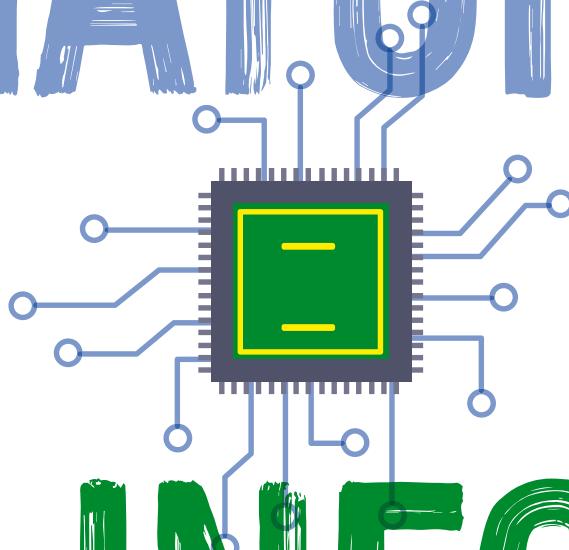


PRZEMYSŁAW GŁOWACZ WALDEMAR WALCZAK

# MATURA



## ZINFOR

## MATYKI

ZBIÓR ZADANÍ



**Helion**  
EDUKACJA

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiejkolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicielami.

Autor oraz wydawca dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz wydawca nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Redaktor prowadzący: Szymon Szwajger, Małgorzata Kulik

Helion S.A.  
ul. Kościuszki 1c, 44-100 Gliwice  
tel. 32 230 98 63  
e-mail: [helion@helion.pl](mailto:helion@helion.pl)  
WWW: <https://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!  
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres  
[https://helion.pl/user/opinie/matinf\\_ebook](https://helion.pl/user/opinie/matinf_ebook)  
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Kody źródłowe wybranych przykładów dostępne są pod adresem:  
<https://ftp.helion.pl/przyklady/matinf.zip>

ISBN: 978-83-289-0227-5

Copyright © Helion S.A. 2023

- [Poleć książkę na Facebook.com](#)
- [Kup w wersji papierowej](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

# Spis treści

<b>Zestaw 1 .....</b>	<b>5</b>
Zadanie 1. Ćwiartkowe zerowanie .....	5
Zadanie 2. Hetman .....	8
Zadanie 3. Odcinki .....	10
Zadanie 4. Ciągi .....	11
Zadanie 5. Podróże pociągiem .....	12
Odpowiedzi i wskazówki do zestawu 1 .....	17
<b>Zestaw 2 .....</b>	<b>33</b>
Zadanie 1. Szyfr przestawieniowy .....	33
Zadanie 2. Tajemnicze hieroglify .....	37
Zadanie 3. Temperatura w Nowym Jorku .....	39
Zadanie 4. Sprzedaż drewna .....	40
Zadanie 5. Sanatorium .....	42
Odpowiedzi i wskazówki do zestawu 2 .....	47
<b>Zestaw 3 .....</b>	<b>66</b>
Zadanie 1. Funkcja inkrementująca .....	66
Zadanie 2. Moc palindromiczna .....	69
Zadanie 3. Liceum dla dorosłych .....	71
Zadanie 4. Wakacyjne postanowienie Ani .....	73
Zadanie 5. Nie tylko SELECT .....	75
Odpowiedzi i wskazówki do zestawu 3 .....	80
<b>Zestaw 4 .....</b>	<b>97</b>
Zadanie 1. Bezpiecznie na nartach .....	97
Zadanie 2. Nowoczesny rolnik .....	104
Zadanie 3. Statystyka wypadków .....	108
Zadanie 4. Struktura książki .....	109
Zadanie 5. Piszemy w SQL-u .....	111
Odpowiedzi i wskazówki do zestawu 4 .....	118

---

<b>Zestaw 5 .....</b>	<b>140</b>
Zadanie 1. Wyścig kolarski .....	140
Zadanie 2. Elf złodziejaski .....	142
Zadanie 3. Grant naukowy .....	147
Zadanie 4. Firmowa poczta .....	149
Zadanie 5. Kwartał w łowiectwie .....	151
Odpowiedzi i wskazówki do zestawu 5 .....	156



Uwaga

Pliki źródłowe potrzebne do wykonania zadań dostępne są pod adresem:  
<https://ftp.helion.pl/przykłady/matinf.zip>

# Zestaw 1

## Zadanie 1. Ćwiartkowe zerowanie

Dane są dwie tablice jednowymiarowe złożone z  $n$ -elementów, gdzie  $n \geq 4$ , i zawierające dodatnie liczby naturalne:  $t[1..n]$  oraz  $p[1..n]$ .

Funkcja zeruj będzie modyfikowała zawartość tablicy  $p$  w oparciu o dane zapisane w tablicy  $t$  w następujący sposób: dopóki liczba elementów tablicy  $t$  jest większa lub równa 4, funkcja powtarza następujący zestaw czynności:

- a) wyznacza liczbę  $m$  równą jednej czwartej aktualnej liczby elementów tablicy  $t$ , zaokrąglonej w dół do liczby całkowitej,
- b) wyznacza liczbę  $min$  równą najmniejszemu spośród  $m$  pierwszych elementów tablicy,
- c) wyznacza liczbę  $maks$  równą największemu spośród  $m$  ostatnich elementów tablicy,
- d) w tablicy  $p$  „zeruje” (wstawia wartość zero) każdy element mniejszy od  $min$  oraz większy od  $maks$ ,
- e) usuwa z tablicy  $t$  pierwsze  $m$  elementów oraz ostatnie  $m$  elementów.

**Przykład:**

*Dla  $n = 15$  dane są tablice:*

$t[2,5,7,4,9,3,7,1,7,5,7,9,3,4,6]$

$p[6,2,8,3,6,5,1,8,9,2,5,4,2,7,6]$

$t[2,5,7,4,9,3,7,1,7,5,7,9,3,4,6] \quad m = 3, min = 2, maks = 6$

$p[6,2,8,3,6,5,1,8,9,2,5,4,2,7,6]$

$p[6,2,0,3,6,5,0,0,0,2,5,4,2,0,6]$

$t[4,9,3,7,1,7,5,7,9] \quad m = 2, min = 4, maks = 9$

$p[6,2,0,3,6,5,0,0,0,2,5,4,2,0,6]$

$p[6,0,0,0,6,5,0,0,0,0,5,4,0,0,6]$

$t[3,7,1,7,5]$                                $m = 1, \min = 3, \max = 5$

$p[6,0,0,0,6,5,0,0,0,2,5,4,2,0,6]$

$p[0,0,0,0,0,5,0,0,0,0,5,4,0,0,0]$

$t[7,1,7]$

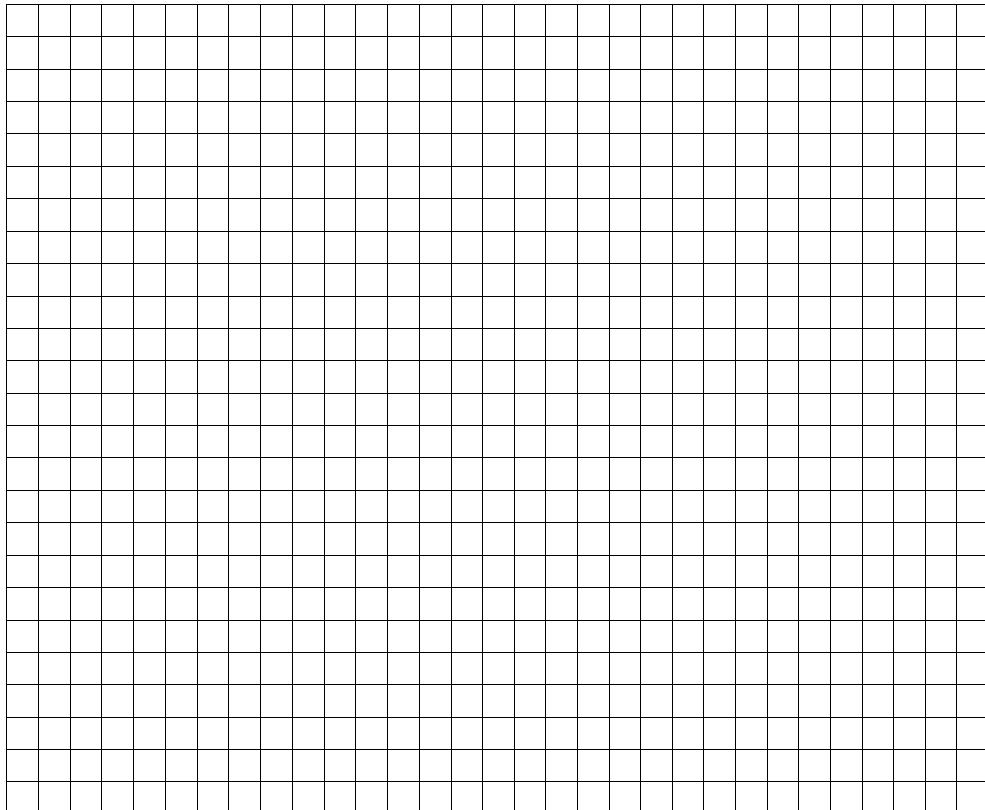
$p[0,0,0,0,0,5,0,0,0,0,5,4,0,0,0]$

## Zadanie 1.1.

Uzupełnij tabelę:

<b><i>n</i></b>	<b>Tablice <i>t</i> i <i>p</i> przed wywołaniem funkcji zeruj</b>	<b>Tablica <i>p</i> po zakończeniu działania funkcji zeruj</b>
9	$t[3,1,2,6,5,3,2,4,1]$ $p[5,2,7,8,4,4,9,2,4]$	$p[.,.,.,.,.]$
12	$t[1,7,3,3,1,2,8,5,7,2,5,8]$ $p[3,8,7,4,2,8,1,9,4,5,7,1]$	$p[.,.,.,.,.,.]$

Miejsce na obliczenia

A large grid of squares, approximately 20 columns by 30 rows, intended for students to perform their calculations on the exam paper.

## Zadanie 1.2.

Tablice  $t$  i  $p$  składają się z 1800 elementów. Przed wywołaniem funkcji zeruj obie tablice wypełniono cyframi w następujący sposób: pierwsze dwieście liczb w obu tablicach to cyfra 1, następne dwieście liczb w obu tablicach to cyfra 2, kolejne dwieście liczb w obu tablicach do cyfra 3 i tak dalej. Ostatnie dwieście liczb w obu tablicach to cyfra 9. Ile elementów tablicy  $p$  zostanie wyzerowanych po zakończeniu działania funkcji zeruj?

Miejsce na obliczenia

## Zadanie 1.3.

Zapisz w pseudojęzyku lub wybranym języku programowania algorytm, którego wynik działania dla danych tablic  $t$  i  $p$  będzie taki sam jak wynik działania funkcji zeruj.



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, odwoływanie się do pojedynczych elementów tablicy i 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. Dla uproszczenia możesz użyć funkcji usun( $t, m$ ), która usuwa  $m$  pierwszych i  $m$  ostatnich elementów tablicy  $t$ .

### Specyfikacja:

Dane:

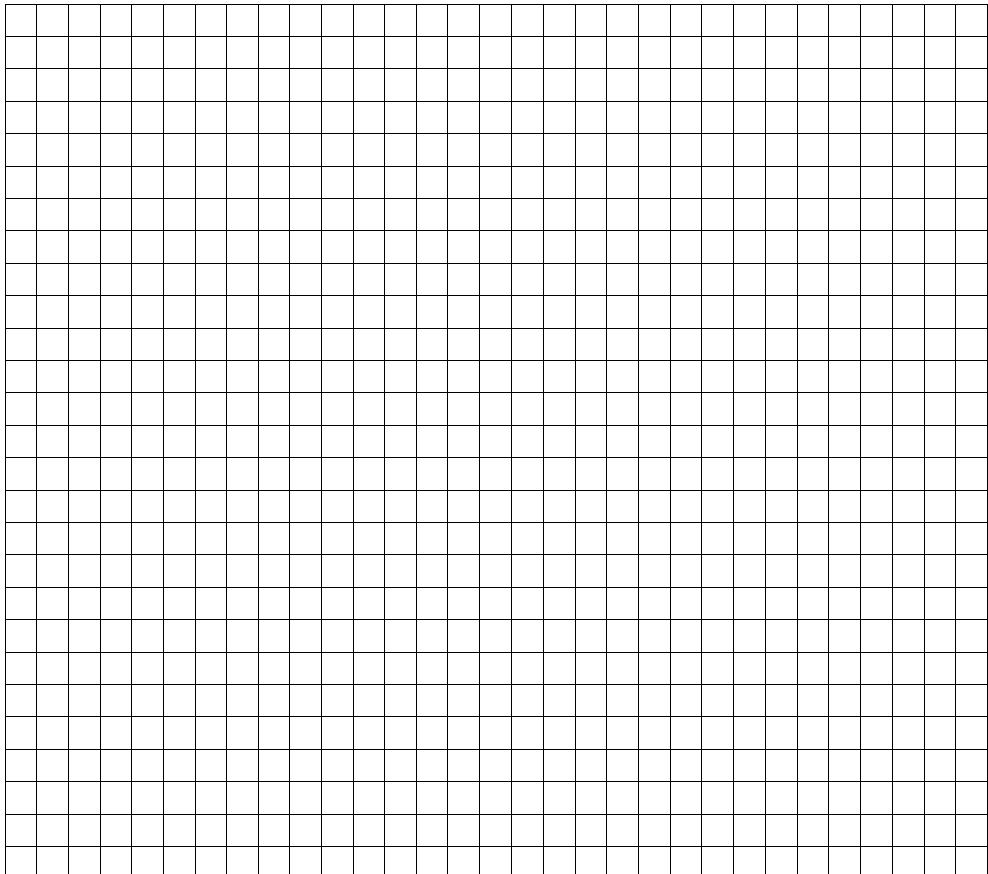
$n$  — dodatnia liczba całkowita

$t[1..n]$  — tablica zawierająca  $n$  dodatnich liczb naturalnych

$p[1..n]$  — tablica zawierająca  $n$  dodatnich liczb naturalnych

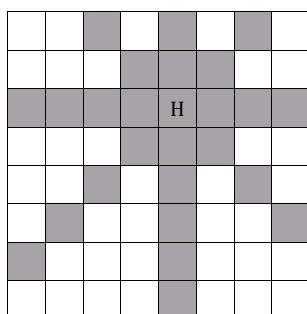
Wynik:

$p[1..n]$  — tablica po „zerowaniu”



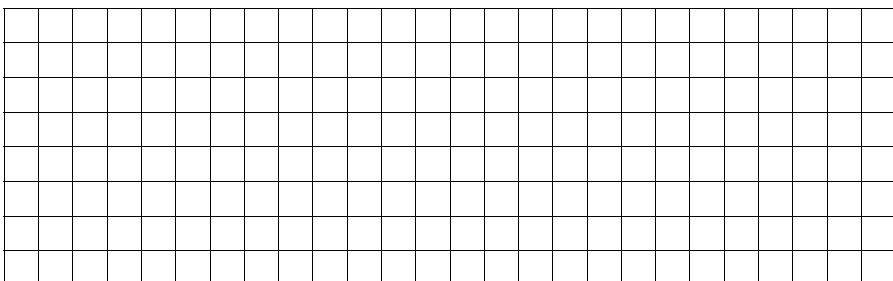
## Zadanie 2. Hetman

Hetman, potocznie zwany królową, to najsilniejsza figura w szachach. Podczas gry atakuje on wszystkie pola położone w poziomie, w pionie i na przekątnych względem swojego położenia. Na poniższym rysunku litera H oznacza położenie hetmana. Szarym kolorem zaznaczono wszystkie pola, które atakuje.



## Zadanie 2.1.

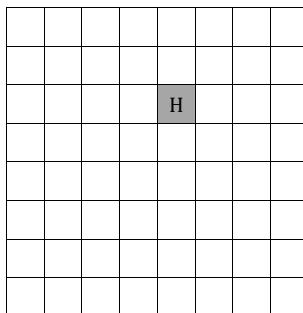
Podaj trzy różne rozmieszczenia ośmiu hetmanów, tak aby wzajemnie się **nie atakowały**.



### Informacje do zadań 2.2 – 2.4.

Podczas zawodów sportowych do opisu pozycji figury na polu szachowym stosuje się szachową notację algebraiczną, w której kolumny oznacza się małymi literami alfabetu od *a* do *h*, a wiersze liczbami naturalnymi od 1 do 8. W naszym zadaniu zawartość pola szachowego będziemy zapisywać za pomocą łańcucha znaków złożonego z 64 znaków. Pierwsze osiem znaków opisuje zawartość pierwszego wiersza, następnie osiem zawartość drugiego, kolejne osiem zawartość trzeciego wiersza i tak dalej. Ostatnie osiem znaków opisuje zawartość pól w ostatnim wierszu.

Na polu szachowym poniżej hetman znajduje się na polu położonym w piątej kolumnie i trzecim wierszu. Pozostałe pola są puste.



Zawartość tego pola szachowego (szachownicy) będzie zapisana następująco:

ppppppppppppppppppppHpp

W pliku *szachownice.txt* znajduje się 50 wierszy opisujących zawartości szachownic, na których znajduje się tylko jeden hetman. Opis szachownicy numer jeden znajduje się w pierwszym wierszu pliku, drugiej szachownicy w drugim wierszu pliku i tak dalej.

Napisz program (lub kilka programów), który znajdzie odpowiedzi do zadań 2.2 – 2.5.

## Zadanie 2.2.

Na ilu szachownicach, których opisy zawartości umieszczone zostały w pliku *szachownice.txt*, hetman znajduje się na brzegu szachownicy?

## Zadanie 2.3.

W przypadku ilu szachownic, których opisy zawartości umieszczone zostały w pliku *sza-chownice.txt*, położony tam hetman i nasz hetman położony w piątej kolumnie i trzecim wierszu atakowałby się wzajemnie?

## Zadanie 2.4.

Na których szachownicach, których opisy zawartości umieszczone zostały w pliku *sza-chownice.txt*, hetman atakuje co najmniej 23 pola?

# Zadanie 3. Odcinki

W pliku *odcinki.txt* znajduje się 10 wierszy. W każdym wierszu zapisano długości trzech odcinków. Długości odcinków są dodatnimi liczbami naturalnymi mniejszymi od tysiąca i oddzielone są spacjami.

**Przykład:**

33 35 17  
5 7 1  
6 18 34  
3 4 5  
600 700 841  
4 5 12  
33 66 99  
1 12 6  
55 3 2  
3 6 12

Napisz program (lub kilka programów), który znajdzie odpowiedzi lub pozwoli wykonać polecenia w zadaniach 3.1 – 3.4.

## Zadanie 3.1.

W przypadku ilu wierszy z podanych w nich odcinków można zbudować trójkąt?

## Zadanie 3.2.

Liczba naturalna jest liczbą **półpierwszą**, gdy jest ona iloczynem dwóch liczb pierwszych. Na przykład liczba 35 jest liczbą półpierwszą, ponieważ jest iloczynem dwóch liczb pierwszych: 5 i 7. W którym wierszu jest najwięcej liczb półpierwszych?

## Zadanie 3.3.

Pole trójkąta o bokach  $a, b, c$  można obliczyć, korzystając ze wzoru Herona:

$$P = \sqrt{p(p-a)(p-b)(p-c)}, \text{ gdzie } p = \frac{1}{2}(a+b+c).$$

Podaj numery wierszy, w których opisane zostały dwa trójkąty o największym polu.

## Zadanie 3.4.

O jakim największym obwodzie jest trójkąt, którego długości boków zapisane są w pliku *odcinki.txt*? Podaj obwód i liczbę trójkątów o tym obwodzie.

## Zadanie 4. Ciągi

W pliku *ciagi.csv* mamy 4 ciągi oznaczone jako  $c1, c2, c3, c4$ . Każdy składający się z tysiąca liczb.

**Przykład:**

$c1;c2;c3;c4$

$58;2;3;18$

$63;0;3;3$

$85;1;0;5$

$52;3;2;12$

$25;4;0;5$

$86;2;1;6$

.....

Z wykorzystaniem danych zawartych w pliku i dostępnych narzędzi informatycznych wykonaj poniższe polecenia.

## Zadanie 4.1.

Ile łącznie zer i jedynek występuje w ciągu  $c2$ ?

## Zadanie 4.2.

Ile jest takich pozycji, na których elementy wszystkich ciągów są takie same?

## Zadanie 4.3.

Który ciąg ma element występujący najczęściej? Jaki to element?

## Zadanie 4.4.

W którym ciągu występuje najdłuższy niemalejący podciąg?

Z ilu wyrazów się składa?

## Zadanie 4.5.

Utwórz wykres prezentujący liczbę trzech najczęściej wstępujących wyrazów w każdym ciągu.

## Zadanie 5. Podróże pociągiem

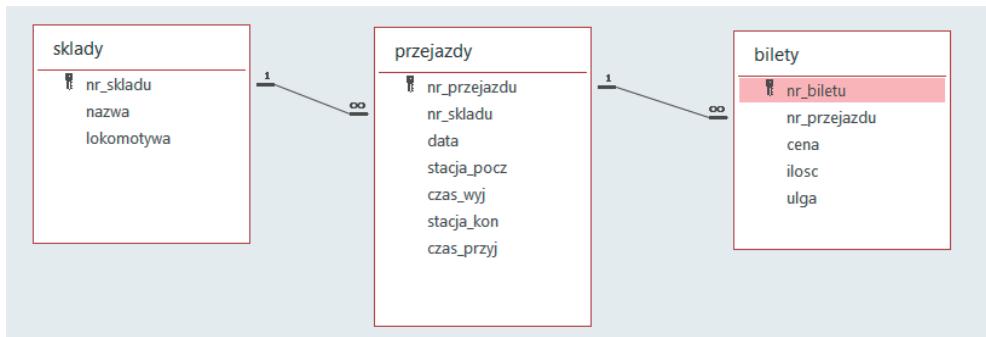
W pewnej miejscowości na dworcu kolejowym zainstalowano samoobsługowy terminal, za pomocą którego podróżny może kupić bilet. Terminal automatycznie po transakcji drukuje bilet. Informacje o dokonanych zakupach zapisywane są w bazie danych złożonej z trzech tabel.

Tabela **składy** (`nr_składu, nazwa, lokomotywa`) zawiera informacje o kursujących składach pociągów: numer składu, nazwa składu oraz model lokomotywy.

Tabela **przejazdy** (`nr_przejazdu, nr_składu, data, stacja_pocz, czas_wyj, stacja_kon, czas_przyj`) zawiera informacje o istniejących w rozkładzie jazdy przejazdach: identyfikator przejazdu, numer składu, za pomocą którego przejazd jest realizowany, dzień przejazdu, nazwa stacji początkowej, czas wyjazdu ze stacji początkowej, nazwa stacji końcowej i czas przyjazdu do stacji końcowej.

Tabela **bilety** (`nr_biletu, nr_przejazdu, cena, ilosc, ulga`) zawiera informacje o sprzedanych biletach: numer biletu, numer przejazdu, cena podstawowa bez zniżek, liczba biletów, ulga w punktach procentowych. W trakcie sprzedaży cena podstawowa jest pomniejszana o ulgę.

Między tabelami w bazie danych zawiązano następujące relacje:



Zawartości tabel zostały zapisane w plikach tekstowych, których nazwy odpowiadają nazwom tabel.

*składy.txt*

```

nr_skadlu;nazwa;lokomotywa
101;Bachus;EP-07
102;Zefir;EP-09
103;Cegielski;EP-09
104;Malczewski;EP-09
105;Barbakan;EP-07
106;Berolina;Husarz
107;Berolina;Husarz
    
```

108; Powstaniec Wielkopolski; EP-09  
109; Szkłarka; EP-09  
110; Hetman; Pendolino

przejazdy.txt

nr\_przejazdu;nr\_skladu;data;stacja\_pocz;czas\_wyj;stacja\_kon;czas\_przyj  
501;104;15.07.2022;Warszawa Wschodnia;08:12:00;Berlin Hauptbahnhof;14:55:00  
502;106;15.07.2022;Warszawa Wschodnia;16:10:00;Berlin Hauptbahnhof;20:55:00  
503;109;15.07.2022;Gdynia;03:12:00;Jelenia Gora;13:12:00  
504;108;16.7.2022;Konin;05:12:00;Poznan;07:12:00  
505;108;16.7.2022;Poznan;12:12:00;Konin;16:44:00  
506;105;17.07.2022;Katowice;06:12:00;Gdynia;14:55:00  
507;103;17.07.2022;Poznan;08:10:00;Katowice;12:40:00  
508;101;18.07.2022;Poznan;05:12:00;Zielona Gora;08:12:00  
509;102;18.7.2022;Katowice;07:12:00;Gdynia;14:12:00  
510;108;18.7.2022;Poznan;12:12:00;Konin;16:44:00

bilety.txt

nr\_biletu;nr\_przejazdu;cena;iłosc;ulga  
2001;501;180;1;0,33  
2002;501;180;2;0  
2003;502;69;1;0,33  
2004;502;69;2;0  
2005;503;180;2;0  
2006;504;32;10;0  
2007;506;120;1;0,33  
2008;504;32;1;0,33  
2009;509;180;1;0  
2010;503;180;1;0,33  
2011;504;32;2;0  
2012;509;180;1;0,33  
2013;504;32;2;0  
2014;505;32;1;0,33  
2015;506;120;1;0,33

Odpowiadając na pytania lub wykonując polecenia w zadaniach 5.1 – 5.8, postaraj się wykonać następujące czynności w podanej kolejności:

- a)** Na początek, nie używając komputera, napisz zapytanie w języku SQL.
  - b)** Następnie, analizując zawartości plików (podane wyżej) i nie używając komputera, spróbuj udzielić odpowiedzi. W celu ułatwienia analizy pliki składają się tylko z około 10 - 15 wierszy.
  - c)** Na końcu sprawdź poprawność odpowiedzi udzielonych w podpunktach a) i b), wykorzystując dostępne narzędzia informatyczne.

## Zadanie 5.1.

Które składы pociągów nie wyjechały (nie były używane w żadnym przejeździe)?

## Zadanie 5.2.

Napisz zapytanie, które wyświetli, posortowane malejąco według ilości, numery biletów na przejazd o numerze 504.

## Zadanie 5.3.

Ile modeli lokomotyw znajduje się w składach pociągów?

## Zadanie 5.4.

Podaj numer przejazdu, w którym łączna kwota zakupionych biletów na ten przejazd jest największa. Pamiętaj o uwzględnieniu liczby osób na bilecie i ewentualnie zastosowanej ulgi.

## Zadanie 5.5.

Z której stacji początkowej wyjechało najwięcej pasażerów, biorąc pod uwagę wszystkie przejazdy?

## Zadanie 5.6.

Podaj numer przejazdu na najkrótszym czasie podróży od stacji początkowej do stacji końcowej oraz numer przejazdu o najdłuższym czasie podróży.

## Zadanie 5.7.

Podaj numery przejazdów, na które nikt nie kupił biletu.

## Zadanie 5.8.

Który model lokomotywy i w którym dniu był najczęściej wykorzystywany?

# Odpowiedzi i wskazówki do zestawu 1

## Zadanie 1.1.

<i>n</i>	Tablice <i>t</i> i <i>p</i> przed wywołaniem funkcji zeruj	Tablica <i>p</i> po zakończeniu działania funkcji zeruj
9	$t[3,1,2,6,5,3,2,4,1]$ $p[5,2,7,8,4,4,9,2,4]$	$p[0,2,0,0,0,0,0,2,0]$
12	$t[1,7,3,3,1,2,8,5,7,2,5,8]$ $p[3,8,7,4,2,8,1,9,4,5,7,1]$	$p[3,0,0,4,0,0,0,0,4,5,0,0]$

## Zadanie 1.2.

Liczba elementów wyzerowanych = **1600**

## Zadanie 1.3.

Przykładowy algorytm:

```

N<-n
dopóki n>=4
    m <- n div 4
    min <- t[1]
    dla i = 2,...,m wykonuj:
        jeżeli t[i] < min
            min<-t[i]
    maks<-t[n]
    dla i=n-1,..., n-m+1 wykonuj:
        jeżeli t[i] > maks
            maks <-t[i]
    dla i=1...N wykonuj:
        jeżeli p[i] < min
            p[i] <-0;
        jeżeli p[i] > maks
            p[i]<-0
usun(t,m)
n<-n - 2*m

```

## Zadanie 2.1.

Przykład:

			H					
							H	
		H						
								H
		H						
				H				
H								
								H

## Zadanie 2.2.

```
#include <iostream>
#include <fstream>
using namespace std;
main()
{
    int h[8][8];
    for(int i=0; i<8; i++)
        for(int j=0; j<8; j++)
            h[i][j] = 0;
    int h_i = 2;
    int h_j = 4;
    h[h_i][h_j] = 1; //pozycja hetmana z zadania
    for(int j=0; j<8; j++)
        h[h_i][j] = 1;
    for(int i=0; i<8; i++)
        h[i][h_j] = 1;
    for(int i=h_i+1, j=h_j+1; i<8 && j<8 ;i++,j++)
        h[i][j] = 1;
    for(int i=h_i-1, j=h_j+1; i>=0 && j<8 ;i--,j++)
        h[i][j] = 1;
    for(int i=h_i+1, j=h_j-1; i<8 && j>=0 ;i++,j--)
        h[i][j] = 1;
    for(int i=h_i-1, j=h_j-1; i>=0 && j>=0 ;i--,j--)
        h[i][j] = 1;
    //sprawdzenie
    for(int i=0; i<8; i++)
    {
        for(int j=0; j<8; j++)
            cout<<h[i][j]<<" ";
        cout<<endl;
    }
    int szachownica[8][8]; //położenie hetmana w kolejnym wierszu
    int poz_i, poz_j;
    ifstream plik;
    plik.open("szachownice.txt");
    string wiersz;
    int licznik = 0; //liczy pozycje zadania 2.2
    for(int i=1; i<=50; i++)

```

```

{
    plik>>wiersz;
    int k=0; //indeksowanie literek w wierszu
    for(int i=0; i<8; i++) // kolejny wiersz
        for(int j=0; j<8; j++) // kolejna kolumna
        {
            if(wiersz[k]=='H')
            {
                szachownica[i][j] = 1;
                poz_i = i;
                poz_j = j;
            }
            else szachownica[i][j]=0;
            k++;
        }
        if(poz_i == 0 || poz_i == 7 || poz_j == 0 || poz_j == 7) licznik++;
    }
    cout<<"zad 2.2 = "<<licznik;
    plik.close();
}

```

**Odpowiedź:**

Na 21 szachownicach hetman znajduje się na brzegu szachownicy.

**Zadanie 2.3.**

```

int licznik = 0; // wzajemne atakowanie
for(int i=1; i<=50; i++)
{
    plik>>wiersz;
    int k=0;
    for(int i=0; i<8; i++)
        for(int j=0; j<8; j++)
        {
            if(wiersz[k]=='H')
            {
                szachownica[i][j] = 1;
                poz_i = i;
                poz_j = j;
            }
            else szachownica[i][j]=0;
            k++;
        }
        if(h[poz_i][poz_j] == 1) licznik++;
}
cout<<licznik;

```

**Odpowiedź:**

Jest 18 ustawień, w których hetman w pliku i hetman z zadania wzajemnie by się atakowały.

## Zadanie 2.4.

```
#include <iostream>
#include <fstream>
using namespace std;
main()
{
    int h[8][8];
    for(int i=0; i<8; i++)
        for(int j=0; j<8; j++)
            h[i][j] = 0;
    int h_i = 2;
    int h_j = 4;
    h[h_i][h_j] = 1;
    for(int j=0; j<8; j++)
        h[h_i][j] = 1;
    for(int i=0; i<8; i++)
        h[i][h_j] = 1;
    for(int i=h_i+1, j=h_j+1; i<8 && j<8 ;i++,j++)
        h[i][j] = 1;
    for(int i=h_i-1, j=h_j+1; i>=0 && j<8 ;i--,j++)
        h[i][j] = 1;
    for(int i=h_i+1, j=h_j-1; i<8 && j>=0 ;i++,j--)
        h[i][j] = 1;
    for(int i=h_i-1, j=h_j-1; i>=0 && j>=0 ;i--,j--)
        h[i][j] = 1;
    for(int i=0; i<8; i++)
    {
        for(int j=0; j<8; j++)
            cout<<h[i][j]<<" ";
        cout<<endl;
    }
    int szachownica[8][8];
    int poz_i, poz_j;
    ifstream plik;
    plik.open("szachownice.txt");
    string wiersz;
    int licznik = 0;
    for(int i=1; i<=50; i++)
    {
        plik>>wiersz;
        int k=0;
        for(int i=0; i<8; i++)
            for(int j=0; j<8; j++)
            {
                if(wiersz[k]=='H')
                {
                    szachownica[i][j] = 1;
                    poz_i = i;
                    poz_j = j;
                }
                else szachownica[i][j]=0;
            }
        k++;
    }
    for(int j=0; j<8; j++)
        szachownica[poz_i][j] = 1;
    for(int i=0; i<8; i++)
        szachownica[i][poz_j] = 1;
```

```

        for(int i=poz_i+1, j=poz_j+1; i<8 && j<8 ;i++,j++)
            szachownica[i][j] = 1;
        for(int i=poz_i-1, j=poz_j+1; i>=0 && j<8 ;i--,j++)
            szachownica[i][j] = 1;
        for(int i=poz_i+1, j=poz_j-1; i<8 && j>=0 ;i++,j--)
            szachownica[i][j] = 1;
        for(int i=poz_i-1, j=poz_j-1; i>=0 && j>=0 ;i--,j--)
            szachownica[i][j] = 1;
        int licznik2 = 0;
        for(int i=0; i<8; i++)
            for(int j=0; j<8; j++)
                if(szachownica[i][j]==1)
                    licznik2++;
        if(licznik2>=24) licznik++; //23 + położenie hetmana
    }
    cout<<licznik;
    plik.close();
}

```

**Odpowiedź:**

W 29 przypadkach.

**Zadanie 3.1.**

```

#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    ifstream plik("odcinki.txt");
    int x, y, z;
    int licznik = 0;
    for(int i=1; i<=10; i++)
    {
        plik>>x>>y>>z;
        if(x+y>z && x+z>y && y+z>x) licznik++;
    }
    cout<<licznik;
    plik.close();
}

```

**Odpowiedź:**

Są 3 takie wiersze.

**Zadanie 3.2.**

```

#include <iostream>
#include <fstream>
using namespace std;
bool pierwsza(int n)
{
    for(int i=2; i<n; i++)
        if(n%i==0) return false;
    return true;
}
int main()

```

```

{
ifstream plik("odcinki.txt");
int x, y, z;
int liczba_pp = 0;
for(int j=1; j<=10; j++)
{
    plik>>x>>y>>z;
    liczba_pp = 0;
    for(int i=2; i<x; i++)
    {
        if(x%i==0)
        {
            if(pierwsza(i)&&pierwsza(x/i))
            {
                liczba_pp++;
                break;
            }
        }
    }
    for(int i=2; i<y; i++)
    {
        if(y%i==0)
        {
            if(pierwsza(i)&&pierwsza(y/i))
            {
                liczba_pp++;
                break;
            }
        }
    }
    for(int i=2; i<z; i++)
    {
        if(z%i==0)
        {
            if(pierwsza(i)&&pierwsza(z/i))
            {
                liczba_pp++;
                break;
            }
        }
    }
    if(liczba_pp==3) cout<<"wiersz:<< j<< - 3 liczby"<<endl;
    if(liczba_pp==2) cout<<"wiersz:<< j<< - 2 liczby"<<endl;
    if(liczba_pp==1) cout<<"wiersz:<< j<< - 1 liczba"<<endl;
    if(liczba_pp==0) cout<<"wiersz:<< j<< - 0 liczby"<<endl;
}
plik.close();
}

```

**Odpowiedź:**

W wierszu 1 i 3 dwie liczby.

**Zadanie 3.3.**

```
#include <iostream>
#include <fstream>
#include <cmath>
```

```
using namespace std;
main()
{
double p;
double pole;
ifstream plik;
plik.open("odcinki.txt");
double a, b, c;
int wiersze[10];
double pola[10];

for(int i=1; i<=10; i++)
{
    wiersze[i-1] = i;
    plik>>a>>b>>c;
    if ((a+b>c) && (a+c>b) && (c+b>a))
    {
        p =(a+b+c)/2.0;
        pole = sqrt(p*(p-a)*(p-b)*(p-c));
        pola[i-1] = pole;
    }
    else
        pola[i-1] = 0;
}
//wypisanie
for(int i=0; i<10; i++)
{
    if(pola[i]!=0)
    {
        cout<<wiersze[i] <<" "<<pola[i]<<endl;
    }
}
//sortowanie tablic z wartościami pól i numerami wierszy
for(int i=1; i<10; i++)
{
    for(int j=0; j<10-1;j++)
        if(pola[j]>pola[j+1])
        {
            int pom = pola[j];
            pola[j] = pola[j+1];
            pola[j+1] = pom;
            pom = wiersze[j];
            wiersze[j] = wiersze[j+1];
            wiersze[j+1] = pom;
        }
}
cout<<"wynik="<<endl;
cout<<"wiersz="<<wiersze[9]<<" pole="<<pola[9]<<endl;
cout<<"wiersz="<<wiersze[8]<<" pole="<<pola[8]<<endl;
plik.close();
}
```

**Odpowiedź:**

W wierszu 1 i 5.

## Zadanie 3.4.

```

#include <iostream>
#include <fstream>
#include <cmath>
using namespace std;
main()
{
double p;
double pole;
ifstream plik;
plik.open("odcinki.txt");
double a, b, c;
double obwod, maxobwod = 0;

for(int i=1; i<=10; i++)
{
    plik>>a>>b>>c;
    if ((a+b>c) && (a+c>b) && (c+b>a))
    {
        obwod = a+b+c;
        if(obwod>maxobwod)
            maxobwod = obwod;
    }
}
plik.close();
int licznik = 0;
plik.open("odcinki.txt");
for(int i=1; i<=10; i++)
{
    plik>>a>>b>>c;
    if ((a+b>c) && (a+c>b) && (c+b>a))
    {
        obwod = a+b+c;
        if(obwod == maxobwod)
            licznik++;
    }
}
plik.close();
cout<<"maxobwod = "<<maxobwod<<endl;
cout<<"liczba maxobwod = "<<licznik;
}

```

**Odpowiedź:**

Maksymalny obwód: 2141. Jest tylko jeden taki trójkąt.

## Zadanie 4.1.

E	F	G	H
	=LICZ.JEŻELI(B:B;0) + LICZ.JEŻELI(B:B;1)		

**Odpowiedź:**

290

## Zadanie 4.2.

Poprzez 1 oznaczamy te pozycje, na których wyrazy wszystkich ciągów mają taką samą wartość. Następnie sumujemy, ile tych pozycji jest:

	A	B	C	D	E	F	G	H
1	c1	c2	c3	c4	takie same			
2	58	2	3	18	=JEŻELI(ORAZ(A2=B2;A2=C2;A2=D2);1;0)			
3	63	0	3	3	0			
4	85	1	0	5	0			
5	52	3	2	12	0	=SUMA(E:E)		
6	25	4	0	5	0			
7	86	2	1	6	0			
8	31	3	1	11	0			

Odpowiedź:

34

## Zadanie 4.3.

Dla **każdego ciągu** tworzymy tabelę przestawną, która policzy, ile razy występuje każda wartość w ciągu. Dodatkowo wyniki sortujemy malejąco.

