz weterani (od 50 lat).

Wykonaj zestawienie, w którym dla każdej kategorii wiekowej podasz największą liczbę ocen wystawionych jednej grze przez użytkowników z tej kategorii wiekowej oraz nazwy gier z tą liczbą ocen.

Jeżeli gier, które otrzymały taką samą największą liczbę ocen od użytkowników z danej kategorii wiekowej, jest więcej niż jedna – podaj tytuły ich wszystkich.

7.4.

0–1–
2–3

Do oceny oddajesz:

- plik tekstowy `wyniki7.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):

.....
.....

id_gry – identyfikator gry
cena – cena gry
promocja – informacja, czy cena jest ceną promocyjną (wartość *true* – jeśli cena jest promocyjna albo *false* – kiedy nie jest promocyjna)

```

erDiagram
    Sklep ||--}| Gry : "1"
    Sklep {
        string id_gry
        float cena
        string promocja
    }
    Gry {
        string id_gry
        string nazwa
        string kategoria
    }

```

- gra może mieć dwie ceny (cena w promocji i cena bez promocji), tj. może występować w tabeli *sklep* dwa razy
- tabela *sklep* zawiera tylko identyfikatory gier, które są w ofercie sklepu (nie musi zawierać wszystkich identyfikatorów z tabeli *gry*).

Miejsce na zapis zapytania

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.

BRUDNOPIS (*nie podlega ocenie*)

INFORMATYKA

Poziom rozszerzony

Formuła 2023



INFORMATYKA

Poziom rozszerzony

Formuła 2023



INFORMATYKA

Poziom rozszerzony

Formuła 2023