Np. dla c2:

L	M	O
3	Etykiety wierszy	Liczba z c2
4	1	152
5	2	150
6	3	146
7	6	138
8	0	138
9	4	138
10	5	137
11	8	1
12	Suma końcowa	1000

Pola tabeli przestawnej

Wybierz pola, które chcesz dodać do raportu:

Wyszukaj

c1  
 c2  
 c3  
 c4  
 takie same

Więcej tabel...

Przeciągnij pola między obszarami poniżej:

Filtery Kolumny

Wiersze Wartości

c2 Liczba z c2

Odpowiedź:

W c3 liczba 3 występuje 215 razy.

## Zadanie 4.4.

Dla każdego ciągu wyodrębniamy wszystkie podciągi niemalejące:

A	B	C	D	E	F	G	H	I	J	K
c1	c2	c3	c4	takie same	niemalejący c1	niemalejący c2	niemalejący c3	niemalejący c4		max
58	2	3	18	0	1	1	1	1	=MAX(F:I)	
63	0	3	3	0	=JEŻELI(A2<=A3;E2+1;1)		1	2		
85	1	0	5	0		3	2	1		
52	3	2	12	0		1	3	2		
25	4	0	5	0		1	4	1		
86	2	1	6	0		2	1	2		
31	3	1	11	0		1	2	3		
97	6	2	17	0		2	3	4		

Następnie wyznaczamy maksymalną wartość i szukamy jej w arkuszu.

**Odpowiedź:**

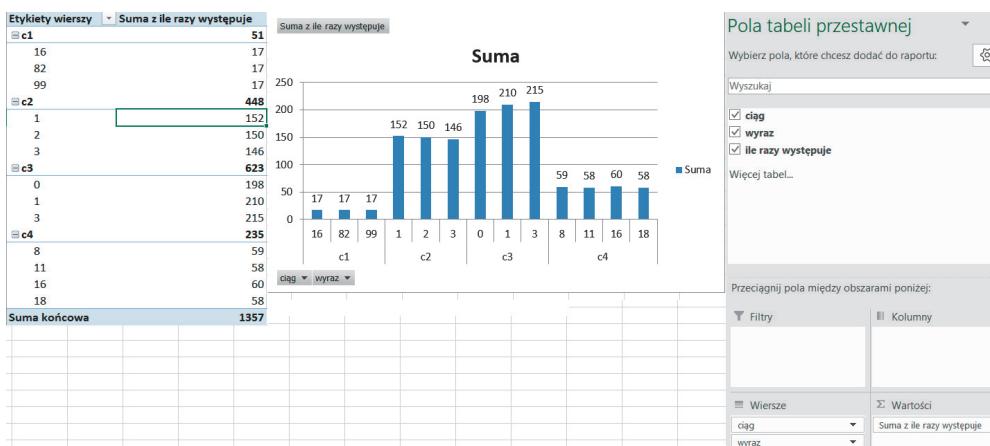
W c2. Składa się z 8 wyrazów.

## Zadanie 4.5.

Z wcześniejszej przygotowanych i posortowanych tabel przestawnych przygotowujemy dane w nowym arkuszu:

A	B	C
1 ciąg	wyraz	ile razy występuje
2 c1	16	17
3 c1	82	17
4 c1	99	17
5 c2	1	152
6 c2	2	150
7 c2	3	146
8 c3	3	215
9 c3	1	210
10 c3	0	198
11 c4	16	60
12 c4	8	59
13 c4	18	58
14 c4	11	58

Następnie tworzymy na podstawie tych danych wykres przestawny:



## Zadanie 5.1.

```
SELECT *
FROM Sklady
WHERE not EXISTS (SELECT nr_przejazdu
FROM Przejazdy
WHERE Sklady.nr_skladu = Przejazdy.nr_skladu);
```

The screenshot shows the Microsoft Access query builder interface. At the top right is the 'Właściwości sprzężenia' (Join Properties) dialog, which is set to option 2: "Uwzględnia WSZYSTKIE rekordy z „Sklady” i tylko te rekordy z „Przejazdy”, dla których sprężone pola są równe." (Shows all records from "Sklady" and only those from "Przejazdy" where joined fields are equal). Below the dialog is the SQL query grid, which contains the generated SQL code:

```
Pole: nazwa      lokomotywa      nr_skladu      nr_skladu
Tabela: Sklady   Sklady          Sklady         Przejazdy
Sortuj:          checked        checked        checked
Pokaz:          checked        checked        checked
Kryteria: lub:  Is Null
```

**Odpowiedź:**

nazwa	lokomotywa	nr_skladu
Berolina	Husarz	107
Hetman	Pendolino	110

## Zadanie 5.2.

```
SELECT Bilety.nr_biletu, Bilety.nr_przejazdu, Bilety.ilosc
FROM Bilety
WHERE ((Bilety.nr_przejazdu)=504)
ORDER BY Bilety.ilosc DESC;
```

The screenshot shows the Microsoft Access query builder interface. At the top is the 'Bilety' table, with its primary key 'nr\_biletu' highlighted. Below it is the SQL query grid, which contains the generated SQL code:

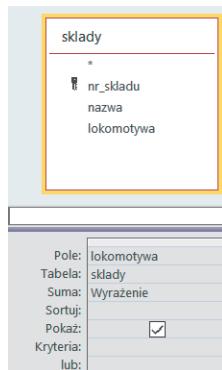
```
Pole: nr_biletu      nr_przejazdu      ilosc
Tabela: Bilety       Bilety           Bilety
Sortuj:             checked        checked        checked
Kryteria: lub:      504
```

**Odpowiedź:**

nr_biletu	nr_przejazdu	ilosc
2006	504	10
2013	504	2
2011	504	2
2008	504	1

### Zadanie 5.3.

```
SELECT DISTINCT sklady.lokomotywa
FROM sklady;
```



**Odpowiedź:**

lokomotywa

EP-07

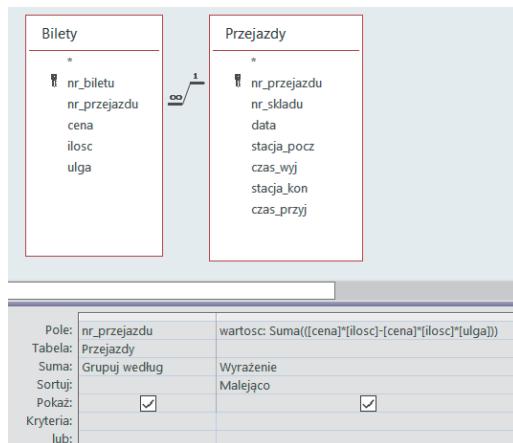
EP-09

Husarz

Pendolino

### Zadanie 5.4.

```
SELECT Przejazdy.nr_przejazdu, Sum(([cena]*[ilosc]-[cena]*[ilosc]*[ulg]) AS
→wartosc
FROM Przejazdy INNER JOIN Bilety ON Przejazdy.nr_przejazdu = Bilety.nr_przejazdu
GROUP BY Przejazdy.nr_przejazdu
ORDER BY Sum(([cena]*[ilosc]-[cena]*[ilosc]*[ulg])) DESC;
```

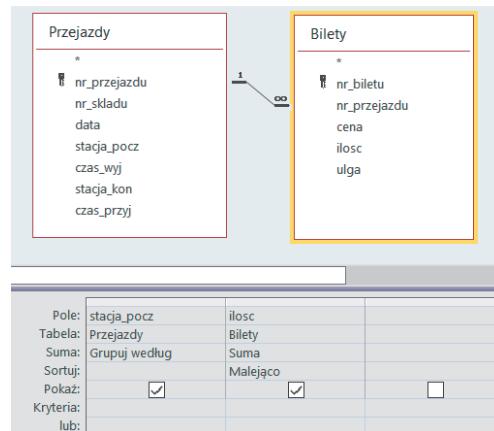


Odpowiedź:

nr_przejazdu	wartość
503	480,6

## Zadanie 5.5.

```
SELECT Przejazdy.stacja_pocz, Sum(Bilety.ilosc) AS Suma0filosc
FROM Przejazdy INNER JOIN Bilety ON Przejazdy.nr_przejazdu = Bilety.nr_przejazdu
GROUP BY Przejazdy.stacja_pocz
ORDER BY Sum(Bilety.ilosc) DESC;
```



Odpowiedź:

stacja_pocz	Suma0filosc
Konin	15

## Zadanie 5.6.

```
SELECT Przejazdy.nr_przejazdu, ([czas_przyj]-[czas_wyj]) AS Wyr1
FROM Przejazdy
ORDER BY ([czas_przyj]-[czas_wyj]) DESC;
```

The screenshot shows the Microsoft Access query builder interface. At the top, there is a table named "Przejazdy" with columns: nr\_przejazdu, nr\_sładu, data, staćja\_pocz, czas\_wyj, staćja\_kon, and czas\_przyj. Below the table, the query definition is displayed:

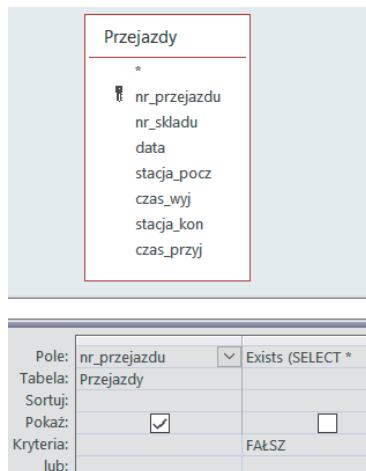
Pole:	nr_przejazdu	Wyr1: ([czas_przyj]-[czas_wyj])
Tabela:	Przejazdy	
Sortuj:		Malejaco
Pokaż:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kryteria:	lub:	

Odpowiedź:

nr_przejazdu	Wyr1
503	0,416666666666667
506	0,363194444444444
509	0,291666666666667
501	0,279861111111111
502	0,197916666666667
510	0,188888888888889
505	0,188888888888889
507	0,1875
508	0,125
504	8,3333333333333E-02

## Zadanie 5.7.

```
SELECT Przejazdy.nr_przejazdu
FROM Przejazdy
WHERE (((Exists (SELECT *
FROM Bilety
WHERE Przejazdy.nr_przejazdu = Bilety.nr_przejazdu))=False));
```

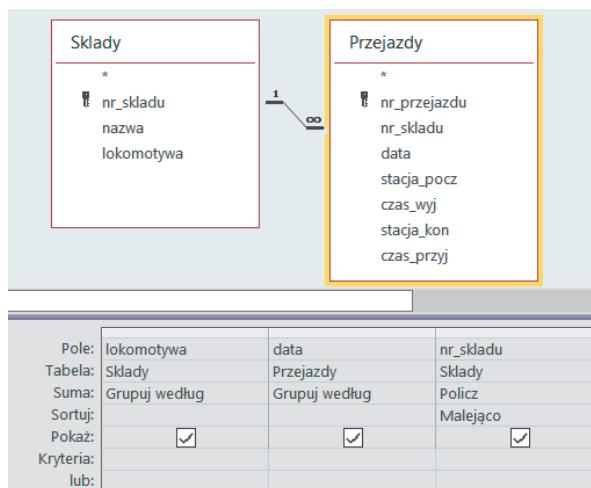


Odpowiedź:

nr\_przejazdu  
507  
508  
510

## Zadanie 5.8.

```
SELECT Skladys.lokomotywa, Przejazdy.data, Count(Sklady.nr_skadu) AS
↳Policz0fnr_skadu
FROM Skladys INNER JOIN Przejazdy ON Skladys.nr_skadu = Przejazdy.nr_skadu
GROUP BY Skladys.lokomotywa, Przejazdy.data
ORDER BY Count(Sklady.nr_skadu) DESC;
```



Odpowiedź:

lokomotywa	data	PoliczOfnr_skladu
EP-09	18.07.2022	2
EP-09	16.07.2022	2
EP-09	15.07.2022	2

# Zestaw 2

## Zadanie 1. Szyfr przestawieniowy

Poniższa rekurencyjna funkcja szyfruj za pomocą pewnego szyfru przestawieniowego z wykorzystaniem klucza szyfruje tekst jawnego i tworzy szyfrogram. Przeanalizuj poniższy kod funkcji, uwzględniając specyfikację:

- w — tekst jawnny (łańcuch znaków — string),
- s — szyfrogram (łańcuch znaków — string),
- m — długość tekstu jawnego i szyfrogramu (dodatnia liczba naturalna),
- k — liczba elementów klucza (dodatnia liczba naturalna),
- klucz[1..k] — klucz (tablica jednowymiarowa, w której każda z liczb naturalnych od 1 do k występuje dokładnie jeden raz).

```
funkcja szyfruj(n)
    jeżeli n+k-1<m
        dla i=1,2,3,...,k wykonuj
            s ← s+w[n+klucz[i]-1]
            szyfruj(n+k)
    w przeciwnym razie
        dopóki n<=m
            jeżeli n<m
                s ← s+w[n+1]+w[n]
                n ← n+2
            w przeciwnym razie
                s ← s+w[n]
                n ← n+1
funkcja szyfruj(n)
    jeżeli n+k-1<m
        dla i=1,2,3,...,k wykonuj
            s ← s+w[n+klucz[i]-1]
            szyfruj(n+k)
    w przeciwnym razie
        dopóki n<=m
            jeżeli n<m
                s ← s+w[n+1]+w[n]
                n ← n+2
```

w przeciwnym razie

$s \leftarrow s + w[n]$

$n \leftarrow n+1$

## Zadanie 1.1.

Pierwsze wywołanie funkcji szyfruj(1). Przed pierwszym wywołaniem funkcji szyfruj zmienna *s* jest pustym łańcuchem znaków. Uzupełnij tabelę:

<i>w</i>	<i>M</i>	<i>k</i>	Klucz	<i>s</i>	Liczba wywołań funkcji szyfruj
KOLORADO	8	3	[2,3,1]		
ANTROPOMORFIZACJA	17	4	[3,4,1,2]		
KOMPUTEREK	10	5	[5,1,4,2,3]		

## Miejsce na obliczenia

## Zadanie 1.2.

Zachowując następujące oznaczenia:

*m* — długość tekstu jawnego,

$k$  — liczba elementów klucza,

napisz wzór na liczbę wywołań funkcji szyfruj w zależności od  $m$  i  $k$ .

Wzór: .....

### Zadanie 1.3.

Podaj przykład tekstu jawnego złożonego z 18 znaków, w którym każda z liter  $a$ ,  $b$ ,  $c$  występuje sześć razy, dla którego szyfrogram utworzony przez funkcję szyfruj z użyciem klucza [3,2,1] jest taki sam jak tekst jawny.

Przykład tekstu jawnego: .....

Miejsce na obliczenia

## Zadanie 1.4.

Zapisz w pseudojęzyku lub wybranym języku programowania **nierekurencyjną** funkcję deszyfrującą, która dokona deszyfrowania.



### 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, odwoływanego się do pojedynczych elementów tablicy i 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.

**Specyfikacja:**

Dane:

$s$  — szyfrogram (łańcuch znaków — string)

$m$  — długość tekstu jawnego i szyfrogramu (dodatnia liczba naturalna)

$k$  — liczba elementów klucza (dodatnia liczba naturalna)

$klucz[1..k]$  — klucz (tablica jednowymiarowa, w której każda z liczb naturalnych od 1 do  $k$  występuje dokładnie jeden raz)

Wynik:

$w$  — tekst jawnny (łańcuch znaków — string)

A large grid of 20 columns and 25 rows, intended for handwritten answers.

## Zadanie 2. Tajemnicze hieroglify

Podczas wykopalisk archeolodzy natrafili na zbiór kilkuset cienkich metalowych tabliczek z interesującym sposobem zapisu informacji. Na tabliczkach do zapisu używano tylko trzech symboli: I, II oraz III. Przekazywana pojedyncza informacja miała postać hieroglifu złożonego z 25 symboli, zapisanych w pięciu kolumnach i w pięciu wierszach.

**Przykład takiego hieroglifu:**

II	I	II	III	I
II	II	I	III	I
II	I	I	II	III
I	I	II	III	I
III	III	III	III	II

W pliku *hieroglify.txt* znajdują się zeskanowane z tabliczek informacje. Symbole w wierszu oddzielone są znakiem tabulacji. Każdy hieroglif oddzielony jest pustym wierszem. Zapisano pięć hieroglifów:

I	III	II	III	II
II	II	III	II	I
I	III	III	II	II
II	II	I	II	II
III	I	II	II	I

I	I	I	I	II
II	I	II	I	III
I	III	II	II	II
I	III	I	I	I
III	I	II	I	III

I	III	II	III	II
II	II	III	II	I
I	III	III	II	II
II	II	I	II	II
III	I	II	II	I

III	I	I	I	II
II	I	II	I	III
I	III	II	II	II
I	III	I	I	I
III	I	II	I	III

III	I	II	III	I
II	II	I	III	I
II	I	I	II	III
I	I	II	III	I
III	III	III	III	II

Tabliczki z hieroglifami miały różne kolory. Pierwszy symbol w pierwszym wierszu w hieroglifie był zawsze wyróżniony. Naukowcy zwracają się do Ciebie z prośbą o napisanie kilku programów, w wybranym przez Ciebie języku programowania, które dadzą odpowiedź na nurtujące ich pytania.

## Zadanie 2.1.

Pierwszy hieroglif w pliku nazywamy hieroglifem głównym. Był on zapisany na złotej tabliczce. Naukowcy przypuszczają, że był najważniejszy. Które z pozostałych w pliku hieroglifów są najbardziej zgodne z hieroglifem głównym? Podaj trzy hieroglify, których procent zgodności symboli na tych samych pozycjach jest największy.

Podaj procent zgodności.

**Przykład:**

*Hieroglif główny*

II	I	I	III	I
I	II	I	I	II
III	II	III	I	II
I	II	I	III	III
I	II	II	I	I

Hieroglif obok jest zgodny z hieroglifem głównym w 28%, ponieważ 7 spośród 25 symboli jest identycznych z symbolami na tych samych pozycjach. Symbole zgodne zostały pogrubione.

I	II	II	<b>III</b>	II
II	<b>III</b>	<b>III</b>	<b>I</b>	I
I	<b>III</b>	<b>III</b>	<b>I</b>	<b>III</b>
II	<b>III</b>	<b>I</b>	II	II
III	<b>II</b>	<b>III</b>	II	<b>I</b>

## Zadanie 2.2.

Czy w pliku *hieroglify.txt* znajdują się dwa takie same hieroglify? W przypadku odpowiedzi twierdzącej wypisz zestawy takich samych hieroglifów.

## Zadanie 2.3.

Hieroglif nazwiemy **prawidłowym**, gdy pierwszy symbol w pierwszym wierszu występuje w hieroglifie najczęściej. Ile hieroglifów prawidłowych znajduje się w pliku *hieroglify.txt*?

Hieroglif obok jest prawidłowy. Pierwszym symbolem w pierwszym wierszu jest III i występuje on 10 razy.

<b>III</b>	II	II	<b>III</b>	II
II	<b>III</b>	<b>III</b>	I	I
I	<b>III</b>	<b>III</b>	I	<b>III</b>
II	<b>III</b>	I	II	II
<b>III</b>	II	<b>III</b>	II	I

## Zadanie 2.4.

Powiemy, że jeden hieroglif jest **permutacją pionową** drugiego, jeżeli mają takie same kolumny znaków, tylko ułożone w innej kolejności. Poniżej znajdują się dwa hieroglify: A i B, gdzie jeden jest permutacją pionową drugiego.

<b>A</b>					<b>B</b>				
I	II	I	II	III	II	III	II	I	I
II	III	I	II	II	II	II	III	I	II
II	II	III	II	III	II	III	II	III	II
I	III	II	I	II	I	II	III	II	I
I	III	I	II	II	II	II	III	I	I

Pierwsza kolumna w hieroglifie A jest taka sama jak piąta kolumna w hieroglifie B, co zapiszymy  $A(1) = B(5)$ , dalej  $A(2) = B(3)$ ,  $A(3) = B(4)$ ,  $A(4) = B(1)$ ,  $A(5) = B(2)$ .

Ile hieroglifów w pliku *hieroglify.txt* jest permutacją pierwszego hieroglifu?

## Zadanie 2.5.

Jeżeli w wierszu wszystkie symbole są takie same, to taki wiersz nazwiemy **jednolitym**. Ile jest hieroglifów, w których występuje jeden wiersz jednolity, a ile takich, w których występują dwa wiersze jednolite?

## Zadanie 3. Temperatura w Nowym Jorku

Pewien mieszkaniec Nowego Jorku postanowił codziennie sprawdzać temperaturę powietrza na swoim balkonie, a wynik obserwacji zapisywać. Stwierdził, że najlepszym dniem na rozpoczęcie odczytywania wartości na termometrze będzie prima aprilis. Niestety wytrwałość starczyło mu na sześć miesięcy. W pliku *temperatury\_nowy\_jork.txt* znajdują się wyniki pomiarów temperatury w okresie od 1 kwietnia 2022 roku do 30 września 2022 roku.

Każdy wiersz w pliku zawiera datę pomiaru i temperaturę w stopniach Celsjusza oddzielone znakiem spacji.

Przykładowy fragment pliku *temperatury\_nowy\_jork.txt*

20.04.2022 10

21.04.2022 11

22.04.2022 15

23.04.2022 13

24.04.2022 12

25.04.2022 11

26.04.2022 11

27.04.2022 10j

28.04.2022 8

29.04.2022 11

30.04.2022 13  
01.05.2022 13  
02.05.2022 11  
03.05.2022 12  
04.05.2022 12

W wybranym przez siebie języku programowania napisz programy, które dadzą odpowiedź na pytania lub wykonają polecenia zamieszczone w poniższych zadaniach.

### Zadanie 3.1.

Ile dni liczył najdłuższy okres, w którym utrzymywała się taka sama temperatura? Ile razy taki okres wystąpił?

*W przykładowym fragmencie pliku najdłuższy taki okres wynosił dwa dni i wystąpił trzy razy: 25.04.2022 – 26.04.2022, 30.04.2022 – 01.05.2022 oraz 03.05.2022 – 04.05.2022.*

### Zadanie 3.2.

W jakim okresie (podaj daty) wystąpił najdłuższy ciąg dni, w którym temperatura w danym dniu nie była niższa niż temperatura w dniu poprzednim?

*W przykładowym fragmencie pliku najdłuższy taki okres wynosił cztery dni i wystąpił jeden raz: 28.04.2022 – 01.05.2022.*

### Zadanie 3.3.

Ile było dni, w których temperatura powietrza była **mniejsza** niż średnia temperatura w całym okresie od 1 kwietnia 2022 do 30 września 2022?

### Zadanie 3.4.

Oblicz średnie temperatury w poszczególne dni tygodnia. Inaczej: jaka była średnia temperatura w poniedziałki w całym okresie pomiarów? A jaka we wtorki? I tak dalej. Pierwszy dzień pomiarów to 1 kwietnia 2022 roku (piątek).

## Zadanie 4. Sprzedaż drewna

W nadleśnictwie Wierzbowo pozyskiwane drewno przeznaczone jest dla przemysłu meblarskiego lub na opał. Dyrekcja nadleśnictwa prowadzi bardzo rozważną politykę wycinków. Dokonuje się odpowiedniego wyboru drzew, a w miejsce wyciętych sadzone są nowe. Ponadto nadleśnictwo kontroluje ilości sprzedanego drewna.

Nadleśnictwo sprzedaje drzewo należące do dwóch klas. Do klasy pierwszej należy drewno przeznaczone dla przemysłu meblarskiego, natomiast do klasy drugiej drewno opałowe uzyskane w wyniku ścinki sanitarnej.

Drewno było sprzedawane w następujących cenach:

Gatunek drewna	Klasa	Cena za 1 m <sup>3</sup>
Sosna	1	450
Sosna	2	250
Akacja	1	550
Akacja	2	400
Buk	1	500
Buk	2	300

W pliku *pozyskane\_drewno.txt* znajdują się informacje o ilości pozyskanego i sprzedanego w nadleśnictwie drewna w okresie od 1 stycznia 2022 roku do 31 grudnia 2022 roku. Każdy wiersz w pliku zawiera następujące informacje oddzielone średnikiem: data sprzedaży, gatunek drewna, klasa i ilość wyrażona w metrach sześciennych.

#### Przykładowy fragment pliku *pozyskane\_drewno.txt*

2022-07-23;sosna;1;47  
2022-07-23;akacja;2;39  
2022-07-23;akacja;1;57  
2022-07-23;buk;1;4  
2022-07-24;akacja;1;35  
2022-07-30;akacja;1;22  
2022-07-31;buk;2;51  
2022-07-31;akacja;2;67  
2022-07-31;buk;1;57

Pozyskiwanie drewna odbywało się w dniach roboczych, natomiast sprzedaż realizowana była w soboty i w niedziele. Z uwagi na ogromne zainteresowanie oraz atrakcyjne ceny całe pozyskane w tygodniu drewno sprzedawano w weekend. W poszczególnych tygodniach pozyskiwano ustalone gatunki drzew i w ustalonych klasach.

Wykorzystując dostępne narzędzia informatyczne i opierając się na danych zawartych w pliku *pozyskane\_drewno.txt*, podaj odpowiedzi do pytań lub wykonaj polecenia zawarte w zadaniach 4.1 – 4.5.

### Zadanie 4.1.

Którego gatunku drzewa sprzedano w ciągu roku **najmniej**? Ile wyniósł przychód z jego sprzedaży?

### Zadanie 4.2.

W którym dniu łączna ilość (liczona od początku roku) sprzedanej sosny przeznaczonej na opał przekroczyła 1000 m<sup>3</sup>?

## Zadanie 4.3.

W który weekend sprzedano najwięcej drewna przeznaczonego do produkcji mebli? W odpowiedzi podaj datę początku weekendu (sobota).

## Zadanie 4.4.

Utwórz zestawienie wartości sprzedanego drewna w poszczególnych miesiącach z uwzględnieniem każdego z gatunków. Na podstawie tego zestawienia utwórz wykres kolumnowy. Pamiętaj o czytelności wykresu.

## Zadanie 4.5.

Założymy, że cena 1 m<sup>3</sup> sosny pierwszej klasy jest większa o 80% od ceny 1 m<sup>3</sup> sosny drugiej klasy. Ceny zaokrąglamy do pełnych złotych. Uwzględniając to założenie, podaj **najniższą** cenę (wyrażoną w pełnych złotych) za jeden metr sześcienny sosny klasy drugiej, przy której roczny przychód ze sprzedaży sosny będzie większy od rocznego przychodu ze sprzedaży buku.

## Zadanie 5. Sanatorium

Sanatorium Pelikan położone jest w uroczej nadmorskiej miejscowości. Zjeżdżają tu kuracjusze z całego kraju. Turnusy w tym sanatorium trwają jeden, dwa lub trzy tygodnie. Pacjenci mogą tu przebywać na podstawie skierowania z Narodowego Funduszu Zdrowia (NFZ) lub, jeżeli nie mają skierowania, mogą wykupić turnus. W dniu przyjazdu pacjenci przybywają do sanatorium w godzinach porannych, a w dniu odjazdu opuszczają sanatorium w godzinach wieczornych. W ten sposób pacjenci korzystają z zabiegów zarówno w dniu przyjazdu, jak i w dniu odjazdu.

W pliku *pacjenci.txt* znajdują się informacje o zgłoszonych pacjentach, którzytrzymali skierowanie z NFZ lub nie mając skierowania, wykupili turnus w sanatorium. Każdy wiersz w pliku zawiera: numer PESEL, imię, nazwisko i województwo, które pacjent zamieszkuje. Dane oddzielone zostały średnikiem.

**Przykładowy fragment pliku *pacjenci.txt*:**

```
72010290115;STEFAN;SZYMCZAK;podlaskie  
86011740579;MIECZYSŁAW;KRUPA;malopolskie  
71020373847;MILENA;SIKORSKA;wielkopolskie  
59031260545;OLIWIA;KAZMIERCZAK;podlaskie  
89051064162;AGATA;SZYMCZAK;slaskie
```

W pliku *pobyt.txt* znajdują się informacje dotyczące pobytów pacjentów, którzy przyjechali do sanatorium. Każdy wiersz w pliku zawiera: identyfikator pobytu, PESEL pacjenta, datę przyjazdu do sanatorium (pierwszy dzień pobytu), czas trwania pobytu w tygodniach, informację o skierowaniu z NFZ (liczba 1, jeśli pacjent został skierowany przez NFZ, liczba 0 w przeciwnym wypadku). Dane oddzielone zostały średnikiem.

## **Przykładowy fragment pliku *pobity.txt*:**

1382;78081144207;2019-06-10;3;0  
1383;90012713883;2019-12-06;1;1  
1386;66120821431;2019-05-07;2;1  
1389;71081957811;2019-01-07;2;1  
1390;60022898913;2019-08-21;1;0

Liczba zabiegów w ciągu dnia, z jakich mogą skorzystać pacjenci bez dodatkowych opłat, uzależniona jest od długości pobytu. Jeżeli pacjent przyjechał na tydzień, to w ramach pakietu podstawowego będzie korzystał z dwóch zabiegów dziennie; jeśli przyjechał na dwa tygodnie, to z trzech zabiegów. Natomiast pacjenci przebywający trzy tygodnie będą korzystali z czterech zabiegów dziennie. Jeżeli pacjent chce skorzystać z większej liczby zabiegów, musi za nie zapłacić. Za każdy jeden dodatkowy zabieg trzeba zapłacić 400 złotych tygodniowo. Dodatkowe zabiegi można wykupić tylko w dniu przyjazdu i realizuje się je przez cały turnus.

W pliku *zabiegi.txt* znajdują się informacje dotyczące zabiegów, z których korzystali kuracjusze. Każdy wiersz w pliku zawiera: identyfikator pobytu i nazwę zabiegu. Dane oddzielone zostały średnikiem.

## **Przykładowy fragment pliku *zabiegi.txt*:**

1246;masaż podwodny  
1290;fasony siarczkowe  
1390;kąpiel solankowa  
1197;krioterapia miejskowa  
1363;kąpiel solankowa

Wykorzystując dostępne narzędzia informatyczne i opierając się na danych zawartych w plikach, podaj odpowiedzi do pytań lub wykonaj polecenia zawarte w zadaniach 5.1 – 5.9.

W przypadku zadań, pod treścią których znajduje się miejsce do zapisu, w pierwszej kolejności napisz zapytanie w języku SQL. Następnie sprawdź jego poprawność, wykorzystując dostępne narzędzia informatyczne.

## Zadanie 5.1.

Wypisz imiona i nazwiska pacjentów z województwa wielkopolskiego, którzy zostali skierowani do sanatorium przez NFZ i przynajmniej jeden z ich pobytów trwał co najmniej dwa tygodnie.

## Zadanie 5.2.

Wypisz numery PESEL, imiona i nazwiska kobiet, które zostały zgłoszone do sanatorium i które urodziły się przed rokiem 1980, ich imię składa się dokładnie z pięciu liter, a nazwisko kończy się literą A.



## **Uwaga**

Pierwsze dwie cyfry numeru PESEL to ostatnie dwie cyfry roku urodzenia. Osoba z numerem **78123027523** urodziła się w roku **1978**. Parzystość przedostatniej cyfry numeru PESEL informuje o płci: w przypadku kobiet jest to cyfra parzysta, w przypadku mężczyzn nieparzysta. Osoba z naszej wskazówka to kobieta — przedostatnia cyfra to cyfra 2.

### Zadanie 5.3.

Ilu pacjentów spośród zgłoszonych do sanatorium nigdy do niego nie przyjechało?

## Zadanie 5.4.

Utwórz zestawienie zawierające imiona i nazwiska pacjentów, którzy spędzili w sanatorium łącznie co najmniej sześć tygodni wykupionych przez siebie — bez skierowania z NFZ. Dane w zestawieniu posortuj rosnąco według liczby tygodni.

## Zadanie 5.5.

Wypisz imiona i nazwiska pacjentów, których cały pobyt w sanatorium miał miejsce we wrześniu 2019 roku, tzn. dzień przyjazdu i dzień odjazdu przypadły we wrześniu 2019 roku.

## Zadanie 5.6.

Podaj imię i nazwisko pacjenta, który zapłacił najwięcej za zabiegi dodatkowe. Podaj także kwotę, jaka wydał na wszystkie zabiegi dodatkowe.

**Zadanie 5.7.**

Z ilu najwięcej różnych zabiegów skorzystał jeden pacjent w ciągu wszystkich swoich pobytów w sanatorium? Ilu było takich pacjentów?

**Zadanie 5.8.**

Utwórz zestawienie średniej liczby dodatkowych zabiegów przypadającej na jeden pobyt pacjenta w poszczególnych latach.

**Zadanie 5.9.**

W przypadku ilu pacjentów, spośród tych, którzy byli w sanatorium co najmniej dwa razy, wszystkie pobuty trwały jeden tydzień?

# Odpowiedzi i wskazówki do zestawu 2

## Zadanie 1.1.

<i>w</i>	<i>M</i>	<i>k</i>	Klucz	<i>s</i>	Liczba wywołań funkcji szyfruj
KOLORADO	8	3	[2,3,1]	OLKRAOOD	3
ANTROPOMORFIZACJA	17	4	[3,4,1,2]	TRANOMOPFIORCJZAA	5
KOMPUTEREK	10	5	[5,1,4,2,3]	UKPOMKTEER	3

## Zadanie 1.2.

Odpowiedź:

Wzór: liczba wywołań =  $(m \text{ div } k) + 1$

## Zadanie 1.3.

Odpowiedź:

Przykładowy tekst jawnny: abaacababbcbcaccbc

## Zadanie 1.4.

Przykładowe rozwiążanie:

```
funkcja deszyfruj()
    p ← m div k
    dla i=1,2,3,...,p wykonuj
        dla j=1,2,3,...,k wykonuj
            z ← 1
            dopóki klucz[z]≠j
                z ← z+1
            w ← w+s[(i-1)*k+z]
        p ← p*k+1
        dopóki p<=m
            jeżeli p<m
                w ← w+s[p+1]+s[p]
                p ← p+2
            w przeciwnym razie
                w ← w+s[p]
                p ← p+1

funkcja deszyfruj()
    p ← m div k
    dla i=1,2,3,...,p wykonuj
        dla j=1,2,3,...,k wykonuj
            z ← 1
            dopóki klucz[z]≠j
                z ← z+1
            w ← w+s[(i-1)*k+z]
        p ← p*k+1
        dopóki p<=m
```

```

jeżeli p<m
    w ← w+s[p+1]+s[p]
    p ← p+2
w przeciwnym razie
    w ← w+s[p]
    p ← p+1

```

## Zadanie 2.1.

```

#include <iostream>
#include <fstream>
using namespace std;
main()
{
int h[5][5];
string z;
ifstream plik;
plik.open("hieroglify.txt");
for(int i=0; i<5; i++)
for(int j=0; j<5; j++)
{
    plik>>z;
    if(z=="I") h[i][j] = 1;
    else if(z=="II") h[i][j] = 2;
    else h[i][j] = 3;
}

int hx[5][5]; // kolejny hieroglif
double zgodny[5];//zerowy pomijamy

for(int n=1; n<=4; n++) // kolejne 4 hieroglify
{
//wczytujemy i konwertujemy kolejny hieroglif
for(int i=0; i<5; i++)
for(int j=0; j<5; j++)
{
    plik>>z;
    if(z=="I") hx[i][j] = 1;
    else if(z=="II") hx[i][j] = 2;
    else hx[i][j] = 3;
}

double licznik = 0; // dla kolejnego zeruje liczbę zgodnych
for(int i=0; i<5; i++)
for(int j=0; j<5; j++)
    if(hx[i][j]==h[i][j]) licznik++;
zgodny[n] = (licznik/25.0)*100.0;
}
for(int i=1; i<=4; i++)
cout<<i+1<<" "<< zgodny[i]<<endl;
plik.close();
}

```

**Odpowiedź:**

2 44

3 100

4 40

## Zadanie 2.2.

```
#include <iostream>
#include <fstream>
using namespace std;
main()
{
    int h[5][5];
    int hx[5][5];
    string z;
    ifstream plik;
    int licznik = 0;
    int n, x, y;
    for(n=1; n<=4; n++) //cztery nawroty otwarcia pliku dla 5 elementów
    {
        plik.open("hieroglify.txt");
        for(x = 1; x<=n; x++)
        {
            //wczytujemy do wzorcowego
            for(int i=0;i<5; i++)
                for(int j=0; j<5; j++)
                {
                    plik>>z;
                    if(z=="I") h[i][j] = 1;
                    else if(z=="II") h[i][j] = 2;
                    else h[i][j] = 3;
                }
        }
        for(y=n+1;y<=5; y++) // kolejny hieroglif
        {
            for(int i=0;i<5; i++)
                for(int j=0; j<5; j++)
                {
                    plik>>z;
                    if(z=="I") hx[i][j] = 1;
                    else if(z=="II") hx[i][j] = 2;
                    else hx[i][j] = 3;
                }
        }
        //sprawdzamy, czy taki sam
        bool flaga = true; // zakładamy, że jest taki sam
        for(int i=0; i<5; i++)
            for(int j=0; j<5; j++)
                if(h[i][j] != hx[i][j]) flaga = false;
        if(flag == true)
        {
            licznik++;
            cout<<n<< " "<<y<<endl;
        }
    }

    plik.close();
}
cout<<"zgodnych = "<<licznik;
```

**Odpowiedź:**

1 3

Zgodnych: 1

## Zadanie 2.3.

```
#include <iostream>
#include <fstream>
using namespace std;
main()
{
    int h[5][5];
    string z;
    ifstream plik;
    int licznik = 0; //liczba prawidłowych
    plik.open("hieroglify.txt");
    int liczniki[4];
    for(int n=1; n<=5; n++)
    {
        for(int x=0; x<4; x++)
            liczniki[x] = 0; //zerowanie liczników
        for(int i=0; i<5; i++)
            for(int j=0; j<5; j++)
            {
                plik>>z;
                if(z=="I") h[i][j] = 1;
                else if(z=="II") h[i][j] = 2;
                else h[i][j] = 3;

                switch(h[i][j])
                {
                    case 1:liczniki[1]++; break;
                    case 2:liczniki[2]++; break;
                    case 3:liczniki[3]++; break;
                }
            }
        if(h[0][0]==1 && liczniki[1]>liczniki[2] && liczniki[1]>liczniki[3]) licznik++;
        if(h[0][0]==2 && liczniki[2]>liczniki[1] && liczniki[2]>liczniki[3]) licznik++;
        if(h[0][0]==3 && liczniki[3]>liczniki[1] && liczniki[3]>liczniki[2]) licznik++;
    }
    plik.close();
    cout<<licznik;
}
```

**Odpowiedź:**

2

## Zadanie 2.4.

```
#include <iostream>
#include <fstream>
using namespace std;
main()
{
    string h[5][5];
    string hx[5][5];
    ifstream plik;
    int licznik_perm=0;
    plik.open("hieroglify.txt");
    //wczytuje do pozycji wzorcowego
```

```
for(int i=0; i<5; i++)
    for(int j=0; j<5; j++)
        plik>>h[i][j];
    for(int x=1; x<=4; x++)
    {
        for(int i=0; i<5; i++)
            for(int j=0; j<5; j++)
                plik>>hx[i][j];
    int liczba_zgodnych=0;
    for(int n=0; n<5; n++) //kolumny wzorcowego hieroglifu
    {
        for(int m=0; m<5; m++) //kolumny kolejnego hieroglifu
        {
            int licznik = 0;
            for(int i=0; i<5; i++) //porównanie wierszy
                if(hx[i][n]==h[i][m]) licznik++;
            if(licznik==5)
            {
                liczba_zgodnych++;
                hx[0][n]='X'; //oznaczenie zgodnej kolumny
            }
        }
    }
    if(liczba_zgodnych==5) licznik_perm++;
}
cout<<licznik_perm<<endl;
plik.close();
}
```

**Odpowiedź:**

1

## Zadanie 2.5.

```
#include <iostream>
#include <fstream>
using namespace std;
main()
{
    string h[5][5];
    ifstream plik;
    int jedn1=0, jedn2 = 0;
    plik.open("hieroglify.txt");
    for(int n=1; n<=5; n++)
    {
        for(int i=0; i<5; i++)
            for(int j=0; j<5; j++)
                plik>>h[i][j];
        int wierszyjednolitych = 0;
        for(int i=0; i<5; i++)
        if(h[i][0]==h[i][1] && h[i][0]==h[i][2] && h[i][0]==h[i][3] && h[i][0]==h[i][4])
            wierszyjednolitych++;
        if(wierszyjednolitych==1) jedn1++;
        if(wierszyjednolitych==2) jedn2++;
    }
    cout<<jedn1<<endl;
```

```
cout<<jedn2<<endl;
plik.close();
}
```

**Odpowiedź:**

0 i 0. Otwórz plik i dokonaj modyfikacji, aby otrzymać inne wyniki.

**Zadanie 3.1.**

```
#include <iostream>
#include <fstream>
using namespace std;
main()
{
    ifstream plik;
    plik.open("temperatury_nowy_jork.txt");
    string data="";
    int temp, nastepna;
    int tasama=0, max_tasama=0, max_ilie=0;

    plik>>data;
    plik>>temp;
    tasama = 1;
    while(!plik.eof())
    {
        plik>>data;
        plik>>nastepna;
        if(temp==nastepna)
        {
            tasama++;

            if(tasama == max_tasama)
                max_ilie++;
            if(tasama>max_tasama)
            {
                max_tasama = tasama;
                max_ilie=1;           //nowy max, więc liczba najdłuższych okresów na 1
            }
        }
        else tasama = 1;
        temp = nastepna;
    }
    cout<<max_tasama<<" "<<max_ilie;
    plik.close();
}
```

**Odpowiedź:**

4 i 1

**Zadanie 3.2.**

```
#include <iostream>
#include <fstream>
using namespace std;
main()
{
```

```
ifstream plik;
plik.open("temperatury_nowy_jork.txt");
string data="";
int temp, nastepna;
int niemalejaca=0, max_niemalejaca=0;
string data_od_do, max_data_od_do;
plik>>data;
plik>>temp;
niemalejaca = 1;
data_od_do = data;
while(!plik.eof())
{
    plik>>data;
    plik>>nastepna;
    if(nastepna>=temp)
    {
        niemalejaca++;
        data_od_do += "\n";
        data_od_do += data;
        if(max_niemalejaca<niemalejaca)
        {
            max_niemalejaca = niemalejaca;
            max_data_od_do = data_od_do;
        }
    }
    else
    {
        niemalejaca = 1;
        data_od_do = data;
    }
    temp = nastepna;
}
cout<<max_data_od_do <<endl <<max_niemalejaca;
plik.close();
}
```

**Odpowiedź:**

1.08.2022

2.08.2022

3.08.2022

4.08.2022

5.08.2022

6.08.2022

7.08.2022

8.08.2022

9.08.2022

Zatem 9.

### Zadanie 3.3.

```
#include <iostream>
#include <fstream>
using namespace std;
main()
{
    ifstream plik;
    plik.open("temperatury_nowy_jork.txt");
    string data;
    double temp;
    int licznik = 0;
    double l_temp = 0;
    double s_temp = 0;
    while(!plik.eof())
    {
        plik>>data;
        plik>>temp;
        l_temp++;
        s_temp +=temp;
    }
    plik.close();
    double srednia = s_temp/l_temp;
    plik.open("temperatury_nowy_jork.txt");
    while(!plik.eof())
    {
        plik>>data;
        plik>>temp;
        if(temp<srednia)
            licznik++;
    }
    plik.close();
    cout<<licznik;
}
```

**Odpowiedź:**

73

### Zadanie 3.4.

```
#include <iostream>
#include <fstream>
using namespace std;
main()
{
    ifstream plik;
    plik.open("temperatury_nowy_jork.txt");
    string data="";
    int temp;
    double suma_temp[7]={0,0,0,0,0,0,0};
    double liczba_temp[7]={0,0,0,0,0,0,0};
    //reszta z dzielenia przez 7, kolejny dzień, czyli niedziela, indeks 0
    int licznik = 4; //liczy kolejną przeczytaną temperaturę
    while(!plik.eof())
    {
```

```

plik>>data;
plik>>temp;
licznik++;
int dzien = licznik%7; //od 0 do 6
suma_temp[dzien] +=temp;
liczba_temp[dzien]++;
}
for(int i=1; i<=6; i++)
cout<<suma_temp[i]/liczba_temp[i]<<endl;
cout<<suma_temp[0]/liczba_temp[0]<<endl;
plik.close();
}

```

**Odpowiedź:**

20.6538

20.9615

20.3846

20.6923

20.3333

20.7308

20.5385

**Zadanie 4.1.**

The screenshot shows a Microsoft Excel spreadsheet with data in columns A through O. The first few rows contain labels like 'Etykiety wierszy' and 'Suma z ilością w m3'. The data starts with 'akacja' in row 4, followed by 'druk' in row 5, and 'sosna' in row 6. Row 7 is a summary row labeled 'Suma kórciowa' with the value '9786'. The data continues with various values for each tree type across different categories. On the right side of the screen, there is a PowerPivot ribbon with several sections: 'Wybierz pola, które chcesz dodać do raportu' (Select fields to add to report), 'Wyszukaj' (Search), and a list of checked fields: 'data sprzedaży' (checked), 'gatunek' (checked), 'klaś' (unchecked), and 'ilość w m3' (checked). Below these are buttons for 'Więcej tabel...' (More tables...), 'Przeciągnij pola między obszarami poniżej:' (Drag fields between areas below), 'Filtry' (Filters), 'Kolumny' (Columns), 'Wiersze' (Rows), and 'Wartości' (Values). At the bottom, there are dropdown menus for 'gatunek' and 'Suma z ilością w m3'.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1				gatunek	klaś	ilość w m3		z klasą 1		gatunek		ilość w m3		
2				akacja	1	10	767 250,00 zł		akacja	2	53			
3	Etykiety wierszy	T Suma z ilością w m3		akacja	1	9		z klasą 2	akacja	2	16			
4		2927		akacja	1	38	612 800,00 zł		akacja	2	54			
5	druk	3291		akacja	1	35		razem	akacja	2	16			
6	sosna	3568		akacja	1	5	1 380 050,00 zł		akacja	2	38			
7	Suma kórciowa	9786		akacja	1	60			akacja	2	20			
8				akacja	1	22			akacja	2	31			
9				akacja	1	67			akacja	2	19			
10				akacja	1	39			akacja	2	58			
11				akacja	1	62			akacja	2	60			
12				akacja	1	34			akacja	2	62			
13				akacja	1	48			akacja	2	29			
14				akacja	1	36			akacja	2	49			
15				akacja	1	57			akacja	2	63			
16				akacja	1	4			akacja	2	22			
17				akacja	1	24			akacja	2	16			
18				akacja	1	24			akacja	2	38			
19				akacja	1	37			akacja	2	68			
20				akacja	1	11			akacja	2	11			
21				akacja	1	20			akacja	2	7			
22				akacja	1	9			akacja	2	31			
23				akacja	1	14			akacja	2	56			
24				akacja	1	33			akacja	2	26			
25				akacja	1	21			akacja	2	4			
26				akacja	1	57			akacja	2	39			
27				akacja	1	35			akacja	2	67			
28														

**Odpowiedź:**

Akacja, 2927

## Zadanie 4.2.

	A	B	C	D	E	F
4	09.01.2022	sosna	2	37	97	
5	23.01.2022	sosna	2	29	126	
6	29.01.2022	sosna	2	6	132	
7	30.01.2022	sosna	2	63	195	
8	19.02.2022	sosna	2	29	224	
9	20.02.2022	sosna	2	8	232	
10	26.02.2022	sosna	2	18	250	
11	12.03.2022	sosna	2	14	264	
12	19.03.2022	sosna	2	68	332	
13	27.03.2022	sosna	2	6	338	
14	03.04.2022	sosna	2	26	364	
15	09.04.2022	sosna	2	59	423	
16	10.04.2022	sosna	2	48	471	
17	24.04.2022	sosna	2	26	497	
18	30.04.2022	sosna	2	7	504	
19	07.05.2022	sosna	2	30	534	
20	08.05.2022	sosna	2	68	602	
21	14.05.2022	sosna	2	13	615	
22	29.05.2022	sosna	2	54	669	
23	05.06.2022	sosna	2	49	718	
24	12.06.2022	sosna	2	36	754	
25	19.06.2022	sosna	2	56	810	
26	03.07.2022	sosna	2	39	849	
27	10.07.2022	sosna	2	56	905	
28	16.07.2022	sosna	2	49	954	
29	14.08.2022	sosna	2	58	1012	
30	21.08.2022	sosna	2	15	1027	
31	28.08.2022	sosna	2	68	1095	

Odpowiedź:

14.08.2022

## Zadanie 4.3.

Wygenerowanie kolejnego weekendu z przefiltrowanych danych:

	A	B	C	D	E	F
1	data sprzedaży	gatunek	klasa	ilość w m3	kolejny weekend	
2	01.01.2022	buk	1	17		
3	02.01.2022	buk	1	50	=JEŻELI((A3-A2>1;E2+1;E2))	
4	02.01.2022	akacja	1	10	1	
5	08.01.2022	sosna	1	41	2	
6	08.01.2022	akacja	1	9	2	
7	08.01.2022	buk	1	7	2	
8	09.01.2022	buk	1	64	2	
9	09.01.2022	akacja	1	9	2	
10	15.01.2022	sosna	1	63	3	
11	16.01.2022	sosna	1	47	3	
12	16.01.2022	akacja	1	38	3	

### Odpowiedź:

37,211

## Zadanie 4.4.

Obliczamy wartości z podziałem na klasy:

Łaczmy analizowane wartości obu klas w jedną listę i na jej podstawie tworzymy tabelę przestawną i wykres:

A B C D E F G H I J K L M N

**Suma z wartością**

Etykiety kolumn ▾ Etykiety wierszy ▾ akacja buk sosna Suma końcowa

	akacja	buk	sosna	Suma końcowa
1	106 350,00 zł	131 600,00 zł	127 050,00 zł	365 000,00 zł
2	77 450,00 zł	110 700,00 zł	76 750,00 zł	264 900,00 zł
3	135 600,00 zł	95 600,00 zł	50 350,00 zł	281 550,00 zł
4	178 450,00 zł	39 700,00 zł	133 750,00 zł	351 900,00 zł
5	93 200,00 zł	173 500,00 zł	89 400,00 zł	356 100,00 zł
6	65 550,00 zł	170 800,00 zł	134 250,00 zł	379 600,00 zł
7	189 300,00 zł	111 900,00 zł	119 250,00 zł	420 450,00 zł
8	109 200,00 zł	111 900,00 zł	80 700,00 zł	301 800,00 zł
9	79 050,00 zł	147 300,00 zł	119 400,00 zł	345 750,00 zł
10	154 350,00 zł	132 800,00 zł	109 000,00 zł	396 150,00 zł
11	79 250,00 zł	54 400,00 zł	147 500,00 zł	281 150,00 zł
12	112 300,00 zł	84 700,00 zł	82 600,00 zł	279 600,00 zł
17 Suma końcowa	1380050	1373900	1270000	4023950

Suma z wartością ▾ wartość sprzedaży w poszczególnych miesiącach z podziałem na gatunek

gatunek ▾

akacja buk sosna

miesiąc ▾

Pola tabeli przestawnej

Wybierz pole, które chcesz dodać do raportu:

Wyszukaj

gatunek  klasa  w m3  cena  wartość  miesiąc

Więcej tabel..

Przeciągnij pole między obszarami poniżej:

Filtry Kolumny

gatunek

Wiersze Wartości

miesiąc Suma z wartością

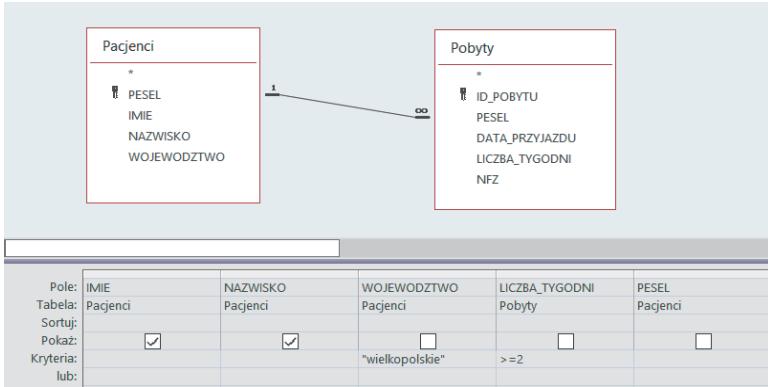
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
id gatunek		klasa	ilość w m3	cena	wartość	miesiąc	data sprzed gatunek	klasa	ilość w m3	cena	wartość	miesiąc	gatunek	klasa	ilość w m3	cena	wartość	miesiąc			
2	2 buk	1	17	500	8500	1	01.01.2022 akacja	2	53	400	21200	1	buk	1	17	500	8500	1			
3	2 buk	1	50	500	25000	1	02.01.2022 akacja	2	16	400	6400	1	buk	1	50	500	25000	1			
4	2 buk	1	10	500	5000	1	02.01.2022 sosna	2	41	250	10250	1	akacja	1	10	550	5500	1			
5	2 sosna	1	41	450	18450	1	03.01.2022 sosna	2	19	250	4750	1	sosna	1	41	450	18450	1			
6	2 akacja	1	9	550	4950	1	09.01.2022 sosna	2	37	250	9250	1	akacja	1	9	550	4950	1			
7	2 buk	1	7	500	3500	1	15.01.2022 buk	2	66	300	19800	1	buk	1	7	500	3500	1			
8	2 buk	1	64	500	32000	1	16.01.2022 akacja	2	54	400	21600	1	buk	1	64	500	32000	1			
9	2 akacja	1	9	550	4950	1	22.01.2022 akacija	2	4	400	1600	1	akacja	1	9	550	4950	1			
10	2 sosna	1	63	450	28350	1	23.01.2022 buk	2	56	300	16800	1	sosna	1	63	450	28350	1			
11	2 sosna	1	47	450	21150	1	23.01.2022 sosna	2	29	250	7250	1	sosna	1	47	450	21150	1			
12	2 akacja	1	38	500	20900	1	29.01.2022 sosna	2	6	250	1500	1	akacja	1	38	500	20900	1			
13	2 sosna	1	23	450	10350	1	30.01.2022 sosna	2	63	250	15750	1	sosna	1	23	450	10350	1			
14	2 buk	1	52	500	26000	1	06.02.2022 buk	2	15	300	4500	2	buk	1	52	500	26000	1			
15	2 akacja	1	35	550	19250	1	06.02.2022 akacija	2	16	400	6400	2	akacja	1	35	550	19250	1			
16	2 sosna	1	30	400	12000	2	10.02.2022 sosna	2	29	250	7250	2	sosna	1	30	450	13500	2			
17	2 akacja	1	5	500	2750	2	20.02.2022 akacia	2	8	250	2000	2	akacia	1	5	500	2750	2			
18	2 sosna	1	54	450	24300	2	20.02.2022 akacija	2	38	400	15200	2	sosna	1	54	450	24300	2			
19	2 akacja	1	60	550	33000	2	20.02.2022 buk	2	38	300	11400	2	akacja	1	60	550	33000	2			
20	2 buk	1	56	500	28000	2	26.02.2022 sosna	2	18	250	4500	2	buk	1	56	500	28000	2			
21	2 sosna	1	28	450	12600	2	27.02.2022 akacija	2	20	400	8000	2	sosna	1	28	450	12600	2			
22	2 buk	1	46	500	23000	2	27.02.2022 buk	2	46	300	13800	2	buk	1	46	500	23000	2			
23	2 buk	1	60	500	30000	2	12.03.2022 sosna	2	14	250	3500	3	buk	1	60	500	30000	2			
24	2 sosna	1	28	450	12600	2	13.03.2022 buk	2	22	300	6600	3	sosna	1	28	450	12600	2			

## Zadanie 4.5.

Do samodzielnego rozwiązania.

## Zadanie 5.1.

```
SELECT Pacjenci.IMIE, Pacjenci.NAZWISKO
FROM Pacjenci INNER JOIN Pobuty ON Pacjenci.PESEL = Pobuty.PESEL
WHERE ((Pacjenci.WOJEWODZTWO)="wielkopolskie") AND ((Pobuty.LICZBA_TYGODNI)>=2);
```



Odpowiedź:

Imię	Nazwisko
AGATA	ZIOLKOWSKA
KAZIMIERA	LASKOWSKA

## Zadanie 5.2.

```
SELECT Pacjenci.IMIE, Pacjenci.NAZWISKO, Pacjenci.PESEL
FROM Pacjenci
WHERE (((Left([PESEL],2)<80) AND ((Mid([pesel],10,1) Mod 2)=0))
AND ((Pacjenci.NAZWISKO) Like "*A") AND ((Len([imie]))=5));
```

The screenshot shows the Microsoft Access query builder interface. At the top, there is a diagram of the 'Pacjenci' table with fields: PESEL, IMIE, NAZWISKO, and WOJEWODZTWO. Below the diagram is a query grid:

Pole:	IMIE	NAZWISKO	PESEL	Left([PESEL];2)	Mid([pesel];10;1) Mod 2	Len([imie])
Tabela:	Pacjenci	Pacjenci	Pacjenci			
Sortuj:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<80	<input type="checkbox"/>	<input type="checkbox"/>
Pokaz:						
Kryteria:		Like "A"			0	5
lub:						

Odpowiedź:

Imię	Nazwisko	PESEL
HANNA	JASINSKA	55050588767
AGATA	ZIOLKOWSKA	61120159427
ANETA	NOWAKOWSKA	55111311282
LUCJA	LEWANDOWSKA	58062467060

## Zadanie 5.3.

```
SELECT Pacjenci.PESEL
FROM Pacjenci LEFT JOIN Pobuty ON Pacjenci.PESEL = Pobuty.PESEL
WHERE ((Pobuty.ID_POBYTU) Is Null);
```

The screenshot shows the Microsoft Access query builder interface. It includes a relationship diagram where the 'Pobuty' table is joined to the 'Pacjenci' table on the 'PESEL' field. A '1' is shown near the 'Pacjenci' side of the relationship line, indicating a one-to-many join. Below the diagram is a query grid:

Pole:	PESEL	ID_POBYTU				
Tabela:	Pacjenci	Pobuty				
Sortuj:	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
Pokaz:		Is Null				
Kryteria:						
lub:						

A 'Właściwości sprzężenia' (Join Properties) dialog box is open on the right, showing the following settings:

- Nazwa tabeli po lewej: Pacjenci
- Nazwa tabeli po prawej: Pobuty
- Nazwa kolumny po lewej: PESEL
- Nazwa kolumny po prawej: PESEL

Three radio button options are listed:

- Uwzględnia tylko te wiersze, w których sprzężone pola z obu tabel są równe.
- Uwzględnia WSZYSTKIE rekordy z „Pacjenci” i tylko te rekordy z „Pobuty”, dla których sprzężone pola są równe.
- Uwzględnia WSZYSTKIE rekordy z „Pobuty” i tylko te rekordy z „Pacjenci”, dla których sprzężone pola są równe.

Buttons at the bottom of the dialog: OK, Anuluj, Nowe.

Odpowiedź:

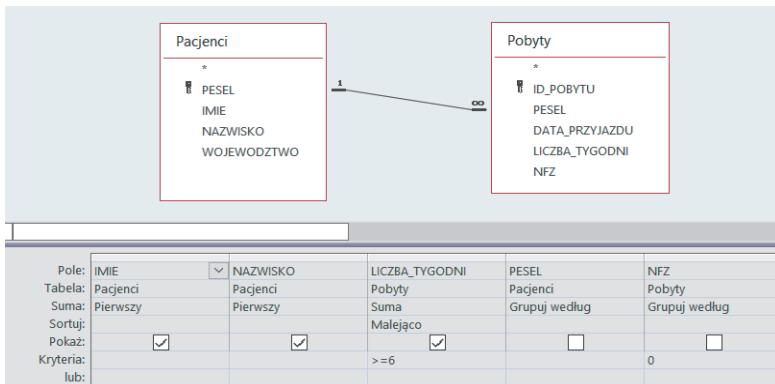
8 pacjentów.

## Zadanie 5.4.

```

SELECT First(Pacjenci.IMIE) AS PierwszyOfIMIE,
       First(Pacjenci.NAZWISKO) AS PierwszyOfNAZWISKO,
       Sum(Pobuty.LICZBA_TYGODNI) AS SumaOfLICZBA_TYGODNI
  FROM Pacjenci INNER JOIN Pobuty ON Pacjenci.PESEL = Pobuty.PESEL
 GROUP BY Pacjenci.PESEL, Pobuty.NFZ
 HAVING (((Sum(Pobuty.LICZBA_TYGODNI))>=6) AND ((Pobuty.NFZ)=0))
 ORDER BY Sum(Pobuty.LICZBA_TYGODNI) DESC;

```

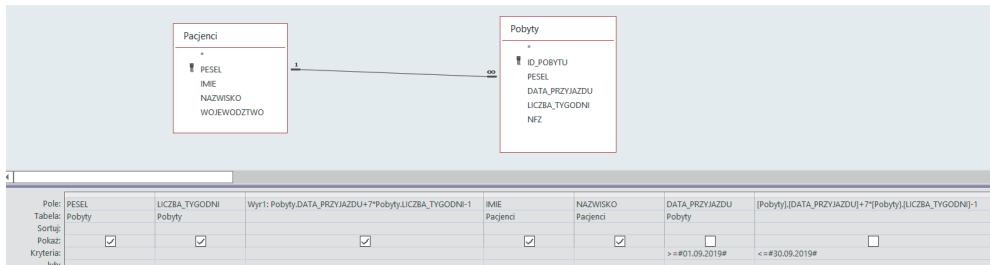


Odpowiedź:

Imię	Nazwisko	Liczba tygodni
ARTUR	BRZEZINSKI	11
ALEKSANDRA	KRUPA	9
NATALIA	WROBEL	8
EUGENIA	SOKOLOWSKA	7
ELZBIETA	URBANSKA	7
ROBERT	WOZNIAK	6

## Zadanie 5.5.

```
SELECT Pobuty.PESEL, Pobuty.LICZBA_TYGODNI,
       Pobuty.DATA_PRZYJAZDU+7*Pobuty.LICZBA_TYGODNI-1 AS Wyr1,
       Pacjenci.IMIE, Pacjenci.NAZWISKO
  FROM Pacjenci INNER JOIN Pobuty ON Pacjenci.PESEL = Pobuty.PESEL
 WHERE ((Pobuty.DATA_PRZYJAZDU)>=#9/1/2019#)
   AND ([Pobuty].[DATA_PRZYJAZDU]+7*[Pobuty].[LICZBA_TYGODNI]-1)<=#9/30/2019#);
```



Odpowiedź:

Imię	Nazwisko
KRYSTIAN	LEWANDOWSKI
LUKASZ	SOBCZAK
FILIP	ZAJAC
BOZENA	WYSOCKA
MIROSLAW	SIKORA
GABRIELA	SZCZEPANIAK
JAKUB	GAJEWSKI
ANGELIKA	OLSZEWSKA

## Zadanie 5.6.

Kwerenda pomocnicza zapisana jako zad\_5\_6\_a:

```
SELECT Zabiegi.ID_POBYTU, First(Pacjenci.PESEL) AS PierwszyOfPESEL,
       First(Pacjenci.IMIE) AS PierwszyOfIMIE, First(Pacjenci.NAZWISKO) AS
       ↳ PierwszyOfNAZWISKO,
       Count(Zabiegi.Identyfikator)-First(Pobuty.LICZBA_TYGODNI)-1 AS [liczba
       dodatkowych
       ↳ zabiegow], First(Pobuty.LICZBA_TYGODNI) AS [liczba tygodni],
       (Count(Zabiegi.Identyfikator)-First(Pobuty.LICZBA_TYGODNI)-1)*
       ↳ First(Pobuty.LICZBA_TYGODNI)*400 AS Koszt
  FROM (Pacjenci INNER JOIN Pobuty ON Pacjenci.PESEL = Pobuty.PESEL) INNER JOIN
       ↳ Zabiegi ON Pobuty.ID_POBYTU = Zabiegi.ID_POBYTU
  GROUP BY Zabiegi.ID_POBYTU
  HAVING ((Count([Zabiegi].[Identyfikator])-First([Pobuty].[LICZBA_TYGODNI])
       ↳ -1)>0))
 ORDER BY Count(Zabiegi.Identyfikator)-First(Pobuty.LICZBA_TYGODNI)-1 DESC;
```

```

SELECT zad_5_6_a.PierwszyOfPESEL, First(zad_5_6_a.PierwszyOfIMIE) AS
→ PierwszyOfPierwszyOfIMIE, First(zad_5_6_a.PierwszyOfNAZWISKO) AS
→ PierwszyOfPierwszyOfNAZWISKO, Sum(zad_5_6_a.Koszt) AS SumaOfKoszt
FROM zad_5_6_a
GROUP BY zad_5_6_a.PierwszyOfPESEL
ORDER BY Sum(zad_5_6_a.Koszt) DESC;

```

The screenshot shows a database query builder interface. At the top, there is a table definition for 'zad\_5\_6\_a' with columns: \* (highlighted in pink), ID\_POBYTU, PierwszyOfPESEL, PierwszyOfIMIE, PierwszyOfNAZWISKO, liczba dodatkowych zabiegow, liczba tygodni, and Koszt. Below this is a large empty table area. At the bottom, there is a configuration panel with the following settings:

Pole:	PierwszyOfPESEL	PierwszyOfIMIE	PierwszyOfNAZWISKO	Koszt
Tabela:	zad_5_6_a	zad_5_6_a	Pierwszy	zad_5_6_a
Suma:	Grupuj wedlug	Pierwszy	Pierwszy	Suma
Sortuj:				Malejaco
Pokaż:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kryteria:				
lub:				

**Odpowiedź:**

NATALIA WROBEL, 4800 złotych.

## Zadanie 5.7.

**Kwerenda pomocnicza zapisana jako zad\_5\_7\_a:**

```

SELECT Pacjenci.PESEL, First(Pacjenci.IMIE) AS PierwszyOfIMIE,
First(Pacjenci.NAZWISKO) AS PierwszyOfNAZWISKO, Zabiegi.ZABIEG,
Count(Zabiegi.Identyfikator) AS PoliczOfIdentyfikator
FROM (Pacjenci INNER JOIN Pobuty ON Pacjenci.PESEL = Pobuty.PESEL) INNER JOIN
Zabiegi ON Pobuty.ID_POBYTU = Zabiegi.ID_POBYTU
GROUP BY Pacjenci.PESEL, Zabiegi.ZABIEG
ORDER BY Count(Zabiegi.Identyfikator) DESC;

SELECT zad_5_7_a.PESEL, First(zad_5_7_a.PierwszyOfIMIE) AS
PierwszyOfPierwszyOfIMIE, First(zad_5_7_a.PierwszyOfNAZWISKO) AS
PierwszyOfPierwszyOfNAZWISKO, Count(zad_5_7_a.ZABIEG) AS PoliczOfZABIEG
FROM zad_5_7_a
GROUP BY zad_5_7_a.PESEL
ORDER BY Count(zad_5_7_a.ZABIEG) DESC;

```

zad\_5\_7\_a

\*

PESEL  
PierwszyOfIMIE  
PierwszyOfNAZWISKO  
ZABIEG  
PoliczOfIdentyfikator

Pole:	PESEL	<input type="button" value="▼"/>	PierwszyOfIMIE	PierwszyOfNAZWISKO	ZABIEG
Tabela:	zad_5_7_a		zad_5_7_a	zad_5_7_a	zad_5_7_a
Suma:	Grupuj według		Pierwszy	Pierwszy	Policz
Sortuj:					Malejaco
Pokaz:	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kryteria:					
lub:					

**Odpowiedź:**

Dwóch pacjentów skorzystało z 13 różnych zabiegów.

**Zadanie 5.8.****Kwerenda pomocnicza zapisana jako zad\_5\_8\_a:**

```

SELECT Zabiegi.ID_POBYTU, First(Right([DATA_PRZYJAZDU],4)) AS rok,
    ↳First(Pacjenci.PESEL) AS PierwszyOfPESEL, First(Pacjenci.IMIE) AS
    ↳PierwszyOfIMIE, First(Pacjenci.NAZWISKO) AS PierwszyOfNAZWISKO,
    ↳Count(Zabiegi.Identyfikator)-First(Pobyty.LICZBA_TYGODNI)-1 AS [liczba
    ↳dodatkowych zabiegow]
FROM (Pacjenci INNER JOIN Pobyty ON Pacjenci.PESEL = Pobyty.PESEL) INNER JOIN
    ↳Zabiegi ON Pobyty.ID_POBYTU = Zabiegi.ID_POBYTU
GROUP BY Zabiegi.ID_POBYTU
ORDER BY Count(Zabiegi.Identyfikator)-First(Pobyty.LICZBA_TYGODNI)-1 DESC;

```

```

SELECT zad_5_8_a.rok, Round(Avg(zad_5_8_a.[liczba dodatkowych zabiegow]),4) AS
    ↳[Średnia liczba dodatkowych zabiegow]
FROM zad_5_8_a
GROUP BY zad_5_8_a.rok;

```

**zad\_5\_8\_a**

\*

ID\_POBYTU  
rok  
PierwszyOFIMIE  
PierwszyOFNAZWISKO  
Ilosc dodatkowych zabiegow

Pole: rok Średnia Ilosc dodatkowych zabiegow: Round(Srednia(zad\_5\_8\_a.Ilosc dodatkowych zabiegow)),4)

Tabela: zad\_5\_8\_a Grupuj wedlug Wyrazenie

Suma:

Sortuj:

Pokaz:

Kryteria: lub:

**Odpowiedź:**

Rok	Średnia
2019	0,5640
2020	0,3871
2021	0,4762
2022	0,3333

## Zadanie 5.9.

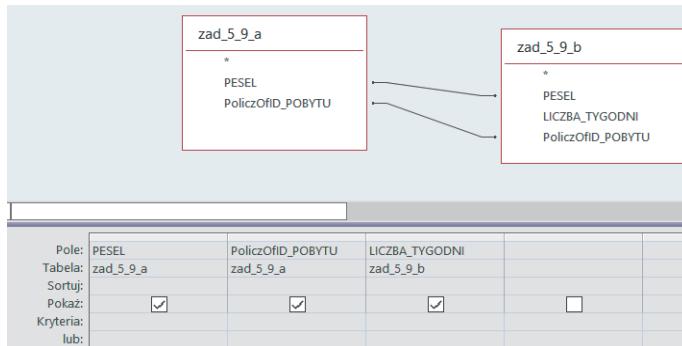
**Kwerenda pomocnicza zapisana jako zad\_5\_9\_a:**

```
SELECT Pacjenci.PESEL, Count(Pobyty.ID_POBYTU) AS PoliczOfID_POBYTU
FROM Pacjenci INNER JOIN Pobyty ON Pacjenci.PESEL = Pobyty.PESEL
GROUP BY Pacjenci.PESEL
HAVING (((Count(Pobyty.ID_POBYTU))>1));
```

**Kwerenda pomocnicza zapisana jako zad\_5\_9\_b:**

```
SELECT Pacjenci.PESEL, Pobyty.LICZBA_TYGODNI, Count(Pobyty.ID_POBYTU) AS
→PoliczOfID_POBYTU
FROM Pacjenci INNER JOIN Pobyty ON Pacjenci.PESEL = Pobyty.PESEL
GROUP BY Pacjenci.PESEL, Pobyty.LICZBA_TYGODNI
HAVING (((Pobyty.LICZBA_TYGODNI)=1));
```

```
SELECT zad_5_9_a.PESEL, zad_5_9_a.PoliczOfID_POBYTU, zad_5_9_b.LICZBA_TYGODNI
FROM zad_5_9_a INNER JOIN zad_5_9_b ON (zad_5_9_a.PoliczOfID_POBYTU =
→zad_5_9_b.PoliczOfID_POBYTU) AND (zad_5_9_a.PESEL = zad_5_9_b.PESEL);
```

**Odpowiedź:**

W przypadku 3 pacjentów.

# Zestaw 3

## Zadanie 1. Funkcja inkrementująca

Operacją najczęściej wykonywaną w funkcji `wyznacz(n)`, gdzie  $n$  jest dodatnią liczbą naturalną, jest operacja inkrementacji, czyli zwiększania wartości zmiennej o 1. Wnikliwa analiza pseudokodu tej funkcji, zaprezentowanego poniżej, pozwoli Ci odpowiedzieć na pytania lub wykonać polecenia zawarte w zadaniach 1.1 – 1.4.

Zmienne  $i, j, n, x, y$  są zmiennymi typu całkowitego.

```
funkcja wyznacz(n)
    x ← 0
    y ← 0
    dla i=1,2,3,...,n wykonuj
        x ← x+1
        dla j=1,2,3,...,i wykonuj
            y ← y+1
    zwróć x+y

funkcja wyznacz(n)
    x ← 0
    y ← 0
    dla i=1,2,3,...,n wykonuj
        x ← x+1
        dla j=1,2,3,...,i wykonuj
            y ← y+1
    zwróć x+y
```

## Zadanie 1.1.

Uzupełnij tabelę:

<b><i>n</i></b>	<b>Wynik zwrócony przez funkcję <code>wyznacz(n)</code></b>
1	2
2	
3	
4	
5	

Miejsce na obliczenia

A large grid of empty squares, approximately 10 columns by 20 rows, intended for students to perform their calculations.

## Zadanie 1.2.

Napisz funkcję rekurencyjną `wyznacz_rek(n)`, która dla podanego  $n$  zwraca tę samą wartość co funkcja `wyznacz(n)`. Określ liczbę wywołań napisanej przez Ciebie funkcji `wyznacz_rek(n)` w zależności od  $n$ .

### Zadanie 1.3.

Podaj wzór na liczbę wykonanych przez funkcję `wyznacz(n)` inkrementacji (łącznie wszystkich zmiennych występujących w funkcji) dla danego  $n$ .



Uwaga

Pętla `dla` inkrementuje zmienną licznika do wartości końcowej, nie przekracza jej. Dla przykładu: pętla dla  $i = 1, 2, 3, \dots, 5$  wykona cztery inkrementacje:  $1 \rightarrow 2, 2 \rightarrow 3, 3 \rightarrow 4, 4 \rightarrow 5$ .

Wzór: .....

Miejsce na obliczenia

### Zadanie 1.4.

Napisz nierekurencyjną funkcję sumuj( $n$ ), której wartością będzie suma:

wyznacz(1) + wyznacz(2) + wyznacz(3) + ... + wyznacz( $n-1$ ) + wyznacz( $n$ )

W funkcji wyznacz( $n$ ) użyto pętli *dla* dwukrotnie. W funkcji sumuj( $n$ ) możesz użyć pętli *dla* też **tylko dwukrotnie** i nie możesz wywoływać innych funkcji.

### Zadanie 2. Moc palindromiczna

Zbiorem wszystkich **podśłów początkowych** słowa  $z_1 z_2 z_3 \dots z_{n-1} z_n$   $z_1 z_2 z_3 \dots z_{n-1} z_n$ , złożonego z  $n$  znaków, będziemy nazywać następujący zbiór słów:

$z_1$

$z_1 z_2$

$z_1 z_2 z_3$

$z_1 z_2 z_3 \dots z_{n-1}$

$z_1 z_2 z_3 \dots z_{n-1} z_n$

**Z<sub>1</sub>****Z<sub>1</sub>Z<sub>2</sub>****Z<sub>1</sub>Z<sub>2</sub>Z<sub>3</sub>**

...

**Z<sub>1</sub>Z<sub>2</sub>Z<sub>3</sub>...Z<sub>n-1</sub>****Z<sub>1</sub>Z<sub>2</sub>Z<sub>3</sub>...Z<sub>n-1</sub>Z<sub>n</sub>**

**Mocą palindromiczną** słowa nazywać będziemy liczbę palindromów występujących w zbiorze wszystkich **podłów** początkowych danego słowa.

### Przykład:

Słowo *aeaeaeii* ma moc palindromiczną równą 3.

Podłowa (TAK oznacza, że dane podłowo jest palindromem):

a	TAK
ae	
aea	TAK
aaeae	
aaeae	TAK
aaeae	
aaeae	
aaeaeii	

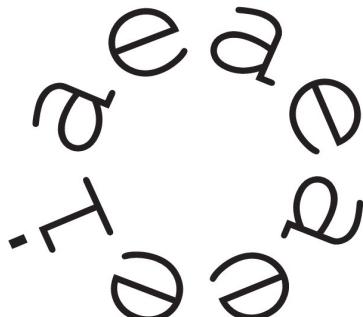
W pliku *słowa.txt* znajduje się sto słów o długości nieprzekraczającej dwudziestu znaków. Na podstawie danych zawartych w pliku *słowa.txt* w wybranym języku programowania napisz program (jeden lub kilka), który da odpowiedź na pytania lub wykona polecenia zawarte w zadaniach 2.1 – 2.3.

## Zadanie 2.1.

Spośród słów zawartych w pliku *słowa.txt* wypisz te o największej **mocy palindromicznej**. Podaj tę moc. Jeżeli słów o największej mocy jest więcej niż jedno, wypisz je wszystkie.

## Zadanie 2.2.

Po połączeniu ostatniego znaku słowa z pierwszym powstanie tzw. **pierścień słowny**. Dla słowa z przykładu pierścień słowny wygląda następująco:



W takim pierścieniu mamy następujące pięcioznakowe **podsłowa**:

aeaea  
eaeae  
aeaae  
eaeee  
aeeia  
eeiae  
eiaea  
iaeae

Ille jest słów o długości co najmniej sześciu znaków, w przypadku których ich pierścień słowny:

- a) nie zawiera żadnego pięcioznakowego **podsłowa** będącego palindromem?
- b) zawiera **dokładnie dwa różne podsłowa** będące palindromami?

### Zadanie 2.3.

Rozpatrując słowa o długości co najmniej sześciu znaków, wykonaj zestawienie średniej liczby **różnych pięcioznakowych podłów** będących palindromami, występujących w pierścieniu słowa według długości słowa. Tzn. oblicz średnią spośród wszystkich słów sześcioczątkowych, średnią spośród siedmioznakowych i tak dalej.

## Zadanie 3. Liceum dla dorosłych

W liceum ogólnokształcącym dla dorosłych Prymusik słuchacze uczęszczają na zajęcia w weekendy. Nauka odbywa się w trybie semestralnym. Przed rozpoczęciem każdego semestru słuchacze deklarują, których przedmiotów chcą się w danym semestrze uczyć. W ten sposób mogą oni dostosować czas nauki w szkole do swoich możliwości czasowych. Gdy słuchacz zaliczy odpowiednią liczbę semestrów z poszczególnych przedmiotów, otrzymuje świadectwo ukończenia liceum.

W pliku *oceny.txt* znajdują się informacje o realizowanych przez słuchaczy przedmiotach i uzyskanych ocenach częściowych w pierwszym semestrze roku szkolnego 2021/2022. Każdy wiersz w pliku zawiera: inicjały słuchacza, skróconą trzyliterową nazwę przedmiotu, na który słuchacz się zapisał, oraz uzyskane oceny częściowe. Dane w wierszu oddzielone zostały spacją.

**Przykładowy fragment pliku *oceny.txt*:**

HL geo 1 1 2  
UK wos  
ZF bio 2 1 1  
FN prz 6 6  
OS bio 1 2 6 4 4 6 1  
RT geo 2 5 3 5  
TT his 4 2  
II bio 5 6 4 3 6

*RT prz*

*UR jpo 6*

*TH bio 5 2 4 4 6*

*UR wos 4 2 2 5 1 1 1*

*ZF wos 1 2 6 1*

*WP his 3*

*HL bio 4 6 2*

W szkole prowadzone są zajęcia z następujących przedmiotów (w nawiasie trzyliterowy skrót): *matematyka (mat)*, *język polski (jpo)*, *język angielski (jan)*, *język hiszpański (jhi)*, *historia (his)*, *biologia (bio)*, *fizyka (fiz)*, *chemia (che)*, *informatyka (inf)*, *wiedza o społeczeństwie (wos)*, *geografia (geo)*, *podstawy przedsiębiorczości (prz)*.

Oceny semestralne oraz informację o nieklasyfikowaniu słuchacza z danego przedmiotu, pod koniec semestru, automatycznie generuje dziennik elektroniczny na podstawie zapisanych ocen cząstkowych.

W wybranym języku programowania napisz program (jeden lub kilka), który da odpowiedź na pytania lub umożliwi wykonanie poleceń zawartych w zadaniach 3.1 – 3.5.

### **Zadanie 3.1.**

W przypadku gdy słuchacz nie ma żadnej oceny cząstkowej z danego przedmiotu, dziennik elektroniczny nie może wystawić z niego oceny klasyfikacyjnej. W przykładowym fragmencie pliku system dwukrotnie nie mógł wystawić oceny klasyfikacyjnej — słuchaczowi o inicjałach *UK* z wiedzy o społeczeństwie oraz słuchaczowi o inicjałach *RT* z podstaw przedsiębiorczości.

W ilu przypadkach system nie wystawił oceny klasyfikacyjnej z powodu braku ocen cząstkowych?

### **Zadanie 3.2.**

W szkole nie ma dwóch słuchaczy o tych samych inicjałach. Każdy aktywny słuchacz musi w danym semestrze zadeklarować udział w zajęciach z co najmniej jednego przedmiotu. Ilu aktywnych słuchaczy było w szkole Prymusik w pierwszym semestrze roku szkolnego 2021/2022?

### **Zadanie 3.3.**

Na zajęcia z każdego z przedmiotów udział zadeklarował co najmniej jeden słuchacz. Ilu słuchaczy zadeklarowało udział w zajęciach z poszczególnych przedmiotów? Zestawienie posortuj malejąco według liczby słuchaczy.

### **Zadanie 3.4.**

Ocena semestralna jest automatycznie generowana przez dziennik elektroniczny. Jest ona średnią arytmetyczną ocen cząstkowych z danego przedmiotu zaokrągloną zgodnie z zasadami zaokrąglania do najbliższej liczby całkowitej.

Poniżej przedstawiono przykładowe oceny uzyskane przez słuchacza:

Przedmiot	Oceny częściowe	Średnia arytmetyczna ocen częściowych (do dwóch miejsc po przecinku)	Ocena semestralna
his	4 2	3,00	3
bio	5 6 4 3 6	4,80	5
jpo	6	6,00	6
mat	5 2 4 4 6 3	3,43	3
wos	4 2 2 5 1 1 1	2,29	2
jan	1 2 1 1	1,25	1
his	3	3,00	3

Średnia ocen semestralnych tego słuchacza wynosi 3,29.

Wiedząc, że słuchacz o inicjałach PT jest klasyfikowany (uzyskał co najmniej jedną ocenę częściową) ze wszystkich przedmiotów, z których zadeklarował udział w zajęciach, oblicz jego średnią ocen semestralnych.

### Zadanie 3.5.

Najlepsi słuchacze w danym semestrze otrzymują nagrodę finansową za dobre wyniki w nauce. Abytrzymać taką nagrodę, słuchacz musi spełnić wszystkie z poniższych warunków:

- a) w danym semestrze musi zadeklarować udział w zajęciach z co najmniej pięciu przedmiotów,
- b) musi być sklasyfikowany ze wszystkich przedmiotów, z których zadeklarował udział w zajęciach,
- c) każda ocena semestralna nie może być niższa niż 3,
- d) średnia ocen semestralnych musi wynosić co najmniej 4,00.

Wypisz inicjały słuchaczy, którzy otrzymali nagrodę za pierwszy semestr roku szkolnego 2021/2022, oraz ich średnie ocen semestralnych. Zestawienie posortuj malejąco według średniej ocen semestralnych.

### Zadanie 4. Wakacyjne postanowienie Ani

Studentka Ania postanowiła podczas zbliżających się wakacji zadbać o swoją kondycję i sprawność fizyczną. Ania jest zadowolona ze swojej figury i wagi, dlatego stwierdziła, że wszystkie dodatkowe działania przez nią podjęte nie mogą spowodować utraty wagi. Stwierdziła ponadto, że będzie zwracać szczególną uwagę na zachowanie bilansu energetycznego.

Ania pobuszuowała w internecie i znalazła dla siebie odpowiedni program. Jego głównym założeniem była cykliczność. Jeden cykl trwał pięć dni. Po zakończeniu cyklu następnego dnia rozpoczynał się kolejny cykl. Założenia programu były następujące:

- a) W pierwszym dniu cyklu Ania jeździ rowerem przez jedną godzinę. Przez kolejne trzy dni, w każdym następnym dniu, zwiększa czas spędzony na rowerze o 15 minut. Na rowerze Ania jeździ około godziny 10.00.
- b) W piątym dniu Ania nie jeździ na rowerze.
- c) Dodatkowo we wtorki i w piątki Ania około godziny 16.00 uprawia jogging, który trwa dwie godziny.

Podczas jednej godziny jazdy rowerem Ania spala 420 kcal, natomiast w trakcie jednej godziny joggingu traci 540 kcal. Aby zachować zdrowy bilans energetyczny, Ania dodatkowo spożywa produkty, które łącznie dostarczają jej każdego dnia dodatkowe 720 kcal. Niespalone kalorie lub niedobór kalorii przechodzi na następny dzień.

Każdego dnia wieczorem, po wszystkich posiłkach, Ania dokonuje bilansu spalonych i dostarczonych kalorii. Jeśli brakuje więcej niż 200 kcal, Ania spożywa dodatkowy posiłek uzupełniający, którego wartość energetyczna wynosi 300 kcal.

#### Przykład:

Założmy, że w pewną środę rano bilans energetyczny Ani był ujemny, brakowało 120 kcal. W tym dniu jeździła rowerem tylko przez jedną godzinę. Podczas jazdy spaliła 420 kcal. Spożywając odpowiednie produkty, jak każdego dnia, dostarczyła organizmowi 720 kcal. Wieczorem dokonała bilansu i otrzymała:  $-120 - 420 + 720 = 180$  kcal. Zatem tego wieczoru studentka nie musiała spożywać posiłku uzupełniającego.

Czwartek rozpoczęła bilansem dodatnim — z nadmiarem 180 kcal. W tym dniu godzinę i piętnaście minut jeździła rowerem oraz dwie godziny spędziła na joggingu. Na rowerze spaliła  $1,25 \times 420 = 525$  kcal, natomiast podczas biegu  $2 \times 540 = 1080$  kcal. Dokonując wieczornego bilansu, otrzymała:  $180 - 525 - 1080 + 720 = -705$  kcal. Ponieważ niedobór kilokalorii był większy o 200, Ania spożyła posiłek uzupełniający, który dostarczył jej 300 kcal. Zatem następny dzień rozpoczął się bilansem  $-705 + 300 = -405$  kcal.

Ania realizowała zaplanowany program przez wakacje, w dniach od 1.07.2022 do 30.09.2022. Rano w pierwszym dniu wakacji jej bilans energetyczny wynosił zero.

Wykorzystując dostępne narzędzia informatyczne, podaj odpowiedzi do pytań lub wykonaj czynności opisane w zadaniach 4.1 – 4.5.

### Zadanie 4.1.

Ile łącznie godzin przez całe wakacje Ania spędziła, jeżdżąc rowerem, a ile uprawiając jogging?

### Zadanie 4.2.

Ile było dni, w których Ania, uprawiając sport, spaliła co najmniej 1600 kcal, ale nie więcej niż 1700 kcal?

### Zadanie 4.3.

Ile razy w ciągu wakacji studentka wieczorem spożywała posiłek uzupełniający?

## Zadanie 4.4.

Utwórz zestawienie łącznej liczby godzin spędzonych przez Anię na rowerze w poszczególne dni tygodnia wakacji, tzn. sumy godzin spędzonych na rowerze we wszystkie poniedziałki, sumy godzin spędzonych na rowerze we wszystkie wtorki i tak dalej.

Do powstałego zestawienia utwórz wykres kolumnowy. Pamiętaj o czytelności wykresu. Zarówno w zestawieniu, jak i na wykresie muszą wystąpić pełne nazwy dni tygodnia.

## Zadanie 4.5.

Jaka jest największa całkowita liczba kilokalorii, które spalać powinna studentka w ciągu jednej godziny joggingu, aby w trakcie wakacji liczba dni, w których spożywały posiłek uzupełniający, nie przekroczyła trzech?

# Zadanie 5. Nie tylko SELECT

Rozwiążując poniższe zadania, w pierwszej kolejności zapisz dane zapytanie w języku SQL w postaci pisemnej w miejscu na to przeznaczonym. Następnie sprawdź jego poprawność, wykonując je w wybranym systemie baz danych.

## Zadanie 5.1.

Utwórz zapytanie w języku SQL, które utworzy tabelę o nazwie *auta* o następującej specyfikacji:

Nazwa pola	Typ danych
Id	klucz główny, liczba całkowita, autonumeracja
nr_rej	tekst, nie może być wartością pustą
Przebieg	liczba całkowita
rok_prod	liczba całkowita
Pojemność	liczba rzeczywista
Klimatyzacja	wartość logiczna: prawda/fałsz

Następnie utwórz kolejne zapytanie w języku SQL, które utworzy tabelę o nazwie *wypożyczenia* o następującej specyfikacji:

Nazwa pola	Typ danych
Id	klucz główny, liczba całkowita, autonumeracja
id_auta	liczba całkowita
Pesel	Tekst
Data	Data
Dystans	liczba całkowita

Pole *id\_auta* w tabeli *wypożyczenia* jest kluczem obcym (klucz główny *id* w tabeli *auta*).

## Zadanie 5.2.

Utwórz zapytanie w języku SQL, które wstawi do tabeli *auta* następujące rekordy:

nr_rej	przebieg	rok_prod	pojemnosc	Klimatyzacja
PO12345	25000	2019	1,6	True
ZS98765	67000	2014	1,8	True
PSZ22233	12000	2020	1,6	True
DW87872	150000	2001	2,0	False
KR12321	75000	2005	2,2	False

Następnie utwórz zapytanie w języku SQL, które wstawi do tabeli *wypożyczenia* następujące rekordy:

<b>id_auta</b>	<b>pesel</b>	<b>data</b>	<b>dystans</b>
1	73100764862	2021-10-10	30
3	86111651722	2021-10-13	50
4	94011552087	2021-10-14	45
2	73100764862	2021-10-14	35
2	86111651722	2021-10-20	50
4	71020373847	2021-10-21	70
2	73100764862	2021-10-25	25

## Zadanie 5.3.

Utwórz zapytanie w języku SQL, które zwiększy przebieg wszystkich samochodów wyprodukowanych przed rokiem 2014 o 5000 kilometrów.

## Zadanie 5.4.

Utwórz zapytanie w języku SQL, które wypisze rok produkcji najmłodszego auta.

## Zadanie 5.5.

Utwórz zapytanie w języku SQL, które dla każdego auta wypisze średni dystans pokonywany tym autem podczas jego wypożyczenia.

## Zadanie 5.6.

Utwórz zapytanie w języku SQL, które wypisze liczbę wypożyczeń poszczególnych pojazdów.

## Zadanie 5.7.

Utwórz zapytanie w języku SQL, które usunie z tabeli *auta* te pojazdy, których nikt nigdy nie wypożyczył.

# Odpowiedzi i wskazówki do zestawu 3

## Zadanie 1.1.

<b>n</b>	<b>Wynik zwrócony przez funkcję <code>wyznacz(n)</code></b>
1	2
2	5
3	9
4	14
5	20

## Zadanie 1.2.

Przykładowe rozwiązanie:

```
funkcja wyznacz_rek(n)
    jeżeli n=1
        zwróć 2
    w przeciwnym razie
        zwróć n+1+wyznacz_rek(n-1)
```

W naszym przykładowym rozwiążaniu liczba wywołań `wyznacz_rek(n)` wynosi  $n$ .

## Zadanie 1.3.

Wzór:  $liczba\ inkrementacji = n^2 + 2n - 1$

Wyjaśnienie:

Pętla ze zmienną  $i$  wykona  $n - 1$  inkrementacji zmiennej  $i$ . W tym czasie  $n$  razy zwiększy wartość zmiennej  $x$ .

W przypadku pętli ze zmienną  $j$  sytuacja wygląda następująco: dla  $i = 1$  nie zostanie wykonana inkrementacja zmiennej  $j$ , natomiast raz zostanie zwiększoa wartość zmiennej  $y$ . Dla  $i = 2$  zostanie wykonana jedna inkrementacja zmiennej  $j$  oraz dwie inkrementacje zmiennej  $y$  i tak dalej. Dla  $i = n$  zmienna  $j$  zostanie zwiększoa  $n - 1$  razy, natomiast zmienna  $y$  o jeden raz więcej, czyli  $n$  razy.

Korzystając ze wzoru na sumę początkowych elementów ciągu arytmetycznego, stwierdzamy, że wszystkich inkrementacji zmiennej  $j$  jest  $\frac{n(n-1)}{2}$ , a wszystkich inkrementacji zmiennej  $y$  jest  $\frac{n(n+1)}{2}$ .

Sumując inkrementacje wszystkich zmiennych, otrzymujemy:

$$liczba\ inkrementacji = (n - 1) + n + \frac{n(n - 1)}{2} + \frac{n(n + 1)}{2} = n^2 + 2n - 1$$

## Zadanie 1.4.

Przykładowe rozwiązanie:

```

funkcja sumuj(n)
    s ← 0
    dla i=1,2,3,...,n wykonuj
        dla j=2,3,...,i+1 wykonuj
            s ← s+j
        zwróć s

funkcja sumuj(n)
    s ← 0
    dla i=1,2,3,...,n wykonuj
        dla j=2,3,...,i+1 wykonuj
            s ← s+j
    zwróć s

```

Zauważ, że:

```

wyznacz(2) - wyznacz(1) = 3
wyznacz(3) - wyznacz(2) = 4
wyznacz(4) - wyznacz(3) = 5
wyznacz(5) - wyznacz(4) = 6
wyznacz(6) - wyznacz(5) = 7

```

i tak dalej.

## Zadanie 2.

```

#include <iostream>
#include <fstream>
using namespace std;
string s, s2, s3, s4;
int i, j, k, l_slow, l_palindr, najw_l_palindr, dl_palindr, najw_dl_palindr,
    l_slow_najw_mocy, l_5palindr, l_5palindr_powt, l_slow_bez_5palindr,
    najw_l_5palindr, l_slow_z_3_5palindr, l_slow_z_3_5palindr_powt;;
    fstream plik1, plik2;
    int l_roznych_5palindr, najw_dlugosc;
    bool rozne;
    double sr;

string slowa_najw_mocy[100];

struct dane_sr
{
    int ilosc, suma;
};

dane_sr tab_do_sr[30];

string samogl[4]={"a","e","o","i"};
string slowa[1000];

string rozne_5palindr[100];

bool jest_palindrom(string ss)
{
    bool ww;

```

```
int ii;
ww=true;
ii=0;
while ((ii<ss.length()/2) && (ww==true))
{if (ss[ii]!=ss[ss.length()-ii-1])
{ww=false;
}
ii++;
}
return ww;
}

int main()
{
//wczytanie słów

l_slow=0;
plik1.open("slowa.txt", ios::in);
if(plik1.good() == true)
{while(!plik1.eof())
{getline(plik1, s);
if (s!="")
{
//cout << s << endl;
slowa[l_slow]=s;
l_slow++;
}
}
plik1.close();
najw_l_palindr=0;
najw_dl_palindr=0;
l_slow_najw_mocy=0;
l_slow_bez_5palindr=0;
najw_l_5palindr=0;
l_slow_z_3_5palindr=0;
l_slow_z_3_5palindr_powt=0;
najw_dlugosc=0;

for (i=0; i<30; i++)
{tab_do_sr[i].ilosc=0;
tab_do_sr[i].suma=0;
}

for (i=0; i<l_slow; i++)
{cout << slowa[i] << endl;

if (slowa[i].length()>najw_dlugosc)
{najw_dlugosc=slowa[i].length();
}
l_palindr=0;
for (j=0; j<slowa[i].length(); j++)
{s2=slowa[i].substr(0,j+1);
cout << " " << s2;
if (jest_palindrom(s2)==true)
{cout << " TAK ";
if (s2.length()>najw_dl_palindr)
```

```
        {najw_dl_palindr=s2.length();
     }
     l_palindr++;
}
cout << endl;
}
if (l_palindr>najw_l_palindr)
{najw_l_palindr=l_palindr;
 slowa_najw_mocy[0]=slowa[i];
 l_slow_najw_mocy=1;
}
else
{if (l_palindr==najw_l_palindr)
 {slowa_najw_mocy[l_slow_najw_mocy]=slowa[i];
 l_slow_najw_mocy++;
}
}
if (slowa[i].length()>5)
{
    l_roznych_5palindr=0;
s3=slowa[i]+slowa[i];
l_5palindr=0;
l_5palindr_powt=0;
for (j=1; j<=slowa[i].length(); j++)
{s4=s3.substr(j-1,j);
cout << " " << s4;
if (jest_palindrom(s4)==true)
{
    cout << " T ";
    rozne=true;
    for (k=0; k<l_roznych_5palindr; k++)
    {if (rozne_5palindr[k]==s4)
     {rozne=false;
    }
}
if (rozne==true)
{rozne_5palindr[l_roznych_5palindr]=s4;
l_roznych_5palindr++;
l_5palindr++;
}
l_5palindr_powt++;
}
cout << endl;
}
if (l_5palindr==0)
{l_slow_bez_5palindr++;
}
if (l_5palindr==2)
{l_slow_z_3_5palindr++;
}
tab_do_sr[slowa[i].length()].ilosc++;
tab_do_sr[slowa[i].length()].suma=tab_do_sr[slowa[i].length()]
➥.suma+l_5palindr;
```

```

    if (l_5palindr_powt==2)
    {l_slow_z_3_5palindr_powt++;
    }

    if (l_5palindr>najw_l_5palindr)
    {najw_l_5palindr=l_5palindr;
    }
    cout << "          roznych = " << l_5palindr << " nierozych = "
    >><< l_5palindr_powt << endl;;
    if (l_5palindr!=l_5palindr_powt)
    {cout << "
    *****" << endl;
    }
    cout << endl;
}

cout << endl << "KONIEC" << endl << endl;
cout << "Najwieksza liczba palindromow = " << najw_l_palindr << endl;
cout << "Najwieksza dlugosc palindromu = " << najw_d_l_palindr << endl;
cout << "Liczba slow bez 5-palindr = " << l_slow_bez_5palindr << endl;
cout << "Liczba slow z 2 5-palindr = " << l_slow_z_2_5palindr << endl;
cout << "Liczba slow z 2 5-palindr z powt = " << l_slow_z_2_5palindr_powt
    >><< endl;
cout << "Najwieksza liczba 5-palindr = " << najw_l_5palindr << endl;

cout << endl << "Liczba slow o najwiekszej mocy: " << l_slow_najw_mocy << endl;
cout << endl << "Słowa o najwiekszej mocy: " << endl;
for (i=0; i<l_slow_najw_mocy; i++)
{cout << słowa_najw_mocy[i] << endl;
}
cout << endl << endl;

for (i=6; i<=najw_dlugosc; i++)
{sr=(tab_do_sr[i].suma*1.0)/tab_do_sr[i].ilosz;
 cout << i << "-znakow, srednia = " << sr << endl;
};

plik1.close();
plik2.close();
return 0;
}

```

## Zadanie 2.1.

**Odpowiedź:**

Największa moc: 3

Słowa o mocy 3:

eeeioaoaaai

iiaiioeoiooooeaaao

iaiaioe

eeeeaeoeeeoaaaoaaao

iaiai

eeeioaaooeaoeooe  
eeeiaae  
aoaaeaeaoaaeaoi

## Zadanie 2.2.



Uwaga

Pięcioznakowe **podsłowa** w pierścieniu słownym łatwo wyznaczysz, „sklejając” ze sobą to samo słowo. Otrzymujesz słowo o długości  $2n$ . Następnie analizujesz pięcioznakowe **podsłowa**, począwszy od pierwszego znaku, a skończysz na znaku na pozycji  $n$ -tej.

Odpowiedź:

- a) 40
- b) 8

## Zadanie 2.3.

6-znakow, srednia = 0.333333  
7-znakow, srednia = 0.428571  
8-znakow, srednia = 0.333333  
9-znakow, srednia = 0.5  
10-znakow, srednia = 0.5  
11-znakow, srednia = 1.11111  
12-znakow, srednia = 0.833333  
13-znakow, srednia = 0.75  
14-znakow, srednia = 0.25  
15-znakow, srednia = 0.666667  
16-znakow, srednia = 0.6  
17-znakow, srednia = 1.5

## Zadanie 3.

Zmienne i struktury zastosowane do rozwiązania zadań:

```
#include <iostream>
#include <fstream>
using namespace std;
string s, s2, s3, s4, wybr_sluch;
int i, j, k, p, m, n, oc, l_liter, l_sluchaczy, l_przed_sluch, l_wierszy,
    l_sluch_akt;
int l_nieklas, ocena_czastk, poczatek, koniec, indeks, l_dobrych;
fstream plik1, plik2;
bool czy_jest, ten_sam;
int oceny_sem_sluch[20];
int l_ocen_sem_sluch, l_ocen_czastk_sluch, suma_ocen_czastk_sluch,
    l_ocena_sem_sluch, suma_ocena_sem_sluch;
double sr_czastk, sr_sem;
const int l_wsz_przed = 12;
struct inf_przed
```

```

    {string skrot;
     int l_sluch;
    };
struct inf_do_styp
{string inicjaly;
 int s_sem, l_sem;
 double sr_sem;
 bool odpada;
};
inf_do_styp sluchacze_do_styp[1000];
int l_sluch_do_styp;
inf_do_styp pom2;
inf_przed pom;
inf_przed l_sluch_na_przed[20];
int l_przedmiotow;
string
przedmioty[l_wsz_przed]={"mat","jpo","jan","jhi","his","bio","fiz","che","inf",
"wos","geo","prz"};
string przedmioty_p[l_wsz_przed];
string litery = "ABCDEFGHIJKLMNOPRSTUWZ";
string wsz_inicjaly[500];
string inicjaly_sluch[50];
string inicjaly_akt[100];
string zaw_pliku[10000];

```

## Zadanie 3.1.

```

int main()
{
    l_nieklass=0;
    plik1.open("oceny.txt", ios::in);
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                 {if (s.length()==6)
                  {l_nieklass++;
                   }
                 }
            }
        }
    plik1.close();
    cout << "Liczba niekasyfikowan = " << l_nieklass << endl;
    getchar();
    return 0;
}

```

**Odpowiedź:**

W 15 przypadkach.

## Zadanie 3.2.

```

int main()
{
    l_sluch_akt=0;
    plik1.open("oceny.txt", ios::in);
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                 {s2=s.substr(0,2);

                  czy_jest=false;
                  i=0;
                  while ((i<l_sluch_akt) && (czy_jest==false))
                      {if (inicjaly_akt[i]==s2)
                         {czy_jest=true;
                          }
                         i++;
                      }

                  if (czy_jest==false)
                      {inicjaly_akt[l_sluch_akt]=s2;
                       l_sluch_akt++;
                      }
                  }
             }
        }
    plik1.close();

    cout << "Liczba sluchaczy aktywnych = " << l_sluch_akt << endl;

    getchar();
    return 0;
}

```

**Odpowiedź:**

Liczba aktywnych słuchaczy = 68

## Zadanie 3.3.

```

int main()
{
    l_przedmiotow=0;
    plik1.open("oceny.txt", ios::in);
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                 {
                     s2=s.substr(3,3);

                     czy_jest=false;
                     i=0;
                     while ((i<l_przedmiotow) && (czy_jest==false))
                         {if (l_sluch_na_przed[i].skrot==s2)
                            {czy_jest=true;
                             }
                           }
                     }
            }
        }
    plik1.close();
}

```

```

        }
        i++;
    }

    if (czy_jest==false)
    {l_sluch_na_przed[l_przedmiotow].skrot=s2;
     l_sluch_na_przed[l_przedmiotow].l_sluch=1;
     l_przedmiotow++;
    }
    else
    {l_sluch_na_przed[i-1].l_sluch++;
    }

}
}

plik1.close();

//sortowanie
for (i=l_przedmiotow-2; i>=0; i--)
{for (j=0; j<=i; j++)
 {if (l_sluch_na_przed[j].l_sluch<l_sluch_na_przed[j+1].l_sluch)
 {
    pom=l_sluch_na_przed[j];
    l_sluch_na_przed[j]=l_sluch_na_przed[j+1];
    l_sluch_na_przed[j+1]=pom;

    }
 }
}

cout << endl << "Liczby sluchaczy na przedmiotach po sortowaniu" << endl;
for (i=0; i<l_przedmiotow; i++)
 {cout << l_sluch_na_przed[i].skrot << " " << l_sluch_na_przed[i].l_sluch << endl;
 }

getchar();
return 0;
}

```

**Odpowiedź:**

jhi 43

bio 41

fiz 41

wos 40

inf 39

jpo 39

che 38

prz 37

geo 36

jan 36

his 36

mat 34

## Zadanie 3.4.

```
int main()
{
    wybr_sluch = "PT";

    l_ocen_sem_sluch=0;
    suma_ocen_sem_sluch=0;
    plik1.open("oceny.txt", ios::in);
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                 {if (s.substr(0,2)==wybr_sluch)
                  {
                      if (s.length()>6)
                          {s2=s.substr(7);

                           suma_ocen_czastk_sluch=0;
                           l_ocen_czastk_sluch=0;
                           k=s2.find(" ");
                           while (k>0)
                               {s3=s2.substr(0,k);
                                s2=s2.substr(k+1);
                                ocena_czastk=atoi(s3.c_str());
                                k=s2.find(" ");

                                suma_ocen_czastk_sluch=suma_ocen_czastk_sluch+ocena_czastk;
                                l_ocen_czastk_sluch++;
                               }
                           ocena_czastk=atoi(s2.c_str());

                           suma_ocen_czastk_sluch=suma_ocen_czastk_sluch+ocena_czastk;
                           l_ocen_czastk_sluch++;
                           sr_czastk=suma_ocen_czastk_sluch*1.0/l_ocen_czastk_sluch;
                           ocena_sem_sluch=round(sr_czastk);

                           l_ocen_sem_sluch++;
                           suma_ocen_sem_sluch=suma_ocen_sem_sluch+ocena_sem_sluch;
                         }
                     }
                 }
             }
         }
    plik1.close();
    sr_sem=suma_ocen_sem_sluch*1.0/l_ocen_sem_sluch;

    cout << endl << "Srednia ocen semestralnych sluchacza " << wybr_sluch
    <><< " wynosi " << sr_sem << endl;

    getchar();
    return 0;
}
```

**Odpowiedź:**

Średnia ocen semestralnych słuchacza PT wynosi 3.5.

## Zadanie 3.5.

```

int main()
{
    l_sluch_do_styp=0;
    plik1.open("oceny.txt", ios::in);
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                {
                    czy_jest=false;
                    indeks=0;
                    while ((indeks<=l_sluch_do_styp) && (czy_jest==false))
                        {if (sluchacze_do_styp[indeks].inicjaly==s.substr(0,2))
                            {czy_jest=true;
                             }
                         indeks++;
                        }

                    if (czy_jest==true)
                        {indeks--;
                         }
                    else
                        {sluchacze_do_styp[indeks].inicjaly=s.substr(0,2);
                         sluchacze_do_styp[indeks].odpada=false;
                         sluchacze_do_styp[indeks].l_sem=0;
                         sluchacze_do_styp[indeks].s_sem=0;
                         sluchacze_do_styp[indeks].sr_sem=0;
                         l_sluch_do_styp++;
                        }
                }

            if (sluchacze_do_styp[indeks].odpada==false)
                {
                    if (s.length()==6)
                        { //nieklasyfikowany
                         sluchacze_do_styp[indeks].odpada=true;
                        }
                    else
                        {s2=s.substr(7);
                         suma_ocen_czastk_sluch=0;
                         l_ocen_czastk_sluch=0;
                         k=s2.find(" ");
                         while (k>0)
                             {s3=s2.substr(0,k);
                              s2=s2.substr(k+1);
                              ocena_czastk=atoi(s3.c_str());
                              k=s2.find(" ");
                              suma_ocen_czastk_sluch=suma_ocen_czastk_sluch+ocena_czastk;
                              l_ocen_czastk_sluch++;
                             }

                         ocena_czastk=atoi(s2.c_str());
                         suma_ocen_czastk_sluch=suma_ocen_czastk_sluch+ocena_czastk;
                         l_ocen_czastk_sluch++;
                         sr_czastk=suma_ocen_czastk_sluch*1.0/l_ocen_czastk_sluch;
                         ocena_sem_sluch=round(sr_czastk);

                         if (ocena_sem_sluch>=3)
                            {sluchacze_do_styp[indeks].l_sem++;
                            }
                        }
                }
            }
        }
}

```

```
sluchacze_do_styp[indeks].s_sem=sluchacze_do_styp[indeks]
↳.s_sem+ocena_sem_sluch;
        }
    else
        {sluchacze_do_styp[indeks].odpada=true;
        }
    }

}
}

}
}

plik1.close();

//sprawdzenie, ile przedmiotów zadeklarowali, i ewentualne obliczenie średniej
for (i=0; i<l_sluch_do_styp; i++)
{if ((sluchacze_do_styp[i].l_sem>=5) && (sluchacze_do_styp[i].odpada==false))
{sluchacze_do_styp[i].sr_sem=(sluchacze_do_styp[i].s_sem*1.0)/
↳(sluchacze_do_styp[i].l_sem*1.0);
}
else
    {sluchacze_do_styp[i].odpada=true;
    }
}

//sortowanie — odpada na koniec
for (i=l_sluch_do_styp-2; i>=0; i--)
{for (j=0; j<=i; j++)
    {if  ((sluchacze_do_styp[j].odpada>sluchacze_do_styp[j+1].odpada))
    {
        pom2=sluchacze_do_styp[j];
        sluchacze_do_styp[j]=sluchacze_do_styp[j+1];
        sluchacze_do_styp[j+1]=pom2;
    }
}
}

//ile odpadlo
l_dobrych=0;
while ((l_dobrych<l_sluch_do_styp) &&
(sluchacze_do_styp[l_dobrych].odpada==false))
{l_dobrych++;
};

cout << endl << "Liczba dobrych = " << l_dobrych << endl;

//sortowanie tylko w ramach dobrych według średniej
for (i=l_dobrych-2; i>=0; i--)
{for (j=0; j<=i; j++)
    {if  ((sluchacze_do_styp[j].sr_sem<sluchacze_do_styp[j+1].sr_sem))
    {
        pom2=sluchacze_do_styp[j];
        sluchacze_do_styp[j]=sluchacze_do_styp[j+1];
        sluchacze_do_styp[j+1]=pom2;
    }
}
}
```

```

for (i=0; i<l_sluch_do_styp; i++)
    {if ((sluchacze_do_styp[i].odpada==false) && (sluchacze_do_styp[i].sr_sem>=4))
     { cout << i+1 << " " << sluchacze_do_styp[i].inicjaly << " "
       <><< sluchacze_do_styp[i].sr_sem << endl;
     }
    }

getchar();
return 0;
}

```

**Wskazówka:**

Wykorzystaj program z zadania 3.4.

**Odpowiedź:**

SD 4.57143

OW 4.4

HE 4.33333

RI 4.28571

RR 4.25

MI 4.22222

JG 4.2

JO 4.11111

**Zadanie 4.1.**

A	B	C	D	E	F	G	H	I	J	K
1			=DZIEŃ.TYG(C5;2)	=JEŻELI(E5<5;E5+1;1)	=JEŻELI(E6=1;1;JEŻELI(E6=5;0;G5+0,25))					
2										
3										
4		Dzień	Dzień tygodnia	Dzień cyklu		Czas na rowerze	Czas joggingu			
5		01.07.2022	5	1		1	2			=JEŻELI(LUB(D5=2;D5=5);2;0)
6		02.07.2022	6	2		1,25	0			
7		03.07.2022	7	3		1,5	0			
8		04.07.2022	1	4		1,75	0			
9		05.07.2022	2	5		0	2			
10		06.07.2022	3	1		1	0			
11		07.07.2022	4	2		1,25	0			
12		08.07.2022	5	3		1,5	2			
13		09.07.2022	6	4		1,75	0			
14		10.07.2022	7	5		0	0			
15		11.07.2022	1	1		1	0			
16		12.07.2022	2	2		1,25	2			
17		13.07.2022	3	3		1,5	0			
18		14.07.2022	4	4		1,75	0			
19		15.07.2022	5	5		0	2			
20		16.07.2022	6	1		1	0			

**Odpowiedź:**

Na rowerze: 101,25 h

Na joggingu: 54 h

## Zadanie 4.2.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1										=G5*420	=H5*540		=J5+K5	=JEŻELI(ORAZ(M5>=1600;M5<=1700);1;0)		
2										Kalorie na rowerze	Kalorie na joggingu		Suma kalorii	suma kalorii		
3																
4				Dzień	Dzień tygodnia	Dzień cyklu			Czas na rowerze	Czas joggingu						
5				01.07.2022	5	1			1	2						
6				02.07.2022	6	2			1,25	0	525	0	525	0		
7				03.07.2022	7	3			1,5	0	630	0	630	0		
8				04.07.2022	1	4			1,75	0	735	0	735	0		
9				05.07.2022	2	5			0	2	0	1080	1080	0		
10				06.07.2022	3	1			1	0	420	0	420	0		
11				07.07.2022	4	2			1,25	0	525	0	525	0		
12				08.07.2022	5	3			1,5	2	630	1080	1710	0		
13				09.07.2022	6	4			1,75	0	735	0	735	0		
14				10.07.2022	7	5			0	0	0	0	0	0		
15				11.07.2022	1	1			1	0	420	0	420	0		
16				12.07.2022	2	2			1,25	2	525	1080	1605	1		
17				13.07.2022	3	3			1,5	0	630	0	630	0		
18				14.07.2022	4	4			1,75	0	735	0	735	0		
19				15.07.2022	5	5			0	2	0	1080	1080	0		
20				16.07.2022	6	1			1	0	420	0	420	0		

Odpowiedź:

Liczba dni = 6

## Zadanie 4.3.

	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1																
2																
3																
4				Dzień	Dzień tygodnia	Dzień cyklu			Czas na rowerze	Czas joggingu		Kalorie na rowerze	Kalorie na joggingu	Suma kalorii		
5				01.07.2022	5	1			1	2	420	1080	1500			
6				02.07.2022	6	2			1,25	0	525	0	525	-780	300	
7				03.07.2022	7	3			1,5	0	630	0	630	-285	0	
8				04.07.2022	1	4			1,75	0	735	0	735	90	0	
9				05.07.2022	2	5			0	2	0	1080	1080	-270	300	
10				06.07.2022	3	1			1	0	420	0	420	330	0	
11				07.07.2022	4	2			1,25	0	525	0	525	525	0	
12				08.07.2022	5	3			1,5	2	630	1080	1710	-465	300	
13				09.07.2022	6	4			1,75	0	735	0	735	-180	0	
14				10.07.2022	7	5			0	0	0	0	0	540	0	
15				11.07.2022	1	1			1	0	420	0	420	840	0	
16				12.07.2022	2	2			1,25	2	525	1080	1605	45	0	
17				13.07.2022	3	3			1,5	0	630	0	630	45	0	
18				14.07.2022	4	4			1,75	0	735	0	735	30	0	
19				15.07.2022	5	5			0	2	0	1080	1080	-330	300	
20				16.07.2022	6	1			1	0	420	0	420	270	0	

Odpowiedź:

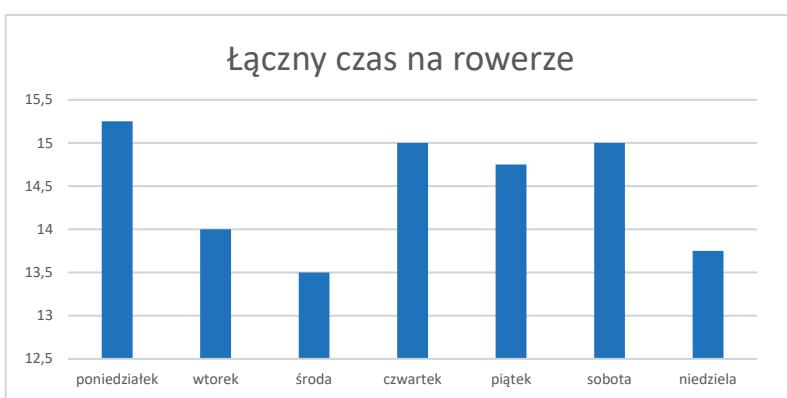
Posiłek uzupełniający spożywała 18 razy.

## Zadanie 4.4.

A	B	C	D	E	F	G	H	I	J
1									
2									
3					=TEKST(C5;"dddd")				
4		Dzień cyklu	Czas na rowerze		Dzień tygodnia				
5	01.07.2022	1	1		piątek				
6	02.07.2022	2	1,25		sobota				
7	03.07.2022	3	1,5		niedziela				
8	04.07.2022	4	1,75		poniedziałek				
9	05.07.2022	5	0		wtorek				
10	06.07.2022	1	1		środa				
11	07.07.2022	2	1,25		czwartek				
12	08.07.2022	3	1,5		piątek				
13	09.07.2022	4	1,75		sobota				
14	10.07.2022	5	0		niedziela				
15	11.07.2022	1	1		poniedziałek				
16	12.07.2022	2	1,25		wtorek				
17	13.07.2022	3	1,5		środa				
18	14.07.2022	4	1,75		czwartek				
19	15.07.2022	5	0		piątek				
20	16.07.2022	1	1		sobota				
21	17.07.2022	2	1,25		niedziela				
22	18.07.2022	3	1,5		poniedziałek				
23	19.07.2022	4	1,75		wtorek				
24	20.07.2022	5	0		środa				
						Etykiety wierszy	Suma z Czas na rowerze		
						poniedziałek	15,25		
						wtorek	14		
						środa	13,5		
						czwartek	15		
						piątek	14,75		
						sobota	15		
						niedziela	13,75		
						Suma końcowa	101,25		

**Odpowiedź:**

Dzień tygodnia	Łączny czas na rowerze
poniedziałek	15,25
wtorek	14
środa	13,5
czwartek	15
piątek	14,75
sobota	15
niedziela	13,75



## Zadanie 4.5.

	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1																		
2																		
3																		
4	Dzień	Dzień tygodnia	Dzień cyklu		Czas na rowerze	Czas joggingu												
5	01.07.2022	5	1		1	2			420	902		1322		-720-M5	->JEŻELI(OS<-200;300;0)			
6	02.07.2022	6	2		1,25	0			525	0		525		-602	300			
7	03.07.2022	7	3		1,5	0			630	0		630		-107	0			
8	04.07.2022	1	4		1,75	0			735	0		735		-32	0			
9	05.07.2022	2	5		0	2			0	902		902		-214	300			
10	06.07.2022	3	1		1	0			420	0		420		386	0			
11	07.07.2022	4	2		1,25	0			525	0		525		581	0			
12	08.07.2022	5	3		1,5	2			630	902		1532		231	200			
13	09.07.2022	6	4		1,75	0			735	0		735		54	0			
14	10.07.2022	7	5		0	0			0	0		0		774	0			
15	11.07.2022	1	1		1	0			420	0		420		1074	0			
16	12.07.2022	2	2		1,25	2			525	902		1427		367	0			
17	13.07.2022	3	3		1,5	0			630	0		630		457	0			
18	14.07.2022	4	4		1,75	0			735	0		735		442	0			
19	15.07.2022	5	5		0	2			0	902		902		260	0			
20	16.07.2022	6	1		1	0			420	0		420		560	0			

**Odpowiedź:**

Największa możliwa liczba kalorii wynosi 451 kcal.

## Zadanie 5.1.

```
CREATE TABLE auta
(id COUNTER PRIMARY KEY,
nr_rej text NOT NULL,
przebieg int,
rok_prod int,
pojemnosc float,
klimatyzacja bit);
```

```
CREATE TABLE wypożyczenia
(id COUNTER PRIMARY KEY,
id_autu int REFERENCES auta(id),
pesel text,
data date,
dystans int);
```

## Zadanie 5.2.

```
INSERT INTO auta(nr_rej, przebieg, rok_prod, pojemnosc, klimatyzacja)
VALUES("P012345",25000,2019,1.6,1);
```

```
INSERT INTO auta(nr_rej, przebieg, rok_prod, pojemnosc, klimatyzacja)
VALUES("ZS98765",67000,2014,1.8,1);
```

```
INSERT INTO auta(nr_rej, przebieg, rok_prod, pojemnosc, klimatyzacja)
VALUES("PSZ22233",12000,2020,1.6,1);
```

```
INSERT INTO auta(nr_rej, przebieg, rok_prod, pojemnosc, klimatyzacja)
VALUES("DW87872",150000,2001,2.0,0);
```

```
INSERT INTO auta(nr_rej, przebieg, rok_prod, pojemnosc, klimatyzacja)
VALUES("KR12321",75000,2005,2.2,0);
```

```
INSERT INTO wypożyczenia ( id_autu, pesel, data, dystans )
VALUES (1, "73100764862", "2021-10-10", 30);
```

```

INSERT INTO wypożyczenia ( id_auta, pesel, data, dystans )
VALUES (3, "86111651722", "2021-10-13", 50);

INSERT INTO wypożyczenia ( id_auta, pesel, data, dystans )
VALUES (4, "94011552087", "2021-10-14", 45);

INSERT INTO wypożyczenia ( id_auta, pesel, data, dystans )
VALUES (2, "73100764862", "2021-10-14", 35);

INSERT INTO wypożyczenia ( id_auta, pesel, data, dystans )
VALUES (2, "86111651722", "2021-10-20", 50);

INSERT INTO wypożyczenia ( id_auta, pesel, data, dystans )
VALUES (4, "71020373847", "2021-10-21", 70);

INSERT INTO wypożyczenia ( id_auta, pesel, data, dystans )
VALUES (2, "73100764862", "2021-10-25", 25);

```

### Zadanie 5.3.

```

UPDATE auta
SET przebieg = przebieg + 5000
WHERE rok_prod<2014;

```

### Zadanie 5.4.

```

SELECT TOP 1 rok_prod
FROM auta
ORDER BY rok_prod DESC;

```

### Zadanie 5.5.

```

SELECT auta.id, Avg(wypożyczenia.dystans) AS Średnia
FROM auta INNER JOIN wypożyczenia ON auta.id = wypożyczenia.id_auta
GROUP BY auta.id;

```

### Zadanie 5.6.

```

SELECT id_auta, count(id_auta) as Ilość
FROM wypożyczenia
GROUP BY id_auta;

```

### Zadanie 5.7.

Te auta, których nikt nie wypożyczał:

```

SELECT id
FROM auta
WHERE id NOT IN (SELECT id_auta FROM wypożyczenia)
id = 5
DELETE *
FROM auta
WHERE id = 5;

```

# Zestaw 4

## Zadanie 1. Bezpiecznie na nartach

W ośrodku narciarskim Śnieżynka dużą wagę przywiązuje się do bezpieczeństwa. Jednym z rozwiązań zainstalowanych w ośrodku, mającym podnieść poziom bezpieczeństwa i komfortu narciarzy, jest system monitoringu natężenia ruchu, wysokości i jakości pokrywy śnieżnej i warunków pogodowych na poszczególnych odcinkach tras zjazdowych. W ramach systemu na trasach zamontowane zostały urządzenia pomiarowe. Urządzenia te łączą się bezprzewodowo z serwerem, który znajduje się w budynku administracyjnym ośrodka. W celu zapewnienia poprawnej komunikacji pomiędzy urządzeniami a serwerem wysyłane informacje są w odpowiedni sposób kodowane.

Poniżej zamieszczony został kod rekurencyjnej procedury `odkoduj_rek`, której zadaniem jest analiza poprawności otrzymanego pakietu danych oraz odkodowanie zapisanej w pakiecie informacji.

### Specyfikacja:

Dane:

$n$  — długość (ilość liczb) pakietu danych

$p[1..n]$  — otrzymany pakiet danych, ciąg  $n$  liczb złożony z zer i jedynek, zapisany w tablicy jednowymiarowej

Wynik:

$m$  — w przypadku poprawnego pakietu długość odkodowanej informacji (ilość liczb), w przypadku pakietu niepoprawnego  $m = -1$

$w[1..m]$  — odkodowana informacja, ciąg  $m$  liczb złożony z zer i jedynek, zapisany w tablicy jednowymiarowej

procedura `odkoduj_rek(s)`

$i \leftarrow s$

$a \leftarrow i$

$z \leftarrow p[i]$

$m \leftarrow m + 1$

$w[m] \leftarrow z$

```
i ← i + 1
k ← false
dopóki (i ≤ n) oraz (k = false) wykonuj
    jeżeli p[i] = z
        i ← i + 1
        m ← m + 1
        w[m] ← z
    w przeciwnym razie
        k ← true
    m ← i - m
    jeżeli (i ≤ n)
        z ← p[i]
        b ← 1
    w przeciwnym razie
        b ← 0
    i ← i + 1
    k ← true
dopóki (i ≤ n) oraz (b < a) oraz (k = true) wykonuj
    jeżeli p[i] = z
        b ← b + 1
        i ← i + 1
    w przeciwnym razie
        k ← false
jeżeli b < a
    k ← false
jeżeli (k = true) oraz (i ≤ n)
    odkoduj_rek(i)
jeżeli k = false
    m ← -1

procedura odkoduj_rek(s)
    i ← s
    a ← i
    z ← p[i]
    m ← m+1
    w[m] ← z
    i ← i+1
    k ← false
    dopóki (i≤n) oraz (k=false) wykonuj
        jeżeli p[i]=z
            i ← i+1
            m ← m+1
            w[m] ← z
```

```

w przeciwnym razie
    k ← true

m ← i-m
jeżeli (i≤n)
    z ← p[i]
    b ← 1
w przeciwnym razie
    b ← 0
    i ← i+1
    k ← true
dopóki (i≤n) oraz (b<a) oraz (k=true) wykonuj
    jeżeli p[i]=z
        b ← b+1
        i ← i+1
w przeciwnym razie
    k ← false
jeżeli b<a
    k ← false
jeżeli (k=true) oraz (i≤n)
    odkoduj_rek(i)
jeżeli k=false
    m ← -1

```

Przed pierwszym wywołaniem funkcji odkoduj\_rek(1) zmienna  $m$  ma wartość 0. Zmienne  $i, s, a, z, b$  są typu całkowitego. Zmienna  $k$  jest typu logicznego. Przeanalizuj dokładnie powyższy kod i wykonaj zadania 1.1 – 1.5.

## Zadanie 1.1.

Uzupełnij tabelę:

Pakiet danych przed odkodowaniem		Wynik uruchomienia procedury odekoduj_rek(1)	
$n$	$p[1..n]$	$m$	$w[1..m]$
			w przypadku pakietu niepoprawnego wpisz ---
12	[0,0,0,1,1,1,0,1,0,1,1,0]	6	[0,0,0,0,0,1]
20	[0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,1,0,0]		
30	[1,1,1,0,0,0,1,1,0,0,1,1,1,0,0,0, 1,1,1,1,1,0,0,0,0,0,1,0,0,1]		
34	[0,0,0,0,0,1,1,1,1,1,0,1,1,0,0,0, 0,1,1,1,0,1,1,1,1,1,0,0,0,0,0,0,0]		
46	[1,1,1,1,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,0,0, 0,1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1]		

Miejsce na obliczenia

## Zadanie 1.2.

Na podstawie kodu procedury odkodu<sub>j</sub>\_rek napisz w pseudojęzyku lub w wybranym języku programowania nerekurencyjną procedurę odkodu<sub>j</sub>, w wyniku działania której dla tego samego pakietu  $p[1..n]$  otrzymamy ten sam wynik co w wyniku działania procedury odkodu<sub>j</sub>\_rek.

## Zadanie 1.3.

Określ stosunek liczby zer do liczby jedynek w poprawnym pakiecie.

Stosunek wynosi: .....

## Zadanie 1.4.

Jaki wynik otrzymamy, gdy procedura odkoduj\_rek otrzyma pakiet danych o długości 500, w którym pierwsze pięćdziesiąt liczb to zera, następnie sto to jedynki, kolejne sto to zera, kolejna setka to jedynki, następna setka to zera, a ostatnie pięćdziesiąt liczb to jedynki.

*m* = .....

Opisz zawartość tablicy  $w[i..m]$

.....  
.....  
.....

Miejsce na obliczenia

## Zadanie 1.5.

Niektóre urządzenia w systemie przesyłają liczby naturalne dodatnie, stosując inny sposób kodowania. Kodowanie odbywa się w trzech krokach:

- ◆ zamieniamy liczbę z systemu dziesiątkowego na system binarny,
  - ◆ liczbę wystąpień ostatniej cyfry podwajamy, przedostatniej cyfry potrajamy i tak dalej,
  - ◆ zamieniamy powstałą przez podwojenie zer i jedynek liczbę w systemie binarnym na system dziesiątkowy.

### Przykład:

*Liczba do zakodowania: 13*

Zamiana na system binarny:  $1310 = 11012$

Zwiększały wystąpienia poszczególnych cyfr. Ostatniej dwa razy, przedostatnich trzy razy, drugiej cztery razy i pierwszej pięć razy.

Wówczas otrzymamy liczbę: 11111111000112

Zamiana na system dziesiątkowy:  $11111111000112 = 1635510$

*Liczba 150 po zakodowaniu to liczba 16355*

Napisz w pseudojęzyku lub wybranym języku programowania algorytm, który zakoduje daną liczbę zgodnie z zasadami kodowania opisanyimi w zadaniu 1.5.



## Uwaga

W zapisie algorytmu możesz korzystać tylko z instrukcji sterujących, operatorów arytmetycznych: dodawania, odejmowania, mnożenia, dzielenia, dzielenia całkowitego i reszty z dzielenia; operatorów logicznych, porówniań i instrukcji przypisania lub samodzielnie napisanych funkcji i procedur wykorzystujących powyższe operacje. Zabronione jest używanie tablic, łańcuchów znaków itp., funkcji wbudowanych oraz operatorów innych niż wymienione, dostępnych w językach programowania.

## Specyfikacja:

Dane:

$n$  — dodatnia liczba naturalna do zakodowania

## Wynik:

$m$  — dodatnia liczba naturalna będąca wynikiem zakodowania liczby  $n$

## Zadanie 1.6.

Niech  $n$  i  $m$  będą liczbami opisanymi w specyfikacji w zadaniu 1.5. Niech  $c_n$  oznacza liczbę cyfr w zapisie binarnym liczby  $n$ , natomiast  $c_m$  niech oznacza liczbę cyfr w zapisie binarnym liczby  $m$ . Podaj wzór na  $c_m$  w zależności od  $c_n$ .

$C_m = \dots$

## Miejsce na obliczenia

## Zadanie 2. Nowoczesny rolnik

Pan Waclaw jest rolnikiem i uprawia różne rodzaje zbóż i roślin. Jego pole ma kształt kwadratu i jest podzielone na 256 mniejszych, jednohektarowych kwadratowych pól. Położenie poszczególnych poletek oraz dotychczasowe zabiegi pana Waclawa wpłynęły na ich klasę bonitacyjną. W Polsce wyróżniamy sześć klas bonitacyjnych określających jakość gleby. Gleby najlepsze mają klasę I, natomiast najsłabsze klasę VI. Dla uproszczenia klasy bonitacyjne bedziemy zapisywać za pomocą cyfr arabskich.

Nasz rolnik planuje zasiew na kolejny sezon; chciałby uprawiać różne gatunki zbóż i roślin. Rozważa także różną wielkość obszarów obsianych poszczególnymi gatunkami. W celu zoptymalizowania kosztów uprawy, poprawienia organizacji pracy postanowił, że obszary poszczególnych zasiewów będą także miały kształt kwadratu.

Chcąc wyznaczyć obszary o najwyższej średniej klasie bonitacyjnej, postąpił w następujący sposób: w pierwszym kroku podzielił całe pole na cztery sektory. W drugim kroku każdy sektor na kolejne cztery sektory. Dzielił tak długo, aż jeden sektor składał się tylko z czterech poletek. Pan Waclaw postanowił obliczyć średnie klasy bonitacji w poszczególnych kwadratowych obszarach i dla każdego podziału wyznaczyć sektor o najmniejszej średniej.

Poniższy przykład ilustruje sposób działania naszego rolnika. Przykładowe pole składa się z 256 poltek o następujących klasach bonitacji:

4	1	2	2	4	5	5	1	1	5	5	2	5	4	2	3
4	3	1	5	6	4	3	4	4	3	3	6	5	2	1	2
6	2	3	4	6	6	2	4	4	4	3	6	6	4	6	3
5	6	6	4	1	2	5	4	5	6	1	3	5	1	2	4
6	2	1	5	1	6	6	2	4	1	5	3	2	4	5	6
1	4	3	6	4	6	1	2	5	2	4	3	6	5	6	6
4	5	2	5	4	6	4	1	6	3	1	1	6	6	5	5
1	5	4	4	4	4	3	2	3	6	2	1	2	2	6	6
4	5	2	1	4	5	6	1	1	6	6	5	3	4	3	2
2	4	5	6	5	6	5	2	5	1	2	5	6	5	2	3
3	1	1	5	3	6	5	4	4	4	6	4	1	6	5	3
3	2	6	1	1	4	6	3	2	5	5	1	3	6	3	4
4	2	6	1	1	2	2	2	6	1	6	4	4	4	4	6
5	1	4	3	5	2	6	6	6	4	5	3	2	1	4	5
6	3	5	1	4	6	6	1	4	5	3	1	1	4	6	3
5	2	6	1	3	5	4	2	2	2	2	1	1	3	6	6

W pierwszym kroku obliczamy średnią klasę bonitacji całego pola. W naszym przykładzie wynosi ona 3,68359. Wszystkie średnie będziemy podawać z domyślną dokładnością, jaką wyświetla kompilator.

W drugim kroku dzielimy całe pole na cztery kwadratowe sektory.

4	1	2	2	4	5	5	1	1	5	5	2	5	4	2	3
4	3	1	5	6	4	3	4	4	3	3	6	5	2	1	2
6	2	3	4	6	6	2	4	4	4	3	6	6	4	6	3
5	6	6	4	1	2	5	4	5	6	1	3	5	1	2	4
6	2	1	5	1	6	6	2	4	1	5	3	2	4	5	6
1	4	3	6	4	6	1	2	5	2	4	3	6	5	6	6
4	5	2	5	4	6	4	1	6	3	1	1	6	6	5	5
1	5	4	4	4	4	3	2	3	6	2	1	2	2	6	6
4	5	2	1	4	5	6	1	1	6	6	5	3	4	3	2
2	4	5	6	5	6	5	2	5	1	2	5	6	5	2	3
3	1	1	5	3	6	5	4	4	4	6	4	1	6	5	3
3	2	6	1	1	4	6	3	2	5	5	1	3	6	3	4
4	2	6	1	1	2	2	2	6	1	6	4	4	4	4	6
5	1	4	3	5	2	6	6	6	4	5	3	2	1	4	5
6	3	5	1	4	6	6	1	4	5	3	1	1	4	6	3
5	2	6	1	3	5	4	2	2	2	2	1	1	3	6	6

W tym podziale sektory numerujemy następująco:

1	2
3	4

Następnie obliczamy średnią klas bonitacyjnych każdego sektora. W naszym przykładzie najniższą średnią wynoszącą 3,57812 ma sektor o numerze 3.

W kolejnym kroku każdy z czterech sektorów ponownie dzielimy na kolejne cztery kwadratowe sektory.

4	1	2	2	4	5	5	1	1	5	5	2	5	4	2	3
4	3	1	5	6	4	3	4	4	3	3	6	5	2	1	2
6	2	3	4	6	6	2	4	4	4	3	6	6	4	6	3
5	6	6	4	1	2	5	4	5	6	1	3	5	1	2	4
6	2	1	5	1	6	6	2	4	1	5	3	2	4	5	6
1	4	3	6	4	6	1	2	5	2	4	3	6	5	6	6
4	5	2	5	4	6	4	1	6	3	1	1	6	6	5	5
1	5	4	4	4	4	3	2	3	6	2	1	2	2	6	6
4	5	2	1	4	5	6	1	1	6	6	5	3	4	3	2
2	4	5	6	5	6	5	2	5	1	2	5	6	5	2	3
3	1	1	5	3	6	5	4	4	4	6	4	1	6	5	3
3	2	6	1	1	4	6	3	2	5	5	1	3	6	3	4
4	2	6	1	1	2	2	2	6	1	6	4	4	4	4	6
5	1	4	3	5	2	6	6	6	4	5	3	2	1	4	5
6	3	5	1	4	6	6	1	4	5	3	1	1	4	6	3
5	2	6	1	3	5	4	2	2	2	2	1	1	3	6	6

W tym podziale sektory numerujemy następująco:

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

Obliczamy średnią klas bonitacyjnych każdego sektora. Spośród 16 sektorów najniższą średnią równą 3,125 ma sektor o numerze 7.

W kolejnym kroku powtarzamy podział.

4	1	2	2	4	5	5	1	1	5	5	2	5	4	2	3
4	3	1	5	6	4	3	4	4	3	3	6	5	2	1	2
6	2	3	4	6	6	2	4	4	4	3	6	6	4	6	3
5	6	6	4	1	2	5	4	5	6	1	3	5	1	2	4
6	2	1	5	1	6	6	2	4	1	5	3	2	4	5	6
1	4	3	6	4	6	1	2	5	2	4	3	6	5	6	6
4	5	2	5	4	6	4	1	6	3	1	1	6	6	5	5
1	5	4	4	4	4	3	2	3	6	2	1	2	2	6	6
4	5	2	1	4	5	6	1	1	6	6	5	3	4	3	2
2	4	5	6	5	6	5	2	5	1	2	5	6	5	2	3
3	1	1	5	3	6	5	4	4	4	6	4	1	6	5	3
3	2	6	1	1	4	6	3	2	5	5	1	3	6	3	4
4	2	6	1	1	2	2	2	6	1	6	4	4	4	4	6
5	1	4	3	5	2	6	6	6	4	5	3	2	1	4	5
6	3	5	1	4	6	6	1	4	5	3	1	1	4	6	3
5	2	6	1	3	5	4	2	2	2	2	1	1	3	6	6

Jest to ostatni podział. Każdy z 64 sektorów składa się z czterech jednohektarowych poletek. Numeracja sektorów wygląda następująco:

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

Po obliczeniu średnich klas w poszczególnych 64 sektorach okazuje się, że najniższa średnia wynosząca 1,25 jest w sektorze o numerze 30.

Podsumowując, mamy:

**Liczba sektorów = 1 Najmniejsza średnia = 3,68359 w kwadracie 1/1**

**Liczba sektorów = 4 Najmniejsza średnia = 3,57812 w kwadracie 3/4**

**Liczba sektorów = 16 Najmniejsza średnia = 3,125 w kwadracie 7/16**

**Liczba sektorów = 64 Najmniejsza średnia = 1,25 w kwadracie 30/64**

W pliku *pole\_waclawa.txt* znajdują się informacje dotyczące klas bonitacyjnych poletek należących do naszego rolnika. W pliku znajduje się 16 wierszy, a w każdym wierszu zapisano 16 liczb z przedziału od 1 do 6, oznaczających klasy bonitacyjne. Liczby w wierszu oddzielone są spacją.

## Zadanie 2.1.

W wybranym języku programowania napisz program **niewykorzystujący rekurencji**, który na podstawie informacji o polu pana Wacława, zawartych w pliku *pole\_waclawa.txt*, wypisze podsumowanie analogiczne do podsumowania z przykładu.

## Zadanie 2.2.

W wybranym języku programowania napisz program **wykorzystujący rekurencję przy wyznaczaniu poszczególnych kwadratów**, który na podstawie informacji o polu pana Wacława, zawartych w pliku *pole\_waclawa.txt*, wypisze podsumowanie analogiczne do podsumowania z przykładu.

## Zadanie 2.3.

W pliku *pola.txt* znajdują się dane piętnastu kwadratowych pól składających się z 16, 64, 256, 1024 lub 4096 hektarowych poletek. Informacje dotyczące klas bonitacji poletek należących do danego pola zapisane są w sposób analogiczny do sposobu zapisu informacji o polu pana Wacława. Miedzy danymi poszczególnych pól wstawiony został pusty wiersz.

W wybranym języku programowania napisz program, który dokona dla każdego z pól podsumowania przedstawionego w przykładzie.

## Zadanie 3. Statystyka wypadków

W urzędzie statystycznym doszło do awarii serwera, na którym przechowywane były niektóre dane. Niestety część danych została bezpowrotnie utracona. Należała do nich informacja o liczbie wypadków w ciągu dnia w poszczególnych powiatach w okresie wakacyjnym, od 1 lipca 2020 do 31 sierpnia 2020. Przed awarią w bazie znajdowały się informacje o liczbie wypadków każdego dnia w wymienionym okresie w każdym powiecie Polski. Te dane dotyczące liczby wypadków, które udało się odzyskać, zapisane są w pliku *wypadki.txt*.

Każdy wiersz pliku zawiera następujące informacje: dwuliterowe lub trzyliterowe oznaczenie powiatu (stosowane w numerach rejestracyjnych pojazdów), dzień (data), liczba wypadków, które miały miejsce w danym dniu na obszarze danego powiatu. Dane w wierszu oddzielone są spacją.

**Przykładowy fragment pliku *wypadki.txt***

DZA 2020-08-05 9  
NEB 2020-08-24 4  
ZDR 2020-07-20 0  
SR 2020-08-07 5  
RKL 2020-08-22 5  
EZG 2020-08-24 10  
RNI 2020-07-02 8  
SMI 2020-07-03 3  
NE 2020-07-18 3  
DGR 2020-07-05 0

Na podstawie danych zawartych w pliku *wypadki.txt* napisz w wybranym języku programowania program (lub programy), który pozwoli udzielić odpowiedzi lub wykonać poleceń zamieszczone w zadaniach 3.1 – 3.3.

## Zadanie 3.1.

Pierwsza litera oznaczenia powiatu informuje nas, w jakim województwie leży dany powiat.

Dolnośląskie	D	Podkarpackie	R
Kujawsko-pomorskie	C	Podlaskie	B
Lubelskie	L	Pomorskie	G
Lubuskie	F	Śląskie	S
Łódzkie	E	Świętokrzyskie	T
Małopolskie	K	Warmińsko-mazurskie	N
Mazowieckie	W	Wielkopolskie	P
Opolskie	O	Zachodniopomorskie	Z

Poprzez jedną informację o wypadkach będziemy rozumieć jeden wiersz w pliku. Z którego województwa w pliku zachowało się najmniej informacji? Podaj oznaczenie województwa oraz liczbę informacji.

## Zadanie 3.2.

Dla każdego województwa wypisz oznaczenie powiatu, w którym łączna liczba wypadków była największa. Jeżeli takich powiatów jest więcej niż jeden, to wypisz wszystkie.

## Zadanie 3.3.

Pracownicy urzędu statystycznego zastanawiali się, czy średnia dzienna liczba wypadków w danym powiecie obliczona na podstawie wybrakowanych danych będzie zbliżona do średniej obliczonej na podstawie kompletnych danych ze wszystkich dni. Z tym problemem zwróciły się do zaprzyjaźnionych matematyków. Ci ostatni stwierdzili, że taka średnia będzie zbliżona, jeśli z każdej trzeciej części miesiąca (dni 1 – 10, dni 11 – 20, dni 21 – 31) zachowały się dane z co najmniej siedmiu dni.

W przypadku których powiatów spełniony będzie warunek określony przez matematyków?

## Zadanie 4. Struktura książki

Firma programistyczna Ebuczek zajmuje się tworzeniem e-booków z różnych dziedzin nauki i życia. Niedawno podjęła współpracę z grupą autorów, którzy chcą stworzyć podręcznik do informatyki. W pierwszej kolejności autorzy, po wielodniowych naradach, ustalili wstępную zawartość książki — działy, tematy, podtematy.

Swoje ustalenia zapisali w pliku tekstowym *cms.txt* (do dyspozycji jest dodatkowy plik *cms2.txt* z większą ilością danych). Poniższy przykład obrazuje sposób zapisu działów, tematów i podtematów.

**Przykład:**

Następujący fragment układu treści:

*Informatyka*  
*Programowanie*  
    *Python*  
    *C++*  
*Sieci komputerowe*  
    *Model ISO OSI*  
*Grafika komputerowa*  
*Programowanie aplikacji*  
    *HTML i CSS*  
    *JS*  
    *Node*  
    *PHP*

zapisany w pliku tekstowym ma postać:

*symbol;temat*  
*a1;Informatyka*  
*a1a1;Programowanie*  
*a1a2;Sieci komputerowe*  
*a1a3;Grafika komputerowa*  
*a1a1a1;Python*  
*a1a1a2;C++*  
*a1a2a1;Model ISO OSI*  
*a1a4;Programowanie aplikacji*  
*a1a4a1;HTML i CSS*  
*a1a4a2;JS*  
*a1a4a3;PHP*  
*a1a4a2a1;Node*

Poziom danego tematu określony jest poprzez długość ciągu znaków postaci  $a_1a_2a_3\dots a_m$  oddzielony od samego tematu średnikiem. Kolejność poszczególnych tematów oraz relację podrzedny – nadrzędny określają liczby naturalne  $n_1, n_2, n_3, \dots, n_m$  należące do przedziału  $(1; 9)$ .

Temat *Informatyka* jest tematem na pierwszym poziomie. Temat *C++* jest tematem określonym na poziomie trzecim. Temat: *Programowanie aplikacji* ma trzy podtematy bezpośrednie (HTML i CSS, JS i PHP).

Po przeanalizowaniu powyższego przykładu **za pomocą arkusza kalkulacyjnego**, na podstawie danych zawartych w pliku *cms.txt* udziel odpowiedzi lub wykonaj polecenia z zadań 4.1 – 4.4.

## Zadanie 4.1.

Ile poziomów zagnieździania tematów wykorzystano?

## Zadanie 4.2.

Który temat ma najwięcej bezpośrednich podtematów? Ile jest tych podtematów?

## Zadanie 4.3.

Utwórz zestawienie zawierające dla każdego podtematu tematu pierwszego poziomu liczbę jego podtematów bezpośrednich. Na podstawie tego zestawienia utwórz wykres kolumnowy. Pamiętaj o czytelności wykresu.

## Zadanie 4.4.

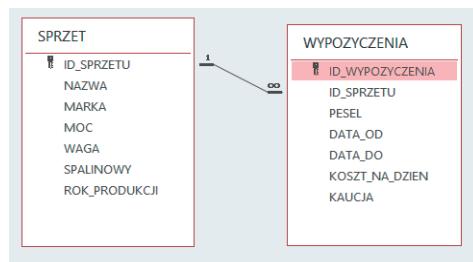
Utwórz zestawienie zawierające liczbę wszystkich tematów na poszczególnych poziomach zagnieźdzania.

W naszym przykładzie:

- poziom 1 — jeden temat,*
- poziom 2 — cztery tematy,*
- poziom 3 — sześć tematów,*
- poziom 4 — jeden temat.*

## Zadanie 5. Piszemy w SQL-u

W bazie danych pewnej wypożyczalni narzędzi istnieją między innymi następujące tabele:



Powyższy schemat obrazuje relację pomiędzy tabelami.

W tabeli SPRZET zdefiniowano następująco pola:

ID_SPRZETU	identyfikator narzędzia, klucz główny (liczba całkowita)
NAZWA	nazwa narzędzia (tekst)
MARKA	nazwa producenta (tekst)
MOC	moc narzędzia (liczba całkowita)
WAGA	waga narzędzia (liczba całkowita)

<b>SPALINOWY</b>	informacja o rodzaju zasilania, wartość TRUE — oznacza narzędzie o silniku spalinowym, FALSE — silnik elektryczny
<b>ROK PRODUKCJI</b>	rok produkcji (liczba całkowita)

W tabeli WYPOZYCZENIA zdefiniowano następujące pola:

<b>ID_WYPOZYCZENIA</b>	identyfikator wypożyczenia, klucz główny (liczba całkowita)
<b>ID_SPRZETU</b>	identyfikator wypożyczenia, klucz obcy (liczba całkowita)
<b>PESEL</b>	numer PESEL wypożyczającego (tekst)
<b>DATA_OD</b>	pierwszy dzień okresu wypożyczenia (data)
<b>DATA_DO</b>	ostatni dzień okresu wypożyczenia (data)
<b>KOSZT_NA_DZIEN</b>	koszt jednego dnia wypożyczenia (liczba rzeczywista)
<b>KAUCJA</b>	kaucja za sprzęt (liczba całkowita)

W poniższych zadaniach napisz w języku SQL zapytania, które dadzą odpowiedź na postawione pytania lub wykonają zamieszczone polecenia.

## Zadanie 5.1.

Wypisz nazwy wszystkich marek sprzętu, który można wypożyczyć.

## Zadanie 5.2.

W którym roku wyprodukowano najnowszy sprzęt dostępny wypożyczalni? Ile jest sztuk narzędzi z tego roku?

## Zadanie 5.3.

Dla każdego sprzętu podaj kwotę, która wpłynęła do wypożyczalni z tytułu jego wszystkich wypożyczeń. Zestawienie posortuj malejaco według kwot.

## Zadanie 5.4.

Wypisz identyfikatory oraz nazwy narzędzi, których nikt nigdy nie wypożyczył.

## Zadanie 5.5.

Wypisz numery PESEL klientów wraz z nazwą wypożyczanego sprzętu, którego okres wypożyczenia mieścił się w listopadzie 2022 roku.

## Zadanie 5.6.

W którym dniu wypożyczono najwięcej sprzętu?

## Zadanie 5.7.

Wypisz nazwy narzędzi, których moc jest większa od średniej mocy wszystkich narzędzi dostępnych wypożyczalni.

## Zadanie 5.8.

Wypisz identyfikatory oraz nazwy narzędzi, które zostały wypożyczone co najmniej dwa razy, a kaucja przy tych wypożyczeniach nie była niższa niż 100 złotych.

## Zadanie 5.9.

Ille jest narzędzi, których nazwy zawierają słowo „wiertarka”?

## Zadanie 5.10.

Podaj numer PESEL klienta, który wypożyczył najwięcej narzędzi spalinowych.

## Zadanie 5.11.

Wypisz identyfikatory i nazwy narzędzi, w przypadku których kaucja pobierana podczas ich wypożyczania nie zawsze była taka sama.

## Zadanie 5.12.

Dla każdego producenta wypisz narzędzia o napędzie spalinowym, których moc wynosi co najmniej 1000 W.

### Zadanie 5.13.

Wypisz dni, w których wypożyczalnię odwiedziła kobieta i wypożyczyła co najmniej jedno narzędzie.

### Zadanie 5.14.

Ile jest narzędzi, których koszt dziennego wypożyczenia jest większy od kaucji?

## Zadanie 5.15.

Podaj nazwę i markę sprzętu wypożyczonego jako ostatnie.

## Zadanie 5.16.

Ile razy wypożyczano narzędzi na jeden dzień?

## Zadanie 5.17.

Podaj nazwę, markę i moc narzędzia o napędzie spalinowym o największej mocy.

# Odpowiedzi i wskazówki do zestawu 4

## Zadanie 1.1.

Pakiet danych przed odkodowaniem		Wynik uruchomienia procedury odkoduj_rek(1)	
<i>n</i>	<i>p[1..n]</i>	<i>m</i>	<i>w[1..m]</i>
			w przypadku pakietu niepoprawnego wpisz ---
12	[0,0,0,1,1,1,0,1,0,1,1,0]	6	[0,0,0,0,0,1]
20	[0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,1,0,0]	10	[0,0,0,0,0,0,0,0,1,1]
30	[1,1,1,0,0,0,1,1,0,0,1,1,1,0,0,0, 1,1,1,1,0,0,0,0,0,1,0,0,1]	15	[1,1,1,1,1,1,1,1, 1,1,1,1,1,1,0]
34	[0,0,0,0,0,1,1,1,1,1,1,0,1,1,0,0,0, 0,1,1,1,0,1,1,1,1,1,0,0,0,0,0,0,0]	-1	---
46	[1,1,1,1,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,0, 0,1,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1]	23	[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,0,0,0,0,0,0,0,0,0,0]

## Zadanie 1.2.

Przykładowe rozwiązanie:

procedura odkoduj;

*m*  $\leftarrow$  0

*r*  $\leftarrow$  true

*i*  $\leftarrow$  1

dopóki (*i*  $\leq$  *n*) oraz (*r* = true) wykonuj

*a*  $\leftarrow$  *i*

*z*  $\leftarrow$  *p[i]*

*m*  $\leftarrow$  *m* + 1

*w[m]*  $\leftarrow$  *z*

*i*  $\leftarrow$  *i* + 1

*k*  $\leftarrow$  false

dopóki (*i*  $\leq$  *n*) oraz (*k*=false) wykonuj

        jeżeli *p[i]*=*z*

*i*  $\leftarrow$  *i* + 1

*m*  $\leftarrow$  *m* + 1

*w[m]*  $\leftarrow$  *z*

        w przeciwnym razie

*k*  $\leftarrow$  true

*m*  $\leftarrow$  *i* - *m*

jeżeli ( $i \leq n$ )  
 $z \leftarrow p[i]$   
 $b \leftarrow 1$   
w przeciwnym razie  
 $b \leftarrow 0$   
 $i \leftarrow i + 1$   
 $r \leftarrow \text{true}$   
dopóki ( $i \leq n$ ) oraz ( $b < a$ ) oraz ( $r = \text{true}$ ) wykonuj  
jeżeli  $p[i] = z$   
 $b \leftarrow b + 1$   
 $i \leftarrow i + 1$   
w przeciwnym razie  
 $r \leftarrow \text{false}$   
jeżeli  $b < a$   
 $r \leftarrow \text{false}$   
 $m \leftarrow -1$

procedura odkoduj;  
 $m \leftarrow 0$   
 $r \leftarrow \text{true}$   
 $i \leftarrow 1$   
**dopóki** ( $i \leq n$ ) oraz ( $r = \text{true}$ ) **wykonuj**  
     $a \leftarrow i$   
     $z \leftarrow p[i]$   
     $m \leftarrow m+1$   
     $w[m] \leftarrow z$   
     $i \leftarrow i+1$   
     $k \leftarrow \text{false}$   
    **dopóki** ( $i \leq n$ ) oraz ( $k = \text{false}$ ) **wykonuj**  
        jeżeli  $p[i] = z$   
             $i \leftarrow i+1$   
             $m \leftarrow m+1$   
             $w[m] \leftarrow z$   
        w przeciwnym razie  
             $k \leftarrow \text{true}$   
     $m \leftarrow i-m$   
    jeżeli ( $i \leq n$ )  
         $z \leftarrow p[i]$   
         $b \leftarrow 1$   
    w przeciwnym razie  
         $b \leftarrow 0$   
     $i \leftarrow i+1$   
     $r \leftarrow \text{true}$   
**dopóki** ( $i \leq n$ ) oraz ( $b < a$ ) oraz ( $r = \text{true}$ ) **wykonuj**  
    jeżeli  $p[i] = z$   
         $b \leftarrow b+1$   
         $i \leftarrow i+1$   
    w przeciwnym razie  
         $r \leftarrow \text{false}$

```

jeżeli b<a
    r ← false
    m ← -1

```

### Zadanie 1.3.

Liczba zer jest równa liczbie jedynek. Stosunek wynosi 1.

### Zadanie 1.4.

$m = 250$ ,

pięćdziesiąt zer, pięćdziesiąt jedynek, pięćdziesiąt zer, pięćdziesiąt jedynek, pięćdziesiąt zer.

### Zadanie 1.5.

Przykładowe rozwiązanie:

$a \leftarrow 1$

$k \leftarrow 0$

$p \leftarrow 2$

dopóki  $n > 0$  wykonuj

$b \leftarrow n \bmod 2$

$n \leftarrow n \div 2$

dla  $i = 1, 2, \dots, p$  wykonuj

$k \leftarrow k + b * a$

$a \leftarrow a * 10$

$p \leftarrow p+1$

$m \leftarrow 0$

$a \leftarrow 1$

dopóki  $k > 0$  wykonuj

$b \leftarrow k \bmod 10$

$k \leftarrow k \div 10$

$m \leftarrow m + b * a$

$a \leftarrow a * 2$

$a \leftarrow 1$

$k \leftarrow 0$

$p \leftarrow 2$

**dopóki**  $n>0$  **wykonuj**

$b \leftarrow n \bmod 2$

$n \leftarrow n \div 2$

**dla**  $i=1,2,\dots,p$  **wykonuj**

$k \leftarrow k+b*a$

$a \leftarrow a*10$

```

    p ← p+1
    m ← 0
    a ← 1
dopóki k>0 wykonuj
    b ← k mod 10
    k ← k div 10
    m ← m+b*a
    a ← a*2

```

## Zadanie 1.6.



Uwaga

Ostatnia cyfra będzie powtórzona 2 razy, przedostatnia 3 razy i tak dalej. Pierwsza cyfra będzie powtórzona  $c_n + 1$  razy. Zatem liczba  $c_m$  jest sumą wyrazów skończonego ciągu arytmetycznego  $2, 3, 4, \dots, c_n, c_n + 1$ .

$$c_m = \frac{c_n(c_n+3)}{2}$$

## Zadanie 2.

W programach zostały zadeklarowane następujące zmienne i struktury:

```

#include <iostream>
#include <windows.h>
#include <fstream>
using namespace std;
string s, s2, s3, s4;
int i, j, k, rozmiar, wiersz, liczba, liczba_kw, liczba_kw_wymiar, lg_w, lg_k,
    n, numer_kw, suma;
double srednia;
int liczba_podzialow;
fstream plik1, plik2, plik3;
int dane[1000][1000];
double najmniejsze_sr[100];
int numery_kw_najmniejszych_sr[100];

```

## Zadanie 2.1.

```

int main()
{
    plik2.open("wynik.txt", ios::out);
    plik1.open("pole_waclawa.txt", ios::in);

    wiersz=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                {
                    wiersz++;
                    i=0;
                    s2=s;
                    while (s2!="")
                        {k=s2.find(" ");

```

```

        if (k>0)
            {s3=s2.substr(0,k);
             liczba=atoi(s3.c_str());
             s2=s2.substr(k+1);
            }
        else
            {liczba=atoi(s2.c_str());
             s2="";
            }
        i++;
        dane[wiersz][i]=liczba;
    }

    if (wiersz==1)
    {rozmiar=i;
    }

}
}

plik1.close();

cout << "Rozmiar = " << rozmiar << endl;
for (i=1; i<=rozmiar; i++)
{for (j=1; j<=rozmiar; j++)
    {cout << dane[i][j] << " ";
     }
    cout << endl;
}

//podzial na kwadraty

liczba_podzialow=0;

i=rozmiar;
while (i>1)
{
    liczba_kw_wymiar = (rozmiar/i);
    liczba_kw = liczba_kw_wymiar*liczba_kw_wymiar;

    liczba_podzialow++;
    najmniejsze_sr[liczba_podzialow]=7;

    numer_kw=0;
    for (j=1; j<=liczba_kw_wymiar; j++)
    {for (k=1; k<=liczba_kw_wymiar; k++)
        {lg_w=1+(j-1)*i;
         lg_k=1+(k-1)*i;

        numer_kw++;

        suma=0;
        for (m=lg_w; m<lg_w+i; m++)
        {for (n=lg_k; n<lg_k+i; n++)
            {suma=suma+dane[m][n];
            }
        }
        srednia=(suma*1.0)/(i*i);
    }
}
```

```

        if (srednia<najmniejsze_sr[liczba_podzialow])
            {najmniejsze_sr[liczba_podzialow]=srednia;
             numery_kw_najmniejszych_sr[liczba_podzialow]=numer_kw;
            }
        }

        i=i/2;
    }

cout << endl;
cout << "Najmniejsze srednie: " << endl;
k=1;
for (i=1; i<=liczba_podzialow; i++)
    {cout << "Liczba sektorow = " << k << "   Najmniejsza srednia = "
     ><< najmniejsze_sr[i] << " w kwadracie " << numery_kw_najmniejszych_sr[i]
     ><< "/" << k << endl;
k=k*4;
}

plik2 << "Najmniejsze srednie: " << endl;
k=1;
for (i=1; i<=liczba_podzialow; i++)
    {plik2 << "Liczba sektorow = " << k << "   Najmniejsza srednia = "
     ><< najmniejsze_sr[i] << " w kwadracie " << numery_kw_najmniejszych_sr[i]
     ><< "/" << k << endl;
k=k*4;
}

plik2.close();
cout << "GOTOWE" << endl;
getchar();
return 0;
}

```

**Odpowiedź:**

Najmniejsze średnie:

Liczba sektorow = 1 Najmniejsza srednia = 4.17578 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 3.85938 w kwadracie 2/4

Liczba sektorow = 16 Najmniejsza srednia = 2.5 w kwadracie 3/16

Liczba sektorow = 64 Najmniejsza srednia = 1 w kwadracie 64/64

**Zadanie 2.2.**

```

void wyznacz_srednie(int ww, int kk, int rr)
{
    int nowy_rr, numer, i, j, suma, k, pom;
    double srednia;

    numer = (ww/rr)*(rozmiar/rr)+1+(kk/rr);
    //cout << "Rozmiar = " << rr << "  poczatek kwadratu: " << ww << "," << kk << "
    ↳numer = " << numer << endl;
}

```

```
//obliczenie średniej
suma=0;
for (i=ww; i<=ww+rr-1; i++)
{for (j=kk; j<=kk+rr-1; j++)
{suma=suma+dane[i][j];
}
}
srednia = (suma*1.0)/(rr*rr);

pom=rr;
k=1;
while (pom<rozmiar)
{pom=pom*2;
k++;
}
if (srednia<najmniejsze_sr[k])
{najmniejsze_sr[k]=srednia;
numery_kw_najmniejszych_sr[k]=numer;
}

nowy_rr = rr/2;

if (nowy_rr>1)
{wyznacz_srednie(ww, kk, nowy_rr);
wyznacz_srednie(ww, kk+nowy_rr, nowy_rr);
wyznacz_srednie(ww+nowy_rr, kk , nowy_rr);
wyznacz_srednie(ww+nowy_rr, kk+nowy_rr, nowy_rr);
}

}

int main()
{
plik2.open("wynik.txt", ios::out);
plik1.open("pole_waclawa.txt", ios::in);

wiersz=0;
if(plik1.good() == true)
{while(!plik1.eof())
{getline(plik1, s);
if (s!="")
{
wiersz++;
i=0;
s2=s;
while (s2!="")
{k=s2.find(" ");
if (k>0)
{s3=s2.substr(0,k);
liczba=atoi(s3.c_str());
s2=s2.substr(k+1);
}
else
{liczba=atoi(s2.c_str());
s2="";
}
i++;
dane[wiersz][i]=liczba;
}
}
}
}
```

```
    }

    if (wiersz==1)
    {rozmiar=i;
    }

}

}

plik1.close();
cout << "Rozmiar = " << rozmiar << endl;
for (i=1; i<=rozmiar; i++)
{for (j=1; j<=rozmiar; j++)
{cout << dane[i][j] << " ";
}
cout << endl;
}

liczba_podzialow=0;
i=1;
while (i<rozmiar)
{i=i*2;
liczba_podzialow++;
najmniejsze_sr[liczba_podzialow]=7;
}

wyznacz_srednie(1,1,rozmiar);

cout << endl << "Najmniejsze srednie: " << endl;
k=1;
for (i=1; i<=liczba_podzialow; i++)
{cout << "Liczba sektorow = " << k << " Najmniejsza srednia = "
<< najmniejsze_sr[i] << " w kwadracie " << numery_kw_najmniejszych_sr[i]
<< "/" << k << endl;
k=k*4;
}

plik2 << "Najmniejsze srednie: " << endl;
k=1;
for (i=1; i<=liczba_podzialow; i++)
{plik2 << "Liczba sektorow = " << k << " Najmniejsza srednia = "
<< najmniejsze_sr[i] << " w kwadracie " << numery_kw_najmniejszych_sr[i]
<< "/" << k << endl;
k=k*4;
}

plik2.close();
cout << "GOTOWE" << endl;
getchar();
return 0;
}
```

**Odpowiedź:**

Najmniejsze średnie:

Liczba sektorow = 1 Najmniejsza srednia = 4.17578 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 3.85938 w kwadracie 2/4

Liczba sektorow = 16 Najmniejsza srednia = 2.5 w kwadracie 3/16

Liczba sektorow = 64 Najmniejsza srednia = 1 w kwadracie 64/64

## Zadanie 2.3.

```

int main()
{
    plik2.open("wyniki.txt", ios::out);
    plik1.open("pola.txt", ios::in);
    liczba_pol=1;
    wiersz=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                 {
                     wiersz++;
                     i=0;
                     s2=s;
                     while (s2!="")
                         {k=s2.find(" ");
                          if (k>0)
                              {s3=s2.substr(0,k);
                               liczba=atoi(s3.c_str());
                               s2=s2.substr(k+1);
                           }
                          else
                              {liczba=atoi(s2.c_str());
                               s2="";
                           }
                          i++;
                          dane[liczba_pol][wiersz][i]=liczba;
                      }
                 }
             if (wiersz==1)
                 {rozmiary_pol[liczba_pol]=i;
                  }
            }
        else
            {liczba_pol++;
             wiersz=0;
            }
        }
    plik1.close();

    for (m=1; m<=liczba_pol; m++)
        {cout << endl << "Pole numer = " << m << " Rozmiar = " << rozmiary_pol[m]
         << endl;
         for (i=1; i<=rozmiary_pol[m]; i++)
             {for (j=1; j<=rozmiary_pol[m]; j++)
                  {cout << dane[m][i][j] << " ";
                   }
                 cout << endl;
              }
         }
}

```

```
for (p=1; p<=liczba_pol; p++)
{
    liczba_podzialow=0;

    i=rozmiary_pol[p];
    while (i>1)
    {
        liczba_kw_wymiar = (rozmiary_pol[p]/i);
        liczba_kw = liczba_kw_wymiar*liczba_kw_wymiar;

        liczba_podzialow++;
        najmniejsze_sr[liczba_podzialow]=7;

        numer_kw=0;
        for (j=1; j<=liczba_kw_wymiar; j++)
            {for (k=1; k<=liczba_kw_wymiar; k++)
                {lg_w=1+(j-1)*i;
                 lg_k=1+(k-1)*i;

                numer_kw++;
                suma=0;
                for (m=lg_w; m<lg_w+i; m++)
                    {for (n=lg_k; n<lg_k+i; n++)
                        {suma=suma+dane[p][m][n];
                         }
                     }
                srednia=(suma*1.0)/(i*i);
                if (srednia<najmniejsze_sr[liczba_podzialow])
                    {najmniejsze_sr[liczba_podzialow]=srednia;
                     numery_kw_najmniejszych_sr[liczba_podzialow]=numer_kw;
                     }
                }
            }

        i=i/2;
    }

    cout << endl;
    cout << "Najmniejsze srednie dla pola numer: " << p << endl;
    k=1;
    for (i=1; i<=liczba_podzialow; i++)
        {cout << "Liczba sektorow = " << k << "   Najmniejsza srednia = "
         << najmniejsze_sr[i] << " w kwadracie " << numery_kw_najmniejszych_sr[i]
         << "/" << k << endl;
    k=k*4;
    }

plik2 << endl;
plik2 << "Najmniejsze srednie dla pola numer: " << p << endl;
k=1;
for (i=1; i<=liczba_podzialow; i++)
    {plik2 << "Liczba sektorow = " << k << "   Najmniejsza srednia = "
     << najmniejsze_sr[i] << " w kwadracie " << numery_kw_najmniejszych_sr[i]
     << "/" << k << endl;
    k=k*4;
    }
}

plik2.close();
```

```
cout << "GOTOWE" << endl;
getchar();
return 0;
}
```

**Odpowiedź:**

Najmniejsze średnie dla pola numer: 1

Liczba sektorow = 1 Najmniejsza srednia = 4.41797 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 3.5625 w kwadracie 4/4

Liczba sektorow = 16 Najmniejsza srednia = 2.9375 w kwadracie 12/16

Liczba sektorow = 64 Najmniejsza srednia = 1.5 w kwadracie 38/64

Najmniejsze średnie dla pola numer: 2

Liczba sektorow = 1 Najmniejsza srednia = 4.60938 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 4.51172 w kwadracie 3/4

Liczba sektorow = 16 Najmniejsza srednia = 4.01562 w kwadracie 6/16

Liczba sektorow = 64 Najmniejsza srednia = 3 w kwadracie 33/64

Liczba sektorow = 256 Najmniejsza srednia = 1.25 w kwadracie 146/256

Najmniejsze średnie dla pola numer: 3

Liczba sektorow = 1 Najmniejsza srednia = 3.6875 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 2.5 w kwadracie 4/4

Najmniejsze średnie dla pola numer: 4

Liczba sektorow = 1 Najmniejsza srednia = 4.52539 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 4.34473 w kwadracie 3/4

Liczba sektorow = 16 Najmniejsza srednia = 4.23438 w kwadracie 9/16

Liczba sektorow = 64 Najmniejsza srednia = 3.51562 w kwadracie 52/64

Liczba sektorow = 256 Najmniejsza srednia = 2.3125 w kwadracie 8/256

Liczba sektorow = 1024 Najmniejsza srednia = 0.25 w kwadracie 651/1024

Najmniejsze średnie dla pola numer: 5

Liczba sektorow = 1 Najmniejsza srednia = 4.9375 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 4.4375 w kwadracie 3/4

Liczba sektorow = 16 Najmniejsza srednia = 2.25 w kwadracie 3/16

Najmniejsze średnie dla pola numer: 6

Liczba sektorow = 1 Najmniejsza srednia = 4.51123 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 4.43066 w kwadracie 1/4

Liczba sektorow = 16 Najmniejsza srednia = 4.17188 w kwadracie 6/16

Liczba sektorow = 64 Najmniejsza srednia = 3.54688 w kwadracie 34/64

Liczba sektorow = 256 Najmniejsza srednia = 2.9375 w kwadracie 54/256

Liczba sektorow = 1024 Najmniejsza srednia = 0.25 w kwadracie 928/1024

Najmniejsze średnie dla pola numer: 7

Liczba sektorow = 1 Najmniejsza srednia = 4.67578 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 4.4375 w kwadracie 2/4

Liczba sektorow = 16 Najmniejsza srednia = 3.5 w kwadracie 2/16

Liczba sektorow = 64 Najmniejsza srednia = 0.75 w kwadracie 12/64

Najmniejsze średnie dla pola numer: 8

Liczba sektorow = 1 Najmniejsza srednia = 4.82812 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 3.75 w kwadracie 1/4

Liczba sektorow = 16 Najmniejsza srednia = 1.5 w kwadracie 2/16

Najmniejsze średnie dla pola numer: 9

Liczba sektorow = 1 Najmniejsza srednia = 4.37891 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 4.08594 w kwadracie 2/4

Liczba sektorow = 16 Najmniejsza srednia = 3.71875 w kwadracie 3/16

Liczba sektorow = 64 Najmniejsza srednia = 2.6875 w kwadracie 14/64

Liczba sektorow = 256 Najmniejsza srednia = 0.75 w kwadracie 48/256

Najmniejsze średnie dla pola numer: 10

Liczba sektorow = 1 Najmniejsza srednia = 4.4375 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 3 w kwadracie 2/4

Najmniejsze średnie dla pola numer: 11

Liczba sektorow = 1 Najmniejsza srednia = 4.4375 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 3.9375 w kwadracie 3/4

Liczba sektorow = 16 Najmniejsza srednia = 2.25 w kwadracie 12/16

Najmniejsze średnie dla pola numer: 12

Liczba sektorow = 1 Najmniejsza srednia = 4.51636 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 4.43164 w kwadracie 4/4

Liczba sektorow = 16 Najmniejsza srednia = 4.17969 w kwadracie 6/16

Liczba sektorow = 64 Najmniejsza srednia = 3.65625 w kwadracie 64/64

Liczba sektorow = 256 Najmniejsza srednia = 2.5 w kwadracie 158/256

Liczba sektorow = 1024 Najmniejsza srednia = 1 w kwadracie 304/1024

Najmniejsze średnie dla pola numer: 13

Liczba sektorow = 1 Najmniejsza srednia = 4.84375 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 3.5625 w kwadracie 1/4

Liczba sektorow = 16 Najmniejsza srednia = 1 w kwadracie 6/16

Najmniejsze średnie dla pola numer: 14

Liczba sektorow = 1 Najmniejsza srednia = 4.50977 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 4.38281 w kwadracie 2/4

Liczba sektorow = 16 Najmniejsza srednia = 4.30859 w kwadracie 4/16

Liczba sektorow = 64 Najmniejsza srednia = 3.79688 w kwadracie 15/64

Liczba sektorow = 256 Najmniejsza srednia = 2.625 w kwadracie 99/256

Liczba sektorow = 1024 Najmniejsza srednia = 0.25 w kwadracie 389/1024

Najmniejsze średnie dla pola numer: 15

Liczba sektorow = 1 Najmniejsza srednia = 4.6875 w kwadracie 1/1

Liczba sektorow = 4 Najmniejsza srednia = 3 w kwadracie 3/4

## Zadanie 3.

W programach zostały zadeklarowane następujące zmienne i struktury:

```
#include <iostream>
#include <fstream>
using namespace std;
string s, s1, s2, s3;
fstream plik1, plik2, plik3;
int i,j,k,m,n,d;
string zaw_pliku[100000];
int l_wierszy;
int liczba_powiatow;
string powiaty_ozn[400];
int liczba_dat, liczba_dat_pom;
string daty[1000], daty_pom[1000];
struct wypadki_woj
{
    string litera;
    int liczba;
};
string litera_woj;
int liczba_woj;
wypadki_woj liczby_wypadkow_woj[20];
struct wypadki_pow
{
    string ozn;
    int suma;
};
string ozn_pow;
int liczba_pow;
wypadki_pow sumy_wypadkow_pow[500], pom_sumy_pow;
struct wypadki_tyg_pow
{
    string ozn;
    int ilosci[6];
};
wypadki_tyg_pow ilosci_wypadkow_tyg_pow[500];
bool reprezentatywny;
//do generowania dat
time_t czas, czas2;
struct tm * data;
char data_tekst[ 80 ];
string data_string;
```

### Zadanie 3.1.

```
int main()
{
    plik1.open("wypadki.txt", ios::in);
    liczba_woj=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
        {
            getline(plik1, s);
            if (s!="")
                {litera_woj=s[0];

                    if (liczba_woj==0)
                        {liczba_woj=1;
```

```

        liczby_wypadkow_woj[1].liczba=1;
        liczby_wypadkow_woj[1].litera=litera_woj;
    }
    else
    {
        k=1;
        while ((k<=liczba_woj) &&
            (liczby_wypadkow_woj[k].litera!=litera_woj))
            {
                k++;
            }
        if (k>liczba_woj)
            {
                liczba_woj++;
                liczby_wypadkow_woj[liczba_woj].liczba=1;
                liczby_wypadkow_woj[liczba_woj].litera=litera_woj;
            }
        else
            {
                liczby_wypadkow_woj[k].liczba++;
            }
    }
}
plik1.close();

cout << endl << endl << "Liczby wypadkow: " << endl;
for (i=1; i<=liczba_woj; i++)
{
    cout << liczby_wypadkow_woj[i].litera << " "
        << liczby_wypadkow_woj[i].liczba << endl;
}
cout << endl << endl << "GOTOWE";
getchar();
return 0;
}

```

**Odpowiedź:**

K 284

## Zadanie 3.2.

```

int main()
{
    plik1.open("wypadki.txt", ios::in);
    liczba_pow=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
        {
            getline(plik1, s);
            if (s!="")
                {s1=s;
                 k=s1.find(" ");
                 ozn_pow=s1.substr(0,k);
                 s1=s1.substr(k+1);
                 k=s1.find(" ");
                 s1=s1.substr(k+1);
                 m=atoi(s1.c_str());
                 if (liczba_pow==0)
                     {liczba_pow=1;

```

```

        sumy_wypadkow_pow[1].suma=m;
        sumy_wypadkow_pow[1].ozn=ozn_pow;
    }
else
{
    k=1;
    while ((k<=liczba_pow) &&
           !(sumy_wypadkow_pow[k].ozn!=ozn_pow))
    {
        k++;
    }
    if (k>liczba_pow)
        {liczba_pow++;}
        sumy_wypadkow_pow[liczba_pow].suma=m;
        sumy_wypadkow_pow[liczba_pow].ozn=ozn_pow;
    }
    else
    {
        sumy_wypadkow_pow[k].suma=sumy_wypadkow_pow[k].suma+m;
    }
}

}
}
plik1.close();
//sortowanie

for (i=liczba_pow-1; i>=1; i--)
{
    for (j=1; j<=i; j++)
    {
        if ( (sumy_wypadkow_pow[j].ozn[0]>sumy_wypadkow_pow[j+1].ozn[0]) ||
            ( (sumy_wypadkow_pow[j].ozn[0]==sumy_wypadkow_pow[j+1].ozn[0]) &&
              (sumy_wypadkow_pow[j].suma<sumy_wypadkow_pow[j+1].suma) ) )
        {
            pom_sumy_pow=sumy_wypadkow_pow[j];
            sumy_wypadkow_pow[j]=sumy_wypadkow_pow[j+1];
            sumy_wypadkow_pow[j+1]=pom_sumy_pow;
        }
    }
}

cout << endl << endl << "Najwieksze sumy wypadkow: " << endl;
cout << sumy_wypadkow_pow[1].ozn << " " << sumy_wypadkow_pow[1].suma << endl;
k=sumy_wypadkow_pow[1].suma;
for (i=2; i<=liczba_pow; i++)
{
    if (sumy_wypadkow_pow[i].ozn[0]!=sumy_wypadkow_pow[i-1].ozn[0])
        {k=sumy_wypadkow_pow[i].suma;
         cout << sumy_wypadkow_pow[i].ozn << " " << sumy_wypadkow_pow[i].suma << endl;
        }
    else
    {
        if (sumy_wypadkow_pow[i].suma==k)
            {cout << sumy_wypadkow_pow[i].ozn << " " << sumy_wypadkow_pow[i].suma << endl;
            }
    }
}

cout << endl << endl << "GOTOWE";

getchar();
return 0;
}

```

**Odpowiedź:**

BSU 225

BS 225

CGR 249

DGL 270

ELC 235

ERW 235

ESI 235

FGW 282

GSL 281

KN 222

LUB 260

LSW 260

NGI 265

OOL 281

POB 291

RLE 259

SR 299

TOS 265

WPN 308

ZKL 314

**Zadanie 3.3.**

```
int main()
{
    plik1.open("wypadki.txt", ios::in);
    liczba_pow=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {
                getline(plik1, s);
                if (s!="")
                    {s1=s;
                     k=s1.find(" ");
                     ozn_pow=s1.substr(0,k);

                     s1=s1.substr(k+1);
                     k=s1.find(" ");
                     s3=s1.substr(0,k);

                     s2=s1.substr(5,2);
                     m=atoi(s2.c_str());
                     s2=s1.substr(8,2);
                     d=atoi(s2.c_str());

                     k=m-7;
                     if (d<=30)
                         {j=(d-1)/10;
```

```

        }
    else
        {j=2;
     }

    i=k*3+j;

    cout << ozn_pow << " " << m << "-" << d << " " << i << endl;

    if (liczba_pow==0)
        {liczba_pow=1;
         ilosci_wypadkow_tyg_pow[1].ilosci[i]=1;
         ilosci_wypadkow_tyg_pow[1].ozn=ozn_pow;
        }
    else
        {k=1;
         while ((k<=liczba_pow) &&
                (ilosci_wypadkow_tyg_pow[k].ozn!=ozn_pow))
            {k++;
             }
         if (k>liczba_pow)
             {liczba_pow++;
              ilosci_wypadkow_tyg_pow[liczba_pow].ilosci[i]=1;
              ilosci_wypadkow_tyg_pow[liczba_pow].ozn=ozn_pow;
             }
         else
             {ilosci_wypadkow_tyg_pow[k].ilosci[i]++;
              }
         }
      }
    }
plik1.close();

for (i=1; i<=liczba_pow; i++)
    {cout << ilosci_wypadkow_tyg_pow[i].ozn << " ";
for (j=0; j<=5; j++)
    {cout << ilosci_wypadkow_tyg_pow[i].ilosci[j] << " ";
     }
    cout << endl;
    }

cout << endl << endl << "Powiaty z danymi reprezentatywnymi: " << endl;
for (i=1; i<=liczba_pow; i++)
{
reprezentatywny=true;
k=0;
while ((k<=5) && (reprezentatywny==true))
    {if (ilosci_wypadkow_tyg_pow[i].ilosci[k]<7)
       {reprezentatywny=false;
        }
       k++;
    }

if (reprezentatywny==true)
    {cout << ilosci_wypadkow_tyg_pow[i].ozn << endl;
     }
}

```

```
    }
}
cout << endl << endl << "GOTOWE";

getchar();
return 0;
}
```

**Odpowiedź:**

POT  
NOE  
ZDR  
FGW  
SMI  
ZKL  
LSW  
LBL  
PSZ  
TOS  
POB  
WGS  
SG  
ZK  
NGI

## Zadanie 4.1.

A	B	C
1		=DŁ(A5)/2
2		
3		
4	symbol	temat
5	a1	Informatyka
6	a1a1	Programowanie
7	a1a2	Sieci komputerowe
8	a1a3	Grafika komputerowa
9	a1a1a1	Python
10	a1a1a2	C++
11	a1a2a1	Model ISO OSI
12	a1a4	Programowanie aplikacji
13	a1a4a1	Html i css
14	a1a4a2	JS
15	a1a4a3	PHP
16	a1a4a2a1	Node
17		
18		Zadanie 4.1 max = 4

**Odpowiedź:**

4

## Zadanie 4.2.

	A	B	C	D	E	F	G	H
1								
2								
3								
4	symbol	temat		poziom	1	2	3	4
5	a1	Informatyka		1 a1				
6	a1a1	Programowanie		2 a1	a1			
7	a1a2	Sieci komputerowe		2 a1	a2			
8	a1a3	Grafika komputerowa		2 a1	a3			
9	a1a1a1	Python		3 a1	a1	a1		
10	a1a1a2	C++		3 a1	a1	a2		
11	a1a2a1	Model ISO OSI		3 a1	a2	a1		
12	a1a4	Programowanie aplikacji		2 a1	a4			
13	a1a4a1	Html i css		3 a1	a4	a1		
14	a1a4a2	JS		3 a1	a4	a2		
15	a1a4a3	PHP		3 a1	a4	a3		
16	a1a4a2a1	Node		4 a1	a4	a2	a1	

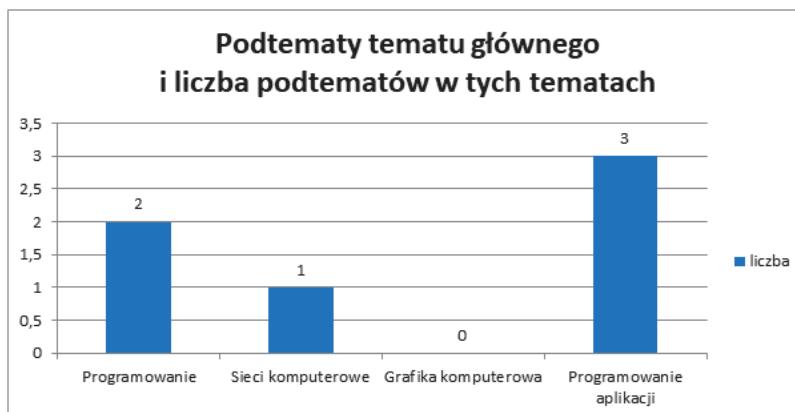
Następnie analiza tabelami przestawnymi:

**Odpowiedź:**

Pierwszy i ma 4 podtematy.

## Zadanie 4.3.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1								=JEŽELI(D1(A5)=4;B5;"")	=JEŽELI(ORAZ(D1(A5)=6;FRAGMENT.TEKSTU(A5;1;4)=\\$I\$4);1;0)					
2								↓ podtematy 2 poziomu	↓	a1a1	a1a2	a1a3	a1a4	
3										0	0	0	0	
4	symbol	temat	poziom	1	2	3	4							
5	a1	Informatyka	1 a1					Programowanie		0	0	0	0	
6	a1a1	Programowanie	2 a1	a1				Sieci komputerowe		0	0	0	0	
7	a1a2	Sieci komputerowe	2 a1	a2				Grafika komputerowa		0	0	0	0	
8	a1a3	Grafika komputerowa	2 a1	a3						1	0	0	0	
9	a1a1a1	Python	3 a1	a1	a1					1	0	0	0	
10	a1a1a2	C++	3 a1	a1	a2					0	1	0	0	
11	a1a2a1	Model ISO OSI	3 a1	a2	a1			Programowanie aplikacji		0	0	0	0	
12	a1a4	Programowanie aplikacji	2 a1	a4						0	0	0	0	
13	a1a4a1	Html i css	3 a1	a4	a1					0	0	0	1	
14	a1a4a2	JS	3 a1	a4	a2					0	0	0	1	
15	a1a4a3	PHP	3 a1	a4	a3					0	0	0	1	
16	a1a4a2a1	Node	4 a1	a4	a2	a1				0	0	0	0	
17		Zadanie 4.1 max =		4						2	1	0	3	
18														
19														
20														
21														
22														
23														



## Zadanie 4.4.

**Wskazówka:**

Policzyć dla każdego poziomu te tematy, które się kończą na danym poziomie, np. poprzez sprawdzenie, czy w następnym poziomie jest pusta komórka, używając funkcji `ile.niepu` `↳stych()`.

## Zadanie 5.1.

```
SELECT DISTINCT MARKA
FROM SPRZET;
```

## Zadanie 5.2.

```
SELECT ROK_PRODUKCJI, Count(ROK_PRODUKCJI) AS LICZBA
FROM SPRZET
WHERE ROK_PRODUKCJI=(SELECT Max(ROK_PRODUKCJI)
FROM SPRZET)
GROUP BY ROK_PRODUKCJI
```

## Zadanie 5.3.

```
SELECT NAZWA, sum((DATA_DO - DATA_OD+1)*KOSZT_NA_DZIEN ) AS SUMA
FROM SPRZET INNER JOIN WYPOZYCZENIA ON SPRZET.ID_SPRZETU = WYPOZYCZENIA.ID_SPRZETU
GROUP BY NAZWA
ORDER BY sum((DATA_DO - DATA_OD+1)*KOSZT_NA_DZIEN ) DESC
```

## Zadanie 5.4.

```
SELECT NAZWA , ID_WYPOZYCZENIA
FROM SPRZET LEFT JOIN WYPOZYCZENIA ON SPRZET.ID_SPRZETU = WYPOZYCZENIA.ID_SPRZETU
WHERE ID_WYPOZYCZENIA is null;
```

*lub*

```
SELECT NAZWA
FROM SPRZET
WHERE ID_SPRZETU NOT IN (SELECT ID_SPRZETU
                           FROM WYPOZYCZENIA)
```

## Zadanie 5.5.

```
SELECT NAZWA, PESEL
FROM SPRZET INNER JOIN WYPOZYCZENIA ON SPRZET.ID_SPRZETU = WYPOZYCZENIA.ID_SPRZETU
WHERE ((DATA_OD) Between #11/1/2022# And #11/30/2022#) AND
  ((DATA_DO)<=#11/30/2022#));
```

## Zadanie 5.6.

```
SELECT TOP 1 DATA_OD, Count(DATA_OD)
FROM WYPOZYCZENIA
GROUP BY DATA_OD
ORDER BY Count(DATA_OD) DESC
```

## Zadanie 5.7.

```
SELECT NAZWA
FROM SPRZET
WHERE MOC > (SELECT AVG(MOC)
               FROM SPRZET);
```

## Zadanie 5.8.

```
SELECT ID_SPRZETU
FROM WYPOZYCZENIA
WHERE KAUCJA >=100
GROUP BY ID_SPRZETU
HAVING Count(ID_SPRZETU) >=2
```

## Zadanie 5.9.

```
SELECT Count(NAZWA)
FROM SPRZET
WHERE NAZWA Like "*wiertarka*"
```

**Zadanie 5.10.**

```
SELECT PESEL, Count(PESEL)
FROM SPRZET, WYPOZYCZENIA
WHERE (SPRZET.ID_SPRZETU=WYPOZYCZENIA.ID_SPRZETU) AND (SPALINOWY=True)
GROUP BY WYPOZYCZENIA.PESEL
ORDER BY Count(PESEL) DESC
```

**Zadanie 5.11.**

```
SELECT ID_SPRZETU, KAUCJA
FROM WYPOZYCZENIA
GROUP BY ID_SPRZETU, KAUCJA
```

**Zadanie 5.12.**

```
SELECT MARKA, Count(ID_SPRZETU)
FROM SPRZET
WHERE MOC >=1000 AND SPALINOWY = true
GROUP BY MARKA
```

**Zadanie 5.13.**

```
SELECT DATA_OD
FROM WYPOZYCZENIA
WHERE PESEL LIKE "?????????0?" OR PESEL LIKE "?????????2?" OR PESEL LIKE
"?????????4?" OR PESEL LIKE "?????????6?" OR PESEL LIKE "?????????8?"
GROUP BY DATA_OD
```

**Zadanie 5.14.**

```
SELECT DISTINCT ID_SPRZETU
FROM WYPOZYCZENIA
WHERE KOSZT_NA_DZIEN > KAUCJA
```

**Zadanie 5.15.**

```
SELECT NAZWA, MARKA
FROM SPRZET INNER JOIN WYPOZYCZENIA ON SPRZET.ID_SPRZETU = WYPOZYCZENIA.ID_SPRZETU
ORDER BY DATA_OD DESC
```

**Zadanie 5.16.**

```
SELECT Count(ID_WYPOZYCZENIA)
FROM WYPOZYCZENIA
WHERE DATA_OD = DATA_DO
```

**Zadanie 5.17.**

```
SELECT MARKA
FROM SPRZET
WHERE SPALINOWY = true AND MOC = (SELECT Max(MOC)
                                    FROM SPRZET)
```

# Zestaw 5

## Zadanie 1. Wyścig kolarski

Grupa miłośników kolarstwa, chcąc popularyzować tę dyscyplinę sportu oraz promować swoje województwo, wpadła na pomysł organizacji wyścigu dookoła regionu, w którym mieszkają. W tym celu zorganizowali konkurs na propozycje tras poszczególnych etapów. Po zakończeniu terminu przesyłania propozycji wybrano pięćdziesiąt najciekawszych. Oprócz walorów krajoznawczych postanowiono dokonać analizy tras, uwzględniając położenie wysokościowe (w metrach nad poziomem morza) poszczególnych fragmentów propozowanych tras.

W pliku *wysokosci\_etapow.txt* znajduje się pięćdziesiąt wierszy. Każdy wiersz zawiera informacje o położeniu wysokościowym odcinków jednego etapu: numer propozycji oddzielony dwukropkiem od wysokości oraz wysokości poszczególnych punktów oddzielone spacją. Pierwsza wysokość to wysokość, na której znajduje się miejsce startu, a ostatnia, na której znajduje się miejsce finiszu.

### Przykładowy fragment pliku *wysokosci\_etapow.txt*

```
37:433 558 316 160 550 653 473 400 388 422 563  
38:181 417 515 480 510 389 685 638 575 675 507 205 588  
39:486 304 482 631 187 654 624 669 539 190 228  
40:455 216 650 454 390 203 665 568 541 282 634 424 438 576 349  
41:499 337 514 217 567 567 544 562 259 264  
42:523 455 203 497 349 613 271 271 545 225 399 326 394 425  
43:377 425 256 222 416 381 619 457 197 392 239 601
```

Na podstawie danych zawartych w pliku *wysokosci\_etapow.txt* napisz program (lub programy), który da odpowiedź lub wykona polecenia zawarte w zadaniach 1.1 – 1.4.

### Zadanie 1.1.

Odcinek trasy nazwiemy **podjazdem**, jeśli początek tego odcinka jest położony niżej niż jego koniec. Analogicznie odcinek trasy nazwiemy **zjazdem**, jeśli początek jest położony wyżej niż koniec.

**Przykład:**

Miejsce startu położone jest na wysokości 378 m n.p.m., natomiast finisz odbędzie się na wysokości 113 m n.p.m. Na tym etapie wyróżniono siedem odcinków:

378	187	442	242	161	161	113							
378	187	↗	442	↘	242	↗	642	↘	161	→	161	↘	113
1	2		3		4		5		6		7		

W tym przykładzie mamy dwa podjazdy (odcinki 2, 4), cztery zjazdy (1, 3, 5, 7) i jeden odcinek płaski (6).

**Współczynnikiem zjazdu** nazwiemy liczbę  $\frac{z}{p+z} \cdot 100$ , gdzie  $p$  to liczba podjazdów, natomiast  $z$  liczba zjazdów.

Napisz program, który wypisze numery propozycji tras ze współczynnikiem zjazdu większym niż 60.

**Zadanie 1.2.**

Trasę uznajemy za **wysokościowo normalną**, jeśli wysokości 70% punktów mieszczą się w przedziale  $(\bar{x} - \sigma; \bar{x} + \sigma)$ , gdzie  $\bar{x}$  jest średnią arytmetyczną wysokości wszystkich punktów na trasie, a  $\sigma$  odchyleniem standardowym zbioru tych punktów.

**Przykład:**

Sprawdzamy, czy trasa z następującymi wysokościami jest wysokościowo normalna. Jednocześnie przypomnijmy niezbędne wzory.

674, 415, 428, 536, 369, 164, 227, 196, 325, 636, 391, 349

Niech  $x_1, x_2, x_3, \dots, x_n$  będzie zbiorem wysokości. Na tej trasie ustalono 12 punktów. Zatem  $n = 12$ ,  $x_1 = 674$ ,  $x_2 = 415$ ,  $x_3 = 428, \dots, x_{11} = 391$ ,  $x_{12} = 349$ .

Obliczamy średnią arytmetyczną tego zestawu danych:

$$\bar{x} = \frac{x_1 + x_2 + x_3 + \dots + x_n}{n} = \frac{674 + 415 + 428 + \dots + 349}{12} = \frac{4710}{12} = 392,5$$

Obliczamy odchylenie standardowe:

$$\begin{aligned} \sigma &= \sqrt{\frac{(x_1 - \bar{x})^2 + (x_2 - \bar{x})^2 + (x_3 - \bar{x})^2 + \dots + (x_n - \bar{x})^2}{n}} \\ &= \sqrt{\frac{(674 - 392,5)^2 + (415 - 392,5)^2 + (428 - 392,5)^2 + \dots + (349 - 392,5)^2}{12}} \approx 154,41 \end{aligned}$$

Sprawdzamy, ile wysokości punktów należy do przedziału  $(\bar{x} - \sigma; \bar{x} + \sigma)$ . W naszym przykładzie będzie to w przybliżeniu przedział  $(392,5 - 154,41; 392,5 + 154,41)$ , czyli  $(238,09; 546,91)$ . Do tego przedziału należy 7 spośród 12 wysokości, co stanowi w przybliżeniu 58%. Zatem trasa ta nie jest wysokościowo normalna.

Napisz program, który wypisze numery tras wysokościowo normalnych wraz z procentem punktów należących do przedziału  $(\bar{x} - \sigma; \bar{x} + \sigma)$ .

**Zadanie 1.3.**

**Częścią podjazdową trasy** będziemy nazywać zbiór odcinków podjazdowych następujących zaraz po sobie. Liczba tych odcinków będzie długością części podjazdowej. Kolejnym elementem, pod kątem którego trasy są badane, jest najdłuższa część podjazdowa.

**Przykład:**

Rozpatrzmy trasę:

278 ↘ 192 ↗ 220 ↗ 248 ↘ 149 ↗ 161 → 161 ↘ 113 ↗ 189 ↗ 301 ↘ 275

Na tej trasie mamy trzy części podjazdowe, które zaznaczono szarym kolorem.

Pierwsza o długości 2 rozpoczyna się w drugim punkcie, następna o długości 1 ma początek w piątym punkcie, ostatnia rozpoczynająca się w ósmym punkcie ma długość 2.

Napisz program, który dla każdej trasy z pliku *wysokości\_etapow.txt* wyznaczy największą długość występujących w niej części podjazdowych oraz poda punkt lub punkty początkowe części podjazdowych o największej długości.

**Zadanie 1.4.**

Trasę etapu nazwiemy **palindromiczną**, jeżeli kolejność występowania podjazdów, zjazdów i odcinków płaskich ze startu do mety jest taka sama, jak w kierunku przeciwnym.

**Przykłady tras palindromicznych:**

278 ↘ 192 ↗ 220 ↗ 248 → 248 ↗ 481 ↘ 413 → 413 ↘ 389 ↘ 301 ↗ 375

Kolejność: *zjazd-podjazd-podjazd-płaski-podjazd-zjazd-płaski-zjazd-zjazd-podjazd*

378 ↘ 242 → 242 ↗ 448 ↗ 508 → 508 ↘ 413 ↘ 389 → 389 ↗ 401

Kolejność: *zjazd-płaski-podjazd-podjazd-płaski-zjazd-zjazd-płaski-podjazd*

Napisz program, który wypisze numery tras palindromicznych.

**Zadanie 2. Elf złodziejaszek**

Nie od dziś wiadomo, że Świętemu Mikołajowi przy rozdawaniu prezentów pomagają elfy. W zeszłym roku jeden z nich okazał się nieuczciwy. Święty Mikołaj skierował niedobrego elfa na trzy osiedla domków w pewnym miasteczku. Domy tworzą osiedla w kształcie kwadratów. Adres każdego domu składa się z dwóch liczb oddzielonych myślnikiem. Pierwsza z tych liczb to numer wiersza, a druga kolumny.

**Przykład:**

	1	2	3	4
1				
2	2-1	2-2	2-3	2-4
3				
4	3-1	3-2	3-3	3-4
4	4-1	4-2	4-3	4-4

*Na powyższym przykładowym planie osiedla znajduje się 16 domów. To osiedle ma rozmiar równy 4. Adresy poszczególnych domów zamieszczono pod graficznym symbolem.*

Zadaniem elfa jest odwiedzić wszystkie domy na osiedlu i wręczyć dzieciom prezenty. Jeżeli w danym domu nie mieszkały dzieci, to elf odwiedza taki dom i tylko składa życzenia domownikom. Informacje o liczbie dzieci mieszkających w danym domu zawarte są w dwuwymiarowej tablicy  $d$ . Liczba dzieci mieszkających w domu o adresie  $x-y$  zapisana jest w elemencie  $d[x][y]$ .

Prezenty znajdują się w workach. W każdym worku znajduje się  $p$  prezentów. Dla każdego dziecka przewidziany jest jeden prezent.

Elf złodziejaszek wymyślił, że część prezentów zatrzyma dla siebie. Postanowił, że do niektórych domów, w których mieszkały dzieci, nie będzie wchodził — wówczas zwiększa się liczba dzieci, które nie otrzymały prezentów. Algorytm postępowania niegodziwego elfa znajduje się poniżej. Przeanalizuj go. Zwróć uwagę na trasę, jaką pokonuje, aby przejść całe osiedle, a także na regułę decydującą o niewchodzeniu do danego domu.

### Specyfikacja:

Dane:

$n$  — rozmiar osiedla

$p$  — liczba prezentów w jednym worku

$d[1..n][1..n]$  — tablica z liczbami dzieci w poszczególnych domach

Wynik:

$w$  — liczba dzieci, które nie otrzymały prezentów

$k$  — liczba domów, do których nie wszedł elf

```
procedura sprawdz(x, y)
    jeśli a+d[j][i]<p
        a ← a + d[j][i]
    w przeciwnym razie
        jeśli a+d[j][i]>p
            w ← w + d[j][i]
            k ← k + 1
        a ← 0
```

### Algorytm:

```
a ← 0
w ← 0
k ← 0
i ← 1
dopóki i<=n wykonuj
    dla każdego j=1,2,3,...,n wykonuj
        sprawdz(j, i)
        i ← i+1
    jeżeli i<=n
        dla każdego j=n,n-1,n-2,...,1 wykonuj
            sprawdz(j, i)
    i ← i+1

procedura sprawdz(x, y)
    jeśli a+d[j][i]<p
        a ← a + d[j][i]
```

```
w przeciwnym razie
jeśli a+d[j][i]>p
    w ← w + d[j][i]
    k ← k + 1
a ← 0
```

Algorytm:

```
a ← 0
w ← 0
k ← 0
i ← 1
dopóki i<=n wykonuj
    dla każdego j=1,2,3,...,n wykonuj
        sprawdz(j, i)
    i ← i+1
jeżeli i<=n
    dla każdego j=n,n-1,n-2,...,1 wykonuj
        sprawdz(j, i)
    i ← i+1
```

**Zadanie 2.1.**

Działając zgodnie z algorytmem obmyślonym przez przebiegłego elfa, uzupełnij tabelę:

<b>n</b>	<b>p</b>	<b>tablica <math>d[1..n][1..n]</math></b>	<b>k</b>	<b>w</b>
5	15	1 0 4 4 3 0 3 0 2 2 1 4 3 0 3 1 4 3 4 1 3 2 3 2 0	2	7
6	10	2 2 0 3 4 3 4 0 0 3 2 4 4 3 4 1 4 3 3 2 0 3 0 0 3 4 3 4 1 0 4 2 4 0 0 3		
7	20	0 0 2 0 0 0 1 2 3 1 0 2 4 0 2 4 1 0 0 3 4 3 1 2 3 0 4 3 4 3 4 0 2 4 4 1 0 2 4 0 3 4 2 2 1 4 1 1 0		

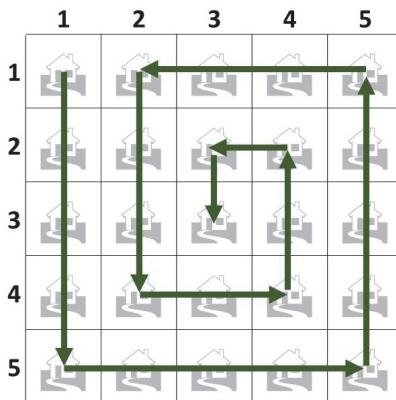
## Zadanie 2.2.

Kiedy niecny plan elfa nie wypali? Kiedy, pomimo zastosowania algorytmu, wszystkie dzieci mieszkające na osiedlu otrzymają prezenty?

## Odpowiedź:

## Zadanie 2.3.

Okazało się, że niecne działania elfa nie wyszły na jaw. W tym roku elf także otrzymał zadanie doręczenia prezentów dzieciom. Postanowił ponownie zastosować swój algorytm. Zmodyfikował jedynie trasę, jaką będzie pokonywał, aby przejść całe osiedle. Reguła decydująca o tym, do których domów nie wchodzi, się nie zmieniła. W tym roku będzie obchodził osiedle metodą ślimaka.



Napisz w pseudokodzie lub wybranym języku programowania algorytm realizujący tego-roczone założenia elfa.

## Specyfikacja:

Dane:

*n* — rozmiar osiedla

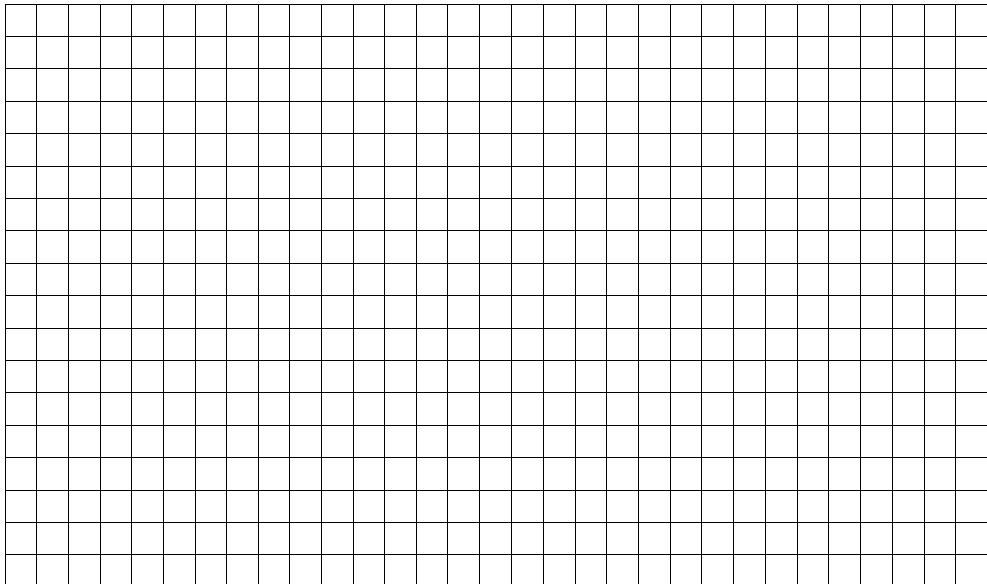
$p$  — liczba prezentów w jednym worku

$d[1..n][1..n]$  — tablica z liczbami dzieci w poszczególnych domach

## Wynik:

w — liczba dzieci, które nie otrzymały prezentów

$k$  — liczba domów, do których nie wszedł elf



## Zadanie 2.4.

Napisz w pseudokodzie lub wybranym języku programowania algorytm realizujący tego-roczone założenia elfa dla osiedli w kształcie prostokąta o wymiarach  $n \times m$ .

Przykład osiedla o wymiarach  $3 \times 6$  ( $n = 3$ ,  $m = 6$ ):

	1	2	3	4	5	6
1						
2	1-1	1-2	1-3	1-4	1-5	1-6
2						
2	2-1	2-2	2-3	2-4	2-5	2-6
3						
3	3-1	3-2	3-3	3-4	3-5	3-6

## Specyfikacja:

Dane:

$n$  – rozmiar osiedla „w pionie”

$m$  – rozmiar osiedla „w poziomie”

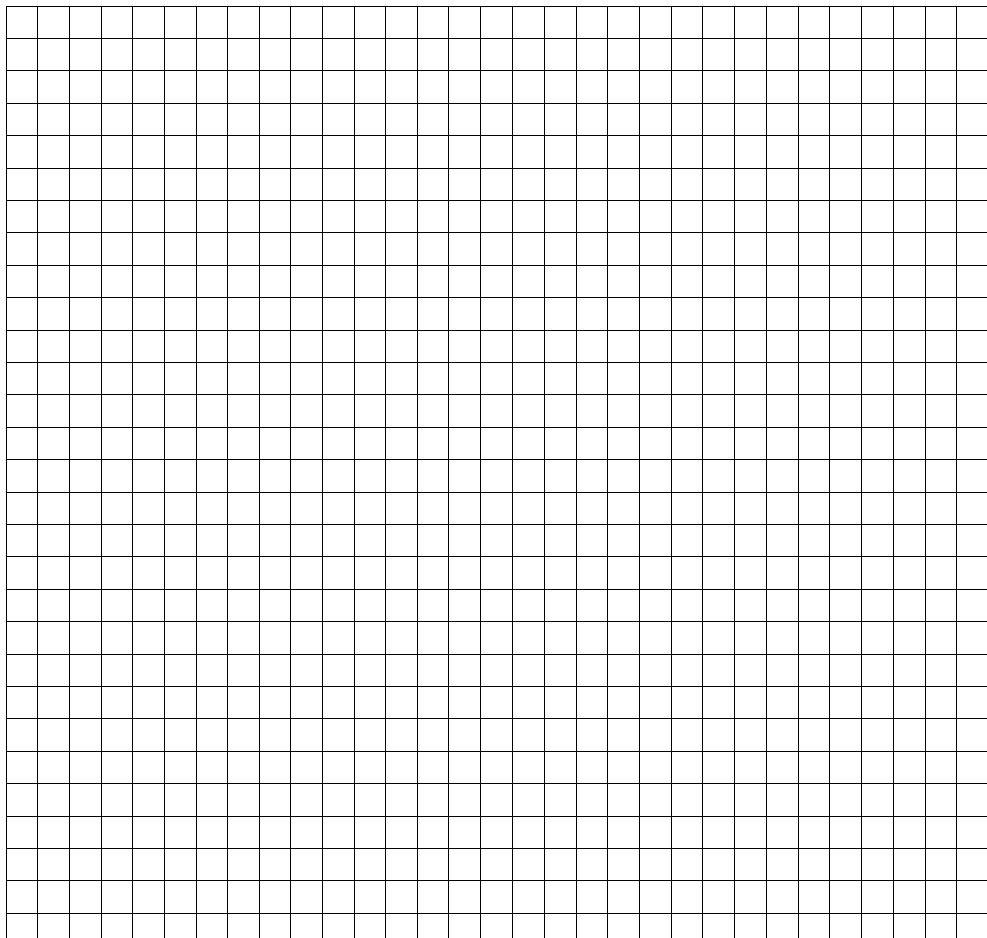
$p$  – liczba prezentów w jednym worku

$d[1..n][1..m]$  — tablica z liczbą dzieci w poszczególnych domach

### Wynik:

$w$  – liczba dzieci, które nietrzymały prezentów

$k$  — liczba domów, do których nie wszedł elf



## Zadanie 3. Grant naukowy

Matematyk z Japonii, pan Noriaki Takahashi, otrzymał grant naukowy, w ramach którego przez jeden rok akademicki będzie prowadził badania oraz zajęcia w pewnym bardzo dużym mieście w Europie. Specjalnością pana Takahashiego jest teoria zbiorów rozmytych.

Miasto, do którego pojedzie nasz naukowiec, cechuje bardzo duża liczba hoteli, których nazwy są imionami greckich bogów. Położenie hoteli określone jest przez dwie liczby całkowite, współrzędne, analogicznie do punktów w kartezjańskim układzie współrzędnych. Punktym opisującym położenie hotelu jest jego wejście główne. Jednostką określającą położenie jest metr. W punkcie  $(0,0)$  znajduje się główne wejście do ratusza miejskiego, siedziby władz miasta. Standard danego hotelu określa przypisana mu liczba gwiazdek — od jednej do pięciu.

W pliku *hotele.txt* znajdują się informacje o dostępnych hotelach. Każdy wiersz w pliku zawiera: nazwę hotelu, liczbę przydzielonych gwiazdek, współrzędne hotelu. Dane oddzielone są średnikami.

### Przykładowy fragment pliku hotele.txt

Tyche;4;-3670;-3734  
 Hermes;1;-2194;-5393  
 Hestia;0;5579;-657  
 Artemida;5;6014;-3305  
 Zefir;2;-2924;-6503  
 Eris;2;-3089;2959  
 Nemezis;2;6424;-1971  
 Posejdon;0;-1559;-3665

Instytut naukowy, w którym pan Noriaki będzie realizował grant, ma współrzędne (500,500). Nasz matematyk w pierwszej kolejności musi wybrać hotel, w którym będzie mieszkał. Pieniądze w ramach grantu są ograniczone. Wiadomo, że im wyższy standard hotelu, tym cena noclegu jest wyższa. Im hotel dalej położony od instytutu, tym wyższe koszty codziennych dojazdów. Jako matematyk pan Takahashi postanowił podejść do problemu matematycznie. Zdefiniował funkcję korzystności oferty  $k$  w zależności od standardu hotelu  $s$  (liczba przydzielonych gwiazdek) i odległości od instytutu  $d$  w następujący sposób:

$$k(s, d) = 0,07s - 0,00013d + 0,65$$

Odległość pomiędzy dwoma punktami  $A(x_A, y_A)$  i  $B(x_B, y_B)$  obliczamy ze wzoru:

$$d = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$$

Zbiorem wartości tej funkcji  $k$  jest przedział  $(-\infty; 1)$ . Następnie określił on stopnie oceny oferty danego hotelu w oparciu o wartości funkcji korzystności. Im wartość funkcji jest większa, tym oferta danego hotelu jest dla naszego matematyka korzystniejsza.

Ocena oferty	Wartości funkcji $k$
Bardzo korzystna	$(0,8; 1)$
Średnio korzystna	$(0,5; 0,8)$
Mało korzystna	$(0,3; 0,5)$
Bardzo mało korzystna	$(0; 0,3)$
Niekorzystna	$(-0,5; 0)$
Bardzo niekorzystna	$(-\infty; -0,5)$

#### Przykład:

Obliczymy wartość funkcji korzystności dla następującego hotelu:

Eris;2;-3089;2959

Standard hotelu  $s = 2$ . Obliczmy odległość punktu  $(-3089, 2959)$  od punktu  $(500, 500)$ :

$$d = \sqrt{(500 - (-3089))^2 + (500 - 2959)^2} = 4350,5863 \dots$$

Wyznaczamy wartość funkcji  $k$ . Dla przejrzystości w przypadku odległości zapiszemy tylko cztery cyfry po przecinku:

$$k(s, d) = 0,07s - 0,00013d + 0,65 = 0,07 * 2 - 0,00013 * 4350,5863 \dots + 0,65 \\ \approx 0,22$$

Dla naszego matematyka oferta tego hotelu jest bardzo mało korzystna.

Napisz program lub programy, które udzielą odpowiedzi lub wykonają polecenia zamieszczone w zadaniach 3.1 – 3.4.

## Zadanie 3.1.

Dla każdego standardu hoteli (liczby gwiazdek) określ najmniejszą odległość między dwoma hotelami w tym standardzie. W zestawieniu podaj nazwy hoteli oraz odległość pomiędzy nimi.

## Zadanie 3.2.

Jaki procent wszystkich hoteli w mieście zdaniem pana Takahashiego ma dla niego ofertę niekorzystną?

## Zadanie 3.3.

Pan Takahashi stwierdził, że wybierze hotel odległy od instytutu o nie więcej niż trzy kilometry i którego oferta jest co najmniej średnio korzystna. Jaki hotel wybrał naukowiec?

## Zadanie 3.4.

Instytut, w którym realizuje grant pan Noriaki, planuje wybudowanie drugiej siedziby. Ustalono, że ma ona być oddalona od obecnej o nie mniej niż dwa kilometry oraz o nie więcej niż pięć kilometrów. Dla ułatwienia komunikacji pomiędzy budynkami ustalono, że nowy budynek ma być położony w poziomie na tej samej linii co pierwsza siedziba, tzn. mają mieć tę samą drugą współrzędną położenia  $y$ .

Podaj wszystkie możliwe wartości pierwszej współrzędnej położenia  $x$  (liczba całkowita) nowego budynku, dla których liczba hoteli z korzystną ofertą jest największa. Podaj także tę liczbę hoteli.

## Zadanie 4. Firmowa poczta

Firma Informapol oferuje swoje usługi w następujących dziedzinach informatyki: sieci komputerowe, systemy operacyjne, sprzęt komputerowy, oprogramowanie, grafika komputerowa, gry i usługi internetowe. Dynamiczny rozwój technologii informatycznych w ostatnich latach oraz właściwa polityka prowadzenia firmy przez jej właścicieli spowodowała, że działa ona na rynkach wielu krajów europejskich. Firma zatrudnia tylko obywateli polskich.

W celu podniesienia bezpieczeństwa i przejrzystości wewnętrzna komisja postanowiła sprawdzić, w jakim stopniu przestrzegane są ustalenia dotyczące poczty elektronicznej. Przyjrzano się tworzeniu nazw kont pocztowych i sile haseł.

W pliku *pracownicy.txt* znajdują się informacje dotyczące kont pocztowych wszystkich pracowników. Każdy wiersz w pliku zawiera informacje o koncie pocztowym jednego pracownika: imię i nazwisko, adres e-mail oraz hasło. Dane oddzielone są średnikami.

**Przykładowy fragment pliku *pracownicy.txt***

CZESLAW;ZALEWSKI;czezal@promocje.internet.informapol.mk;  
#i0U&5uO#%YI8??\$yO^8E#3AAu  
EMILIA;KAZMIERCZAK;e.kazmierczak@reklamacje.gry.informapol.dk;  
4ua28#5I126\$8%a4i03YE?@#  
HANNA;URBANSKA;hurbanska@reklamacje.grafika.informapol.no;u1%EUi74?  
MONIKA;OSTROWSKA;monost@sprzedaz.gry.informapol.fi;%2147ouu\*  
PIOTR;KALINOWSKI;piotr.k@promocje.internet.informapol.ba;  
8?aA^^a3175oI0^o?@^7ai#  
IGNACY;MARCINIAK;ignacy.marciniak@reklamacje.oprogramowanie.  
informapol.ee;@a7y\*I&y12\$  
HELENA;KACZMARCZYK;h.kaczmarczyk@kadry.gry.informapol.es;  
y0Y#0&3!IAU5&auAi4E  
IZABELA;SZYMANSKA;izaszy@szkolenia.sprzet.informapol.al;\$0!U000#IO  
KACPER;KAMINSKI;kacperk@kadry.internet.informapol.sk;YYi4e%I?AY%4E?&5&

W adresie e-mail, zaraz po symbolu @, znajdują się oddzielone kropkami: nazwa działu (sprzedaż, promocje, szkolenia, kadry, finanse, reklamacje), w którym dana osoba pracuje, dziedzina informatyki oraz nazwa firmy. Na końcu znajduje się dwuliterowy kod kraju, w którym dany pracownik realizuje swoje zadania.

**Korzystając z arkusza kalkulacyjnego**, odpowiedz na pytania lub wykonaj polecenia zawarte w zadaniach 4.1 – 4.5.

## Zadanie 4.1.

W którym kraju firma ma najczęściej pracowników? Ilu ich jest?

## Zadanie 4.2.

Utwórz zestawienie liczby pracowników pracujących w poszczególnych działach firmy (bez podziału na kraje). Na podstawie tego zestawienia utwórz wykres kolumnowy. Pamiętaj o czytelności wykresu.

## Zadanie 4.3.

Zgodnie z zasadami przyjętymi w firmie nazwa konta pocztowego, występująca w adresie e-mail przed symbolem @, może mieć jedną z trzech postaci:

*imie.nazwisko@...*  
*pierwsza\_litera\_imienia.nazwisko@...*  
*imie.pierwsza\_litera\_nazwiska@...*

### Na przykład:

*Pracownik Eugeniusz Mazur może mieć nazwę konta w jednej z trzech postaci:*

*eugeniusz.mazur@...*  
*e.mazur@...*  
*eugeniusz.m@...*

Ille nazw kont zostało utworzonych zgodnie z przyjętymi zasadami?

## Zadanie 4.4.

Tworząc hasło, użytkownik ma do wyboru następujące cztery zbiory znaków:

- ◆ sześć wielkich samogłosek: *A, E, I, O, U, Y*,
- ◆ sześć małych samogłosek: *a, e, i, o, u, y*,
- ◆ dziesięć cyfr: *0, 1, 2, 3, 4, 5, 6, 7, 8, 9*,
- ◆ dziewięć znaków specjalnych: *!, @, #, %, ^, &, \*, ?*.

Zdefiniowano także wymagania dotyczące hasła. Stwierdzono, że hasło jest silne, jeżeli spełnia dwa warunki:

- ◆ składa się z co najmniej ośmiu znaków,
- ◆ zawiera co najmniej dwa różne znaki z każdego z czterech zbiorów znaków, tzn. co najmniej dwie różne wielkie samogłoski, co najmniej dwie różne małe samogłoski, co najmniej dwie różne cyfry i co najmniej dwa różne znaki specjalne.

Ile haseł zawartych w pliku *pracownicy.txt* **nie jest** hasłami silnymi?

## Zadanie 4.5.

Władze firmy stwierdziły, że wszystkie hasła powinny spełniać wymagania hasła silnego. W związku z powyższym postanowiono, że do każdego hasła dołożony zostanie z prawej strony pewien ciąg znaków, który spowoduje, że wszystkie hasła będą silne. Ciąg znaków będzie tworzony poprzez dodawanie, po kolej, po jednym znaku z każdego zbioru znaków:

*A*  
*Aa*  
*Aa0*  
*Aa0!*  
*Aa0!E*  
*Aa0!Ee*  
*Aa0!Ee1*  
*Aa0!Ee1@*  
*Aa0!Ee1@I*  
*Aa0!Ee1@li*  
*Aa0!Ee1@li2*  
*Aa0!Ee1@li2#*  
...

Z tak tworzonych ciągów znaków podaj najkrótszy, który dodany do wszystkich haseł powoduje, że wszystkie hasła są silne.

## Zadanie 5. Kwartal w łowiectwie

Portal internetowy zajmujący się łowiectwem postanowił zamieścić kwartalne podsumowanie dokonań kół łowieckich w kraju. Pod lupa wziął ostatni kwartał 2020 roku. Zebrane dane zostały zapisane w trzech plikach.

W pliku *wojewodztwa.txt* znajdują się nazwy województw oraz powierzchnia łowna w kilometrach kwadratowych dostępna dla myśliwych na terenie danego województwa. Każdy wiersz zawiera identyfikator województwa, jego nazwę oraz wielkość powierzchni łownej. Dane oddzielone zostały średnikami.

#### **Przykładowy fragment pliku *wojewodztwa.txt***

```
w10;slaskie;3061  
w05;opolskie;2329  
w09;lubuskie;6681  
w12;pomorskie;5730  
w01;wielkopolskie;6621  
w07;warmińsko-mazurskie;6932  
w06;świętokrzyskie;2251
```

W pliku *zwierzyna.txt* znajdują się nazwy gatunków zwierząt, na które polowali myśliwi. Każdy wiersz zawiera identyfikator gatunku zwierzyny, jego nazwę oraz informację, do której grupy zwierząt łownych się zalicza: zwierzyna gruba czy zwierzyna drobna. Dane oddzielone zostały średnikami.

#### **Przykładowy fragment pliku *zwierzyna.txt***

```
z07;kuna lesna;drobna  
z08;dziki krolik;drobna  
z09;sarna;gruba  
z10;jenot;drobna  
z11;dzik;gruba  
z12;jelen szlachetny;gruba  
z13;kuropatwa;drobna
```

Informacje o ilości upolowanej zwierzyny znajdują się w pliku *polowania.txt*. Każdy wiersz zawiera identyfikator wpisu, datę organizowanych w danym dniu polowań, identyfikator województwa, w którym miały one miejsce, identyfikator upolowanej zwierzyny oraz jej ilość upolowaną na wszystkich polowaniach w województwie w danym dniu. Polowania w każdym województwie odbywają się według harmonogramu ustalonego przez władze województwa.

#### **Przykładowy fragment pliku *polowania.txt***

```
0055;2020-10-10;w13;z12;22  
0201;2020-11-05;w05;z12;29  
0460;2020-12-24;w04;z12;28  
0416;2020-12-18;w16;z13;3  
0222;2020-11-08;w03;z08;2  
0317;2020-11-27;w13;z04;3  
0102;2020-10-18;w12;z12;25  
0108;2020-10-20;w05;z09;36  
0009;2020-10-05;w10;z17;5  
0327;2020-11-29;w08;z18;38
```

Wykorzystując dostępne narzędzia informatyczne, odpowiedz na pytania lub wykonaj polecenia zamieszczone w zadaniach 5.1 – 5.10. W przypadku gdy pod treścią zadania znajduje się miejsce na zapisanie treści zapytania w języku SQL, postaraj się w pierwszej kolejności zapisać to zapytanie, a w drugiej sprawdzić jego poprawność w wybranym narzędziu.

## Zadanie 5.1.

Na ile gatunków grubej, a na ile drobnej zwierzyny mogą polować myśliwi?

## Zadanie 5.2.

Wypisz daty polowań, które miały miejsce pomiędzy 15 października 2020 a 15 listopada 2020 i podczas których upolowano więcej niż 21 gesi jednego z gatunków.

### Zadanie 5.3.

Utwórz zestawienie, w którym dla każdego województwa podana jest łączna ilość upolowanej zwierzyny grubej. Zestawienie posortuj malejąco według ilości zwierzyny.

## Zadanie 5.4.

Ile średnio saren upolowano podczas jednego dnia polowań w województwie?

## Zadanie 5.5.

W których województwach nie upolowano żadnego muflona?

## Zadanie 5.6.

Ille lisów upolowano w całym kraju w okresie świątecznym, tj. w dniach 24 – 26 grudnia 2020 roku?

## Zadanie 5.7.

W którym dniu w jednym województwie upolowano najwięcej różnych gatunków zwierząt? Ile to było gatunków?

## Zadanie 5.8.

Utwórz zestawienie zawierające liczbę kilometrów kwadratowych powierzchni łownej przypadającą na jednego upolowanego dzika w danym województwie, zaokrągloną do czterech miejsc po przecinku. Zestawienie posortuj rosnąco według tej liczby.

## Zadanie 5.9.

W których województwach polowania nie miały miejsca w każdy dzień tygodnia, tzn. był taki dzień tygodnia, w którym nie odbyły się żadne polowania?

## Zadanie 5.10.

W których województwach liczba dni polowań w październiku jest równa liczbie dni polowań w listopadzie?

# Odpowiedzi i wskazówki do zestawu 5

## Zadanie 1.

W programach zostały zadeklarowane następujące zmienne i struktury:

```
#include <iostream>
#include <windows.h>
#include <fstream>
#include <cmath>
using namespace std;
string s, s2, s3, s4;
string zaw_pliku[1000];
int i, j, k, a, dl_ciagu;
int wiersz, numer_ciagu, ile_pkt;
int podjazdy, zjazdy;
int dl_czesci, dl_czesci_najdl, ile_czesci_najdl;
int pocz_czesci_najdl[100];
bool mon_palindrom;
double wsp_jazdy, srednia, odch_stand, suma, suma2, procent;
fstream plik1, plik2, plik3;
int ciag[1000];
```

### Zadanie 1.1.

```
int main()
{
    plik3.open("wyniki.txt", ios::out);
    plik1.open("wysokosci_etapow.txt", ios::in);
    wiersz=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                {
                    wiersz++;
                    s2=s;

                    k=s.find(':');
                    s3=s.substr(0,k);
                    numer_ciagu=atoi(s3.c_str());
                    s=s.substr(k+1);

                    dl_ciagu=0;
                    while (s!="")
                        {k=s.find(' ');
                         if (k>0)
                            {s3=s.substr(0,k);
                             a=atoi(s3.c_str());
                             dl_ciagu++;
                             ciag[dl_ciagu]=a;
                             s=s.substr(k+1);
                            }
                         else
                            {}}}}}}
```

```
        {a=atoi(s.c_str());  
        dl_ciagu++;  
        ciag[dl_ciagu]=a;  
        s="";  
    }  
}  
  
podjazdy=0;  
zjazdy=0;  
for (i=2; i<=dl_ciagu; i++)  
{if (ciag[i]>ciag[i-1])  
{podjazdy++;  
}  
else  
{if (ciag[i]<ciag[i-1])  
{zjazdy++;  
}  
}  
}  
  
wsp_jazdy=(zjazdy*1.0*100)/(podjazdy+zjazdy);  
  
if (wsp_jazdy>=60)  
{cout << numer_ciagu << ". wsp. zjazd. = " << wsp_jazdy << endl;  
plik3 << numer_ciagu << ". wsp. zjazd. = " << wsp_jazdy << endl;  
}  
}  
}  
}  
plik1.close();  
plik3.close();  
return 0;  
}
```

**Odpowiedź:**

1. wsp. zjazd. = 64.2857
3. wsp. zjazd. = 60
13. wsp. zjazd. = 60
15. wsp. zjazd. = 69.2308
18. wsp. zjazd. = 60
22. wsp. zjazd. = 60
24. wsp. zjazd. = 63.6364
28. wsp. zjazd. = 66.6667
35. wsp. zjazd. = 60
40. wsp. zjazd. = 64.2857
45. wsp. zjazd. = 62.5
50. wsp. zjazd. = 62.5

## Zadanie 1.2.

```

int main()
{
    plik3.open("wyniki.txt", ios::out);
    plik1.open("wysokosci_etapow.txt", ios::in);
    wiersz=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                {
                    wiersz++;
                    s2=s;

                    k=s.find(':');
                    s3=s.substr(0,k);
                    numer_ciagu=atoi(s3.c_str());
                    s=s.substr(k+1);

                    dl_ciagu=0;
                    while (s!="")
                        {k=s.find(' ');
                         if (k>0)
                             {s3=s.substr(0,k);
                              a=atoi(s3.c_str());
                              dl_ciagu++;
                              ciag[dl_ciagu]=a;
                              s=s.substr(k+1);
                             }
                         else
                             {a=atoi(s.c_str());
                              dl_ciagu++;
                              ciag[dl_ciagu]=a;
                              s="";
                             }
                        }
                    suma=0;
                    for (i=1; i<=dl_ciagu; i++)
                        {suma=suma+ciag[i];
                     }

                    srednia=(suma*1.0)/dl_ciagu;

                    suma2=0;
                    for (i=1; i<=dl_ciagu; i++)
                        {suma2=suma2+pow(ciag[i]-srednia,2);
                     }

                    odch_stand=sqrt(suma2/dl_ciagu);

                    ile_pkt=0;
                    for (i=1; i<=dl_ciagu; i++)
                        {if ((ciag[i]>=srednia-odch_stand) && (ciag[i]<=srednia+odch_stand))
                            {ile_pkt++;
                         }
                     }
                }
            }
        }
}

```

```
    procent = (ile_pkt*100.0)/dl_ciagu;

    if (procent>=70)
        {cout << numer_ciagu << ". procent=" << procent << endl;
         plik3 << numer_ciagu << ". procent=" << procent << endl;
        }

    }
}

plik1.close();
plik3.close();
return 0;
}
```

**Odpowiedź:**

- 28. procent = 70
- 37. procent = 72.7273
- 41. procent = 70
- 46. procent = 70
- 49. procent = 76.9231

## Zadanie 1.3.

```
int main()
{
    plik3.open("wyniki.txt", ios::out);
    plik1.open("wysokosci_etapow.txt", ios::in);
    wiersz=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                {
                    wiersz++;
                    s2=s;

                    k=s.find(':');
                    s3=s.substr(0,k);
                    numer_ciagu=atoi(s3.c_str());
                    s=s.substr(k+1);

                    dl_ciagu=0;
                    while (s!="")
                        {k=s.find(' ');
                         if (k<0)
                             {s3=s.substr(0,k);
                              a=atoi(s3.c_str());
                              dl_ciagu++;
                              ciag[dl_ciagu]=a;
                              s=s.substr(k+1);
                             }
                         else
                             {a=atoi(s.c_str());
                              dl_ciagu++;
                              ciag[dl_ciagu]=a;
                             }
                        }
                    }
                }
            }
        }
    }
}
```

```

    s="";
}

}

dl_czesci_najdl=0;
dl_czesci=0;
for (i=2; i<=dl_ciagu; i++)
{if (ciag[i]>ciag[i-1])
 {dl_czesci++;
 }
 else
 {if (dl_czesci>dl_czesci_najdl)
 {dl_czesci_najdl=dl_czesci;
 ile_czesci_najdl=1;
 pocz_czesci_najdl[1]=i-dl_czesci-1;
 }
 else
 {if (dl_czesci==dl_czesci_najdl)
 {ile_czesci_najdl++;
 pocz_czesci_najdl[ile_czesci_najdl]=i-dl_czesci-1;
 }
 }
 dl_czesci=0;
}
}

if (dl_czesci>dl_czesci_najdl)
{dl_czesci_najdl=dl_czesci;
ile_czesci_najdl=1;
pocz_czesci_najdl[1]=i-dl_czesci-1;
}

else
{if (dl_czesci==dl_czesci_najdl)
{ile_czesci_najdl++;
pocz_czesci_najdl[ile_czesci_najdl]=i-dl_czesci-1;
}
}

cout << numer_ciagu << ". Najw. dlugosc czesci = " << dl_czesci_najdl
↳<< " Ilosc = " << ile_czesci_najdl << " Poczatki: ";
plik3 << numer_ciagu << ". Najw. dlugosc czesci = " << dl_czesci_najdl
↳<< " Ilosc = " << ile_czesci_najdl << " Poczatki: ";
for (i=1; i<=ile_czesci_najdl; i++)
{cout << pocz_czesci_najdl[i] << " ";
plik3 << pocz_czesci_najdl[i] << " ";
}
cout << endl;
plik3 << endl;

}
}
plik1.close();
plik3.close();
return 0;
}

```

**Odpowiedź:**

1. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 11
2. Najw. dlugosc czesci = 3 Ilosc = 2 Poczatki: 3 14
3. Najw. dlugosc czesci = 3 Ilosc = 1 Poczatki: 7
4. Najw. dlugosc czesci = 2 Ilosc = 2 Poczatki: 5 8
5. Najw. dlugosc czesci = 3 Ilosc = 1 Poczatki: 3
6. Najw. dlugosc czesci = 4 Ilosc = 1 Poczatki: 7
7. Najw. dlugosc czesci = 3 Ilosc = 2 Poczatki: 12 20
8. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 8
9. Najw. dlugosc czesci = 2 Ilosc = 3 Poczatki: 2 12 18
10. Najw. dlugosc czesci = 6 Ilosc = 1 Poczatki: 6
11. Najw. dlugosc czesci = 4 Ilosc = 1 Poczatki: 2
12. Najw. dlugosc czesci = 2 Ilosc = 2 Poczatki: 5 10
13. Najw. dlugosc czesci = 2 Ilosc = 2 Poczatki: 3 10
14. Najw. dlugosc czesci = 2 Ilosc = 3 Poczatki: 5 10 18
15. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 4
16. Najw. dlugosc czesci = 2 Ilosc = 2 Poczatki: 2 6
17. Najw. dlugosc czesci = 4 Ilosc = 1 Poczatki: 14
18. Najw. dlugosc czesci = 3 Ilosc = 1 Poczatki: 3
19. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 5
20. Najw. dlugosc czesci = 3 Ilosc = 1 Poczatki: 26
21. Najw. dlugosc czesci = 3 Ilosc = 3 Poczatki: 8 16 23
22. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 12
23. Najw. dlugosc czesci = 2 Ilosc = 3 Poczatki: 5 8 17
24. Najw. dlugosc czesci = 2 Ilosc = 2 Poczatki: 5 13
25. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 13
26. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 11
27. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 4
28. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 3
29. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 10
30. Najw. dlugosc czesci = 5 Ilosc = 1 Poczatki: 17
31. Najw. dlugosc czesci = 3 Ilosc = 1 Poczatki: 7
32. Najw. dlugosc czesci = 3 Ilosc = 1 Poczatki: 10
33. Najw. dlugosc czesci = 3 Ilosc = 1 Poczatki: 7
34. Najw. dlugosc czesci = 1 Ilosc = 6 Poczatki: 1 3 5 7 9 11
35. Najw. dlugosc czesci = 2 Ilosc = 2 Poczatki: 1 21
36. Najw. dlugosc czesci = 2 Ilosc = 2 Poczatki: 2 11
37. Najw. dlugosc czesci = 2 Ilosc = 2 Poczatki: 4 9
38. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 1
39. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 2
40. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 12

41. Najw. dlugosc czesci = 1 Ilosc = 4 Poczatki: 2 4 7 9  
 42. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 12  
 43. Najw. dlugosc czesci = 1 Ilosc = 5 Poczatki: 1 4 6 9 11  
 44. Najw. dlugosc czesci = 4 Ilosc = 1 Poczatki: 15  
 45. Najw. dlugosc czesci = 2 Ilosc = 3 Poczatki: 3 6 20  
 46. Najw. dlugosc czesci = 1 Ilosc = 4 Poczatki: 1 3 5 9  
 47. Najw. dlugosc czesci = 5 Ilosc = 1 Poczatki: 5  
 48. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 3  
 49. Najw. dlugosc czesci = 2 Ilosc = 2 Poczatki: 2 11  
 50. Najw. dlugosc czesci = 2 Ilosc = 1 Poczatki: 4

## Zadanie 1.4.

```

int main()
{
    plik3.open("wyniki.txt", ios::out);
    plik1.open("wysokosci_etapow.txt", ios::in);
    wiersz=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                {
                    wiersz++;
                    s2=s;

                    k=s.find(':');
                    s3=s.substr(0,k);
                    numer_ciagu=atoi(s3.c_str());
                    s=s.substr(k+1);

                    dl_ciagu=0;
                    while (s!="")
                        {k=s.find(' ');
                         if (k>0)
                             {s3=s.substr(0,k);
                              a=atoi(s3.c_str());
                              dl_ciagu++;
                              ciag[dl_ciagu]=a;
                              s=s.substr(k+1);
                             }
                         else
                             {a=atoi(s.c_str());
                              dl_ciagu++;
                              ciag[dl_ciagu]=a;
                              s="";
                             }
                        }

                    mon_palindrom=true;
                    i=1;
                    while ((i<dl_ciagu/2) && (mon_palindrom==true))

```

```
{if (!( ((ciag[i]>ciag[i+1]) &&
        ↪(ciag[d1_ciagu-i+1]>ciag[d1_ciagu-i])) ||
        ((ciag[i]<ciag[i+1]) && (ciag[d1_ciagu-i+1]<ciag[d1_ciagu-i])) ||
        ↪|||
        ((ciag[i]==ciag[i+1]) &&
        ↪(ciag[d1_ciagu-i+1]==ciag[d1_ciagu-i])) ))
    {mon_palindrom=false;
    }
    i++;
}

if (mon_palindrom==true)
{
    if (d1_ciagu%2==1)
    {if (!( ((ciag[i]>ciag[i+1]) &&
            ↪(ciag[d1_ciagu-i+1]>ciag[d1_ciagu-i])) ||
            ((ciag[i]<ciag[i+1]) &&
            ↪(ciag[d1_ciagu-i+1]<ciag[d1_ciagu-i])) ||
            ((ciag[i]==ciag[i+1]) &&
            ↪(ciag[d1_ciagu-i+1]==ciag[d1_ciagu-i])) ))
        {mon_palindrom=false;
        }
    }
    else
    {if (ciag[i]!=ciag[i+1])
        {mon_palindrom=false;
        }
    }
}

if (mon_palindrom==true)
{cout << numer_ciagu << "." << endl;;
 plik3 << numer_ciagu << "." << endl;;
}
}

plik1.close();
plik3.close();
return 0;
}
```

**Odpowiedź:**

- 16.
- 25.
- 33.
- 34.
- 39.
- 41.
- 42.

## Zadanie 2.1.

<b>n</b>	<b>p</b>	<b>tablica <math>d[1..n][1..n]</math></b>	<b>k</b>	<b>w</b>
5	15	1 0 4 4 3 0 3 0 2 2 1 4 3 0 3 1 4 3 4 1 3 2 3 2 0	2	7
6	10	2 2 0 3 4 3 4 0 0 3 2 4 4 3 4 1 4 3 3 2 0 3 0 0 3 4 3 4 1 0 4 2 4 0 0 3	5	<b>18</b>
7	20	0 0 2 0 0 0 1 2 3 1 0 2 4 0 2 4 1 0 0 3 4 3 1 2 3 0 4 3 4 3 4 0 2 4 4 1 0 2 4 0 3 4 2 2 1 4 1 1 0	0	0

## Zadanie 2.2.

Pierwszy przypadek, gdy liczba wszystkich dzieci na osiedlu jest mniejsza lub równa liczbie prezentów w jednym worku ( $p$ ).

W przypadku, gdy liczba dzieci jest większa od  $p$ , to przy sumowaniu liczby dzieci w kolejnych domach musimy otrzymywać dokładnie  $p$ , z wyjątkiem ostatnich domów.

**Przykład:**

Niech  $p = 10$

Liczba dzieci w odwiedzanych domach: 2 0 2 3 3 0 4 4 1 0 1 2 2 0 2 2 1 1 3 2

Wówczas:  $2 + 0 + 2 + 3 + 3 = 10$ , dalej  $0 + 4 + 4 + 1 + 0 + 1 = 10$ ;  $2 + 2 + 0 + 2 + 2 + 1 + 1 = 10$ , i na koniec  $3 + 2 = 5 < 10$

Przykład, gdy nie wszystkie dzieci dostaną prezenty:

Niech  $p = 10$

Liczba dzieci w odwiedzanych domach: 2 0 2 3 3 0 4 4 3 0 1 2 2 0 2 2 1 1 3 2

Wówczas:  $2 + 0 + 2 + 3 + 3 = 10$ , dalej  $0 + 4 + 4 + 3 = 11 > 10$

## Zadanie 2.3.

Przykładowe rozwiązanie:

```
procedura sprawdz(x, y)
    jeśli a+d[j][i]<p
        a ← a + d[j][i]
    w przeciwnym razie
        jeśli a+d[j][i]>p
            w ← w + d[j][i]
            k ← k + 1
    a ← 0
```

**Algorytm:**

```
a ← 0
w ← 0
k ← 0
x ← 1
dopóki x<=(n+1)/2 wykonuj
    dla każdego i=x, x+1, x+2,...,n-x+1 wykonuj
        sprawdz(i, x)
    dla każdego i=x+1, x+2, x+3,...,n-x+1 wykonuj
        sprawdz(n-x+1, i)
    jeśli n-x+1>x
        dla każdego i=n-x, n-x-1, n-x-2,...,x wykonuj
            sprawdz(i, n-x+1)
        dla każdego i=n-x, n-x-1, n-x-2,...,x+1 wykonuj
            sprawdz(x, i)
    x ← x + 1

procedura sprawdz(x, y)
    jeśli a+d[j][i]<p
        a ← a + d[j][i]
    w przeciwnym razie
        jeśli a+d[j][i]>p
            w ← w + d[j][i]
            k ← k + 1
    a ← 0
```

**Algorytm:**

```
a ← 0
w ← 0
k ← 0
x ← 1
dopóki x<=(n+1)/2 wykonuj
    dla każdego i=x, x+1, x+2,...,n-x+1 wykonuj
        sprawdz(i, x)
    dla każdego i=x+1, x+2, x+3,...,n-x+1 wykonuj
        sprawdz(n-x+1, i)
    jeśli n-x+1>x
        dla każdego i=n-x, n-x-1, n-x-2,...,x wykonuj
            sprawdz(i, n-x+1)
        dla każdego i=n-x, n-x-1, n-x-2,...,x+1 wykonuj
            sprawdz(x, i)
    x ← x + 1
```

## Zadanie 2.4.

Przykładowe rozwiązanie:

```
procedura sprawdz(x, y)
    jeśli a+d[j][i]<p
        a ← a + d[j][i]
    w przeciwnym razie
        jeśli a+d[j][i]>p
            w ← w + d[j][i]
            k ← k + 1
    a ← 0
```

**Algorytm:**

```
a ← 0
w ← 0
k ← 0
x ← 1
y ← 1
dopóki x+1<=(n+1)/2 lub y+1<=(m+1)/2 wykonuj
    dla każdego i=x, x+1, x+2,...,n-x+1 wykonuj
        sprawdz(i, y)
    dla każdego i=y+1, y+2, y+3,...,m-y+1 wykonuj
        sprawdz(n-x+1, i)
    jeśli m-y+1>y
        dla każdego i=n-x, n-x-1, n-x-2,...,x wykonuj
            sprawdz(i, m-y+1)
    jeśli n-x+1>x
        dla każdego i=m-y, m-y-1, m-y-2,...,y+1 wykonuj
            sprawdz(x, i)
        x ← x + 1
        y ← y + 1
```

```
procedura sprawdz(x, y)
    jeśli a+d[j][i]<p
        a ← a + d[j][i]
    w przeciwnym razie
        jeśli a+d[j][i]>p
            w ← w + d[j][i]
            k ← k + 1
    a ← 0
```

**Algorytm:**

```
a ← 0
w ← 0
k ← 0
x ← 1
y ← 1
dopóki x+1<=(n+1)/2 lub y+1<=(m+1)/2 wykonuj
    dla każdego i=x, x+1, x+2,...,n-x+1 wykonuj
        sprawdz(i, y)
    dla każdego i=y+1, y+2, y+3,...,m-y+1 wykonuj
        sprawdz(n-x+1, i)
    jeśli m-y+1>y
        dla każdego i=n-x, n-x-1, n-x-2,...,x wykonuj
```

```

        sprawdz(i, m-y+1)
jeśli n-x+1>x
    dla każdego i=m-y, m-y-1, m-y-2,...,y+1 wykonuj
        sprawdz(x, i)
x ← x + 1
y ← y + 1

```

## Zadanie 3.

W programach zostały zadeklarowane następujące zmienne i struktury:

```

#include <iostream>
#include <windows.h>
#include <fstream>
#include <cmath>
using namespace std;

string s, s1, s2, nazwa;
string bogowie[100];
int liczba_bogow, liczba_hoteli, liczba_niekorz, liczba_korz, najw_liczba_korz;
int i,j,k,x,y,kat;
double odl, najmn_odl, najw_odl, fk, procent;
double fk_najkorz;
string nazwa_najkorz;
int kierunek[2];
int liczba_pkt_najkorz;
int punkty_najkorz[10000];
double najmn_odl_stand[6];
string nazwy_najmn_odl_stand[6][2];

struct hotel
{
    string nazwa;
    int kat,x,y;
};

hotel tab_hoteli[100];
fstream plik1, plik2, plik3;

```

### Zadanie 3.1.

```

int main()
{
    plik1.open("hotele.txt", ios::in);
    liczba_hoteli=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                {
                    s1=s;
                    //nazwa
                    k=s.find(";");
                    nazwa=s.substr(0,k);
                    s=s.substr(k+1);

                    //kategoria
                    k=s.find(";");

```

```

        s2=s.substr(0,k);
        kat=atoi(s2.c_str());
        s=s.substr(k+1);

        //współrzędne
        k=s.find(";");
        s2=s.substr(0,k);
        x=atoi(s2.c_str());
        s=s.substr(k+1);
        y=atoi(s.c_str());

        liczba_hoteli++;
        tab_hoteli[liczba_hoteli].nazwa=nazwa;
        tab_hoteli[liczba_hoteli].kat=kat;
        tab_hoteli[liczba_hoteli].x=x;
        tab_hoteli[liczba_hoteli].y=y;

    }
}
plik1.close();

for (i=1; i<=5; i++)
{ najmn_odl_stand[i]=-1;
}

for (i=2; i<=liczba_hoteli; i++)
{ for (j=1; j<i; j++)
    { if (tab_hoteli[i].kat==tab_hoteli[j].kat)
        {odl=sqrt(pow(tab_hoteli[i].x-tab_hoteli[j].x,2)+pow(tab_hoteli[i].
        y-tab_hoteli[j].y,2));
        if((najmn_odl_stand[tab_hoteli[i].kat]==-1) ||
        (odl<najmn_odl_stand[tab_hoteli[i].kat])) 
        {najmn_odl_stand[tab_hoteli[i].kat]=odl;
        nazwy_najmn_odl_stand[tab_hoteli[i].kat][0]=tab_hoteli[i].nazwa;
        nazwy_najmn_odl_stand[tab_hoteli[i].kat][1]=tab_hoteli[j].nazwa;
        }
    }
}
}

cout << endl << "Najmniejsze odleglosci: " << endl;
for (i=1; i<=5; i++)
{ cout << i << ": odl=" << najmn_odl_stand[i] << " hotele: "
<< nazwy_najmn_odl_stand[i][0] << " i " << nazwy_najmn_odl_stand[i][1] << endl;
}

}

```

**Odpowiedź:**

Najmniejsze odległości:

- 1: odległość = 612,785 hotele: Boreasz i Erato
- 2: odległość = 348,695 hotele: Hypnos i Eris
- 3: odległość = 536,06 hotele: Atropos i Morfeusz
- 4: odległość = 792,934 hotele: Ares i Asklepios
- 5: odległość = 1196,93 hotele: Afrodыта i Artemida

## Zadanie 3.2.

```
int main()
{
    plik1.open("hotele.txt", ios::in);
    liczba_hoteli=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                {
                    s1=s;
                    //nazwa
                    k=s.find(";");
                    nazwa=s.substr(0,k);
                    s=s.substr(k+1);

                    //kategoria
                    k=s.find(";");
                    s2=s.substr(0,k);
                    kat=atoi(s2.c_str());
                    s=s.substr(k+1);

                    //współrzędne
                    k=s.find(";");
                    s2=s.substr(0,k);
                    x=atoi(s2.c_str());
                    s=s.substr(k+1);
                    y=atoi(s.c_str());

                    liczba_hoteli++;
                    tab_hoteli[liczba_hoteli].nazwa=nazwa;
                    tab_hoteli[liczba_hoteli].kat=kat;
                    tab_hoteli[liczba_hoteli].x=x;
                    tab_hoteli[liczba_hoteli].y=y;
                }
            }
        plik1.close();

    liczba_niekorz=0;
    for (i=1; i<=liczba_hoteli; i++)
        {odl=sqrt(pow(tab_hoteli[i].x-500,2)+pow(tab_hoteli[i].y-500,2));
         fk=0.07*tab_hoteli[i].kat-0.00013*odl+0.65;
         if (fk<=0)
            {liczba_niekorz++;
         }

    procent = (liczba_niekorz*100.0)/liczba_hoteli;

    cout << endl << endl;
    cout << "Liczba wszystkich hoteli = " << liczba_hoteli << endl;
    cout << "Liczba hoteli z niekorzystna oferta = " << liczba_niekorz << endl;
    cout << "Procent hoteli z niekorzystna oferta = " << procent << endl;
}
}
```

Odpowiedź:

37,5%

## Zadanie 3.3.

```

int main()
{
    plik1.open("hotele.txt", ios::in);
    liczba_hoteli=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                {
                    s1=s;
                    //nazwa
                    k=s.find(";");
                    nazwa=s.substr(0,k);
                    s=s.substr(k+1);

                    //kategoria
                    k=s.find(";");
                    s2=s.substr(0,k);
                    kat=atoi(s2.c_str());
                    s=s.substr(k+1);

                    //współrzędne
                    k=s.find(";");
                    s2=s.substr(0,k);
                    x=atoi(s2.c_str());
                    s=s.substr(k+1);
                    y=atoi(s.c_str());

                    liczba_hoteli++;
                    tab_hoteli[liczba_hoteli].nazwa=nazwa;
                    tab_hoteli[liczba_hoteli].kat=kat;
                    tab_hoteli[liczba_hoteli].x=x;
                    tab_hoteli[liczba_hoteli].y=y;
                }
            }
        plik1.close();

fk_najkorz=0;
for (i=1; i<liczba_hoteli; i++)
{odl=sqrt(pow(tab_hoteli[i].x-500,2)+pow(tab_hoteli[i].y-500,2));
 fk=0.07*tab_hoteli[i].kat-0.00013*odl+0.65;
 if ((odl<=3000) && (fk>0.5))
    {if (fk>fk_najkorz)
        {fk_najkorz=fk;
         nazwa_najkorz=tab_hoteli[i].nazwa;
        }
    }
}

cout << endl << "Najkorzystniejszy hotel w promieniu 3 km to " << nazwa_najkorz
-><< endl;
}

```

**Odpowiedź:**

Priap.

## Zadanie 3.4.

```
int main()
{
    plik1.open("hotele.txt", ios::in);
    liczba_hoteli=0;
    if(plik1.good() == true)
        {while(!plik1.eof())
            {getline(plik1, s);
             if (s!="")
                {
                    s1=s;
                    //nazwa
                    k=s.find(";");
                    nazwa=s.substr(0,k);
                    s=s.substr(k+1);

                    //kategoria
                    k=s.find(";");
                    s2=s.substr(0,k);
                    kat=atoi(s2.c_str());
                    s=s.substr(k+1);

                    //współrzędne
                    k=s.find(";");
                    s2=s.substr(0,k);
                    x=atoi(s2.c_str());
                    s=s.substr(k+1);
                    y=atoi(s.c_str());

                    liczba_hoteli++;
                    tab_hoteli[liczba_hoteli].nazwa=nazwa;
                    tab_hoteli[liczba_hoteli].kat=kat;
                    tab_hoteli[liczba_hoteli].x=x;
                    tab_hoteli[liczba_hoteli].y=y;
                }
            }
        plik1.close();

    cout << endl;
    najw_liczba_korz=0;
    kierunek[0]=-1;
    kierunek[1]=1;

    for (j=0; j<=1; j++)
        {for (k=2000; k<=5000; k++)
            {liczba_korz=0;
             for (i=1; i<=liczba_hoteli; i++)
                 {odl=sqrt(pow(tab_hoteli[i].x-(500+k*kierunek[j]),2)
                           +pow(tab_hoteli[i].y-500,2));
                  fk=0.07*tab_hoteli[i].kat-0.00013*odl+0.65;
                  if (fk>0)
                      {liczba_korz++;
                       }
                 }
            }
        }
}
```

```
    if (liczba_korz>najw_liczba_korz)
    {najw_liczba_korz=liczba_korz;
     liczba_pkt_najkorz=1;
     punkty_najkorz[1]=500+k*kierunek[j];
    }
else
{if (liczba_korz==najw_liczba_korz)
 {liczba_pkt_najkorz++;
  punkty_najkorz[liczba_pkt_najkorz]=500+k*kierunek[j];
 }
}
}
}

cout << endl << "Liczba polozen najkorzystniejszych = " << liczba_pkt_najkorz
<< endl;
cout << "Liczba korzystnych hoteli = " << najw_liczba_korz << endl;
for (j=1; i<=liczba_pkt_najkorz; i++)
{cout << punkty_najkorz[i] << " ";
 if (j%10==0)
 {cout << endl;
 }
}
}
```

**Odpowiedź:**

Liczba położen najkorzystniejszych = 111

Liczba korzystnych hoteli = 37

-1500 -1501 -1502 -1503 -1504 -1505 -1506 -1507 -1508 -1509  
-1510 -1511 -1512 -1513 -1514 -1515 -1516 -1517 -1518 -1519  
-1520 -1521 -1522 -1523 -1524 -1525 -1526 -1527 -1528 -1529  
-1530 -1531 -1532 -1533 -1534 -1535 -1536 -1537 -1538 -1539  
-1540 -1541 -1542 -1543 -1544 -1545 -1546 -1547 -1548 -1549  
-1550 -1551 -1552 -1553 -1554 -1555 -1556 -1557 -1558 -1559  
-1560 -1561 -1562 -1563 -1564 -1565 -1566 -1567 -1568 -1569  
-1570 -1571 -1572 -1573 -1574 -1575 -1576 2500 2501 2502  
2503 2504 2505 2506 2507 2508 2509 2510 2511 2512  
2513 2514 2515 2516 2517 2518 2519 2520 2521 2522  
2523 2524 2525 2526 2527 2528 2529 2530 2531 2532  
2533

## Zadanie 4.1.

B	C	D	E	F	G	H	I	J	K	L	M
=PRAWY(E5;2)											
1	RAFAŁ	BLASZCZYK	ADRES rafal.s@sprzedaz.sieci.informapol.dk	Kraj dk	Etykiety wiersz	Liczba z Kraj					
2	ROMAN	WITKOWSKI	roman.w@szkolenia.oprogramowanie.informapol.ee	ee	pt	18	It	35			
3	ALEKSANDRA	SZCZEPANIAK	aleszc@szkolenia.grafika.informapol.cz	cz	al	26	sk	32			
4	ROBERT	MICHALSKI	r.michalski@reklamacje.sieci.informapol.ba	ba	at	23	be	29			
5	WITOLD	BORKOWSKI	witoldb@finanse.gry.informapol.si	si	ba	26	gr	29			
6	WŁODZIMIERZ	MICHALSKI	wmichalski@promocje.gry.informapol.it	it	be	29	no	29			
7	JOLANTA	ADAMCZYK	jolanta.adamczyk@reklamacje.grafika.informapol.at	at	ch	22	al	26			
8	KLAUDIA	MROZ	klaudia.m@finanse.oprogramowanie.informapol.ch	ch	cy	14	es	25			
9	MARTYNA	SZCZEPANIAK	m.szczepaniak@sprzedaz.sieci.informapol.is	is	cz	21	md	24			
10	BOŻENA	WOJCIK	bozena.w@sprzedaz.sprzet.informapol.mk	mk	dk	14	mk	24			
11	ANNA	KOWALCZYK	annkow@szkolenia.sprzet.informapol.nl	nl	ee	12	pl				
12	ADRIAN	ADMAMCZYK	adriada@kadry.grafika.informapol.ua	ua	es	25	at	23			
13	ARKADIUSZ	KACZMAREK	a.kaczmarek@kadry.gry.informapol.ua	ua	fi	19	hr	23			
14	JADWIGA	SZYMANSKA	jadwiga.s@sprzedaz.oprogramowanie.informapol.es	es	gb	15	is	23			
15	HALINA	GRABOWSKA	halina.grabowska@szkolenia.grafika.informapol.no	no	gr	29	si	23			
16	MAJA	SADOWSKA	maja.s@finanse.gry.informapol.lt	lt	hr	23	ch	22			
17	MIECZYSŁAW	CZARNECKI	m.czarnecki@kadry.sieci.informapol.nl	nl	hu	19	me	22			
18	CZEZLAW	ŁEJKOWSKI	czezel@promocje.internet.informapol.mk	mk	is	23	nl	22			
19	EMILIA	KAZMIERCZAK	e.kazmierczak@reklamacje.gry.informapol.dk	dk	it	12	ua	22			
20	HANNA	URBANSKA	hurbanska@reklamacje.grafika.informapol.no	no	lt	35	cz	21			
21	MONIKA	OSTROWSKA	monost@sprzedaz.gry.informapol.fi	fi	md	24	fi	19			
22	PIOTR	KALINOWSKI	piotr.k@promocje.internet.informapol.ba	ba	me	22	hu	19			
23	IGNACY	MARCIŃIAK	ignacy.marciniek@reklamacje.oprogramowanie.informapol.ee	ee	mk	24	pt	18			
24	HELENA	KACZMARCZYK	h.kaczmarczyk@kadry.gry.informapol.es	es	nl	22	rs	17			
25	IZABELA	SZYMANSKA	izaszy@szkolenia.sprzet.informapol.al	al	no	29	bg	16			
26	KACPER	KAMINSKI	kacperk@kadry.internet.informapol.sk	sk	pl	24	gb	15			
27	TERESA	MAZUR	tmaszur@szkolenia.gry.informapol.dk	dk	rs	17	cy	14			
28	MALGORZATA	SZEWCCZYK	malgorzata.szewczyk@kadry.sprzet.informapol.es	es	se	14	dk	14			
29	ROMAN	BARANOWSKI	roman.b@promocje.internet.informapol.ee	ee	si	23	se	14			
30	MARCIN	MACIEJEWSKI	mmaciejewski@finanse.sprzet.informapol.gb	gb	sk	32	ee	12			
31	MATEUSZ	WOJCIECHOWSKI	m.wojciechowski@reklamacje.gry.informapol.gr	gr	ua	22	it	12			
32	MAKSYMILIAN	SADOWSKI	maksymilians@szkolenia.systemy.informapol.ch	ch	Suma końcowa 674						

Odpowiedź:

Najwięcej pracowników ma firma na Litwie (lt) — 35.

## Zadanie 4.2.

E	F	G	H	I	J	K	L	M
=ZNAJDŹ(“@”,E5)								
4	ADRES	Pozycja znaku @	Adres bez nazwy konta i znaku @	Posycja pierwszej kropki po znaku @	Dział	Etykiety wiersz	Liczba z Dział	
5	rafal.s@sprzedaz.sieci.informapol.dk	8	sprzedaz.sieci.informapol.dk	9	sprzedaz	finanse	96	
6	roman.w@szkolenia.oprogramowanie.informapol.ee	8	szkolenia.oprogramowanie.informapol.ee	10	szkolenia	kadry	119	
7	aleszc@szkolenia.grafika.informapol.cz	7	szkolenia.grafika.informapol.cz	10	szkolenia	promocje	121	
8	r.michalski@reklamacje.sieci.informapol.ba	12	reklamacje.sieci.informapol.ba	11	reklamacje	reklamacje	112	
9	witoldb@finanse.gry.informapol.si	8	finanse.gry.informapol.si	8	finanse	sprzedaż	113	
10	wmichalski@promocje.gry.informapol.it	11	promocje.gry.informapol.it	9	promocje	szkolenia	113	
11	jadwiga.s@sprzedaz.oprogramowanie.informapol.es	17	reklamacje.grafika.informapol.at	11	reklamacje	finanse	96	
12	halina.grabowska@szkolenia.grafika.informapol.no	10	finanse.grafika.informapol.ch	8	finanse	kadry	119	
13	m.szczepaniak@sprzedaz.sieci.informapol.ls	14	sprzedaz.sieci.informapol.ls	9	sprzedaz	promocje	121	
14	bozena.w@sprzedaz.sprzet.informapol.pl	9	sprzedaz.sprzet.informapol.pl	9	sprzedaz	reklamacje	112	
15	annkow@szkolenia.sprzet.informapol.pl	7	szkolenia.sprzet.informapol.pl	10	szkolenia	sprzedaż	113	
16	adriada@kadry.grafika.informapol.ua	7	kadry.grafika.informapol.us	6	kadry	szkolenia	113	
17	a.kaczmarek@kadry.gry.informapol.ua	12	kadry.gry.informapol.us	6	kadry	finanse	96	
18	jadwiga.s@sprzedaz.oprogramowanie.informapol.es	9	szkolenia.sprzet.informapol.es	9	szkolenia	kadry	119	
19	halina.grabowska@szkolenia.grafika.informapol.no	7	finanse.gry.informapol.lt	8	finanse	promocje	121	
20	maja.s@finanse.gry.informapol.lt	7	finanse.gry.informapol.lt	6	kadry	reklamacje	112	
21	m.czarniecki@kadry.sieci.informapol.pl	12	kadry.sieci.informapol.pl	9	szkolenia	finanse	96	
22	czezel@promocje.internet.informapol.mk	7	promocje.internet.informapol.mk	9	promocje	kadry	119	
23	e.kazmierczak@reklamacje.gry.informapol.dk	14	reklamacje.gry.informapol.dk	11	reklamacje	promocje	121	

Odpowiedź:

Liczba pracowników w poszczególnych działach

Dział	Liczba pracowników w poszczególnych działach
Finanse	96
Kadry	119
Promocje	121
Reklamacje	112
Sprzedaż	113
Szkolenia	113



## Zadanie 4.3.

E	F	G	H	I	J	K	L
ADRES	@	Nazwa konta	Pierwsza postać	Druga postać	Trzecia postać	Czy jedna z postaci?	
6 rafal.b@sprzedaz.sieci.informapol.dk	8 rafal.b	rafal.blaszczyk	r.blaszczyk	rafal.b		1	
7 roman.w@odpowiedniewywag.informapol.ee	8 roman.w	roman.witkowski	r.witkowski	roman.w		1	
8 aleksander.szczepaniak@szkolenia.grafika.informapol.cz	7 aleksander	aleksander.szczepaniak	a.szczepaniak	aleksander.s		0	
9 r.michalski@reklama-i-e.sieci.informapol.ba	7 r.michalski	robert.michalski	r.michalski	robert.michalski		1	
10 witoldb@finanse.gry.informapol.pl	8 witoldb	witold.borkowski	w.borkowski	witoldb		0	
11 w.michalski@finanse.gry.informapol.it	11 w.michalski	włodzimierz.michalski	w.michalski	włodzimierz.m		0	
12 jołanta.adamczyk@reklamacje.grafika.informapol.at	17 jołanta.adamczyk	jołanta.adamczyk	j.adamczyk	jołanta.a		1	
13 klaudia.m@finanse.ooprogramowanie.informapol.ch	10 klaudia.m	klaudia.mroz	k.mroz	klaudia.m		1	
14 m.szczepaniak@sprzedaz.sieci.informapol.ls	14 m.szczepaniak	martyna.szczepaniak	m.szczepaniak	martyna.s		1	
15 bozena.w@sprzedaz.sieci.informapol.mk	9 bozena.w	bozena.wojcik	b.wojcik	bozena.w		1	
16 annkow@szkolenia.sieci.informapol.nl	7 annkow	anna.kowalczyk	a.kowalczyk	anna.k		0	
17 adrada@kadry.grafika.informapol.ua	7 adrada	adrian.adamczyk	a.adamczyk	adrada.a		0	
18 a.kaczmarek@kadry.gry.informapol.ua	12 a.kaczmarek	arkadiusz.kaczmarek	a.kaczmarek	arkadiusz.k		1	
19 jadwiga@sprzedaz.ooprogramowanie.informapol.es	9 jadwiga	jadwiga.szymanska	j.szymanska	jadwiga.s		0	
20 halina.grabowska@szkolenia.grafika.informapol.no	17 halina.grabowska	halina.grabowska	h.grabowska	halina.g		1	
21 maja.s@finanse.gry.informapol.lt	7 maja.s	maja.sadowska	m.sadowska	maja.s		1	
22 m.czarniecki@kadry.sieci.informapol.nl	12 m.czarniecki	mieczysław.czarniecki	m.czarniecki	mieczysław.c		1	
23 czerzal@promocje.internet.informapol.mk	7 czerzal	czesław.lewisiak	c.lewisiak	czesław.z		0	
24 e.kazmierczak@reklamacje.gry.informapol.dk	14 e.kazmierczak	emilia.kazmierczak	e.kazmierczak	emilia.k		1	
25 hurbanska@reklamacje.grafika.informapol.no	10 hurbanska	hanna.urbska	h.urbska	hanna.u		0	
26 monost@sprzedaz.gry.informapol.fi	7 monost	monika.ostrowska	m.ostrowska	monika.o		0	

Odpowiedź:

Liczba kont z poprawnymi nazwami: 356.

## Zadanie 4.4.

D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN
1 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	
2 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36		
3 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36			
4 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36				
5 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36					
6 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36						
7 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36							
8 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36								
9 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36									
10 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36										
11 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36											
12 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36												
13 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36													
14 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36														
15 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36															
16 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36																
17 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36																	
18 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36																		
19 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36																			
20 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36																				
21 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36																					
22 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	23	24	25	26	27	28	29	30	31	32	33	34	35	36																						
23 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	24	25	26	27	28	29	30	31	32	33	34	35	36																							
24 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	25	26	27	28	29	30	31	32	33	34	35	36																								
25 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	26	27	28	29	30	31	32	33	34	35	36																									
26 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	27	28	29	30	31	32	33	34	35	36																										
27 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	28	29	30	31	32	33	34	35	36																											
28 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	29	30	31	32	33	34	35	36																												
29 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	30	31	32	33	34	35	36																													
30 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	31	32	33	34	35	36																														
31 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	32	33	34	35	36																															
32 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	33	34	35	36																																
33 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	34	35	36																																	
34 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	35	36																																		
35 =EJELU.BR/((ZNAJD2(\$D\$5:\$C\$6);0))	36																																			

Odpowiedź:

Liczba haseł, które nie są silne: 353.

## Zadanie 4.5.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1																										
2			Ciąg= Aa0!Ee1@																							
3				=L1ACZ.TEKSTY(C6;C\$C2)																						
4																										
5			HASŁO		A	E	I	O	U	Y	a	e	i	o	u	y	0	1	2	3	4	5	6	7	8	
6	1	7e#I!E152!i?#			13	5	4	0	0	0	14	2	9	0	0	0	15	19	8	0	0	7	0	1	0	0
7	2	\$%?1&9!3!Y@Aa0!Ee1@			14	18	0	0	0	11	12	19	0	0	0	0	16	4	0	9	0	0	0	0	0	7
8	3	Y9!Y!1%@!@U)e\$ol54^2			20	24	15	0	10	1	21	25	11	14	0	0	22	3	19	0	17	16	0	0	0	2
9	4	4AyA01*0\$4UOA^E#61%0e4			2	15	0	8	11	0	24	21	0	0	0	3	5	6	0	0	1	0	17	0	0	0
10	5	\$!Yyy^			7	11	0	0	0	3	8	12	0	0	0	4	9	13	0	0	0	1	0	0	0	0
11	6	u0u6^o*			8	12	0	0	0	9	13	0	6	1	0	2	14	0	0	0	0	4	0	0	0	0
12	7	ESUY?			7	1	0	0	3	4	8	12	0	0	0	0	9	13	0	0	0	0	0	0	0	0
13	8	750!16le@?@Ua0!Ee1@			16	20	0	15	14	0	10	9	0	0	0	0	3	22	5	0	0	2	7	11	0	0
14	9	o@644*3ua!58u0!21UOAo!!			19	20	10	18	17	0	9	29	0	1	8	0	26	16	15	7	4	11	3	0	12	0
15	10	2!IAy@!B89514o@#!Uo!0%7#			5	29	4	21	19	6	15	30	3	14	7	0	27	12	1	0	13	11	0	23	9	10
16	11	'@!16!0%1E55&			13	9	3	0	0	0	14	18	0	2	0	0	6	4	0	0	0	10	5	0	0	0
17	12	A4y!o!Y@%			1	14	6	0	0	7	8	15	0	5	0	3	12	16	0	0	2	0	0	0	0	0
18	13	&e!E%0!A59%ia			7	4	0	0	0	0	12	2	11	5	0	0	15	19	0	0	0	8	0	0	0	9
19	14	e9!Y!Ua0!o*^y			14	18	11	0	6	5	8	1	9	10	0	3	16	20	0	0	7	0	2	0	0	0
20	15	77%!%2553%706%#A\$6!428!0%			17	31	5	0	0	3	28	32	20	0	0	0	13	33	7	10	21	8	14	2	23	0
21	16	Ae09!4\$e#H!o*y			1	19	0	0	0	14	6	2	11	13	0	0	3	21	0	0	6	0	0	0	0	4

T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO	
																	=LICZ.JEŻELI((A6:A16;"<>0"))		Suma =	674		
																	=LICZ.JEŻELI((K6:P6;"<>0"))		Suma (1) =	674		
																	=LICZ.JEŻELI((Q6:Z6;"<>0"))		Suma (0) =	0		
3	4	5	6	7	8	9	!	@	#	\$	%	^	&	*	?	Wielkie	Male	Cyfry	Specjalne	Długość	Czy silne?	
0	0	7	0	1	0	0	16	20	3	0	0	0	0	0	11	3	3	5	4	20	=JEŻELI(ORAZ(AJ6>2;AK6>2;AL6>2;AM6>2;ANG>8);1;0)	
9	0	0	0	0	0	7	8	21	0	1	2	6	5	0	3	3	2	4	7	21		
0	17	16	0	0	0	2	5	9	0	13	7	18	0	0	0	5	4	6	5	27		
0	1	0	17	0	0	0	26	30	16	0	19	14	9	7	0	4	3	4	7	30		
0	0	1	0	0	0	0	10	14	2	0	0	6	0	0	0	3	3	4	14	1		
0	0	0	4	0	0	0	11	15	0	0	0	0	0	0	5	0	2	4	3	15	1	
0	0	0	0	0	0	0	10	14	0	2	0	0	0	0	6	4	2	2	4	14	1	
0	0	2	7	11	0	0	4	23	0	0	0	0	0	0	1	4	2	6	3	23	1	
7	4	11	3	0	12	0	22	2	0	0	0	0	0	0	6	0	5	4	8	31	1	
0	13	11	0	23	9	10	2	8	16	0	22	0	24	0	0	6	5	8	5	32	1	
0	0	10	5	0	0	0	16	20	0	11	7	1	12	0	0	3	3	4	6	20	1	
0	2	0	0	0	0	0	4	17	0	0	9	0	0	0	0	4	4	3	3	17	1	
0	0	8	0	0	0	9	16	20	0	0	6	3	1	0	0	2	4	4	5	20	1	
0	7	0	2	0	0	0	17	21	4	0	0	0	0	12	0	5	5	4	4	21	1	
10	21	8	14	2	23	0	24	34	16	18	4	0	0	0	1	4	3	9	6	34	1	
0	6	0	0	0	0	4	18	22	9	7	5	12	0	0	0	3	4	4	6	22	1	

**Odpowiedź:**

Najkrótszy ciąg do dodania: Aa0!Ee1@

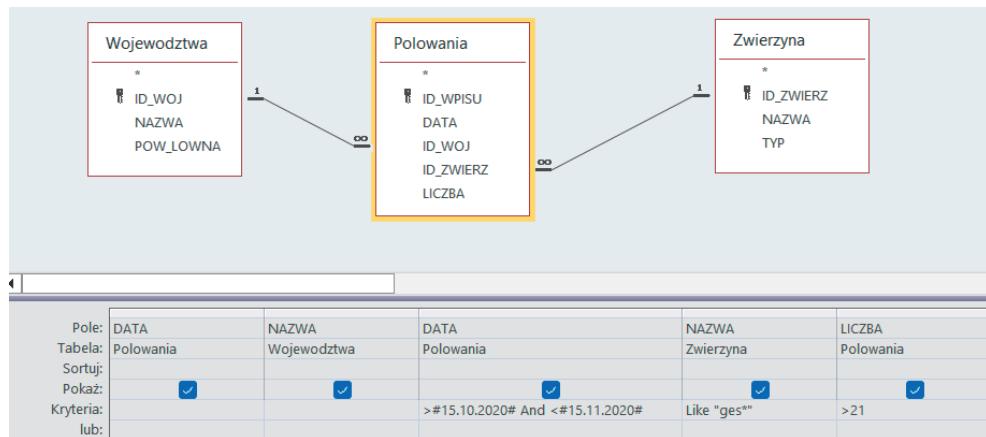
## Zadanie 5.1.

```
SELECT TYP, Count(ID_ZWIERZ)
FROM Zwierzyna
GROUP BY TYP;
```

**Odpowiedź:**

Typ zwierzyny	Liczba gatunków
Drobna	12
Gruba	6

## Zadanie 5.2.



```

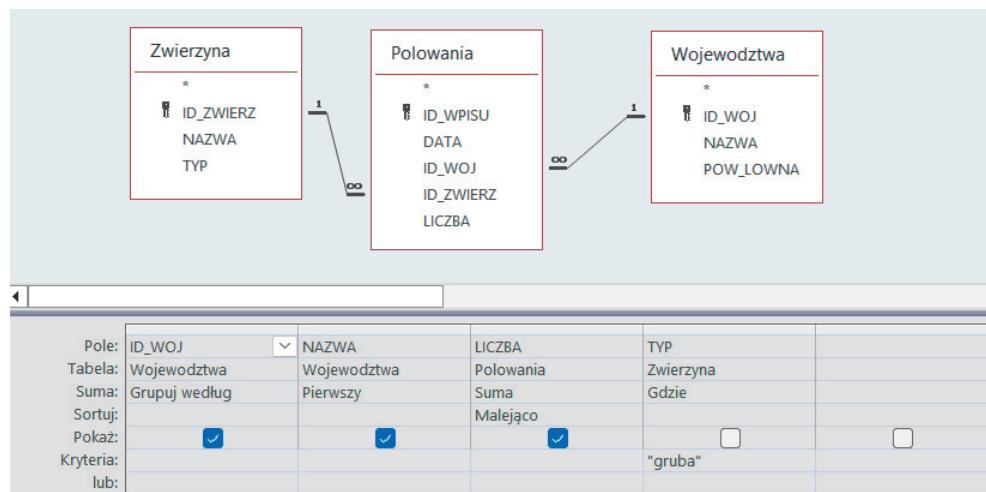
SELECT Polowania.DATA, Wojewodztwa.NAZWA, Polowania.DATA, Zwierzyna.NAZWA,
    ↪Polowania.LICZBA
FROM Zwierzyna INNER JOIN (Wojewodztwa INNER JOIN Polowania ON Wojewodztwa.ID_WOJ =
    ↪Polowania.ID_WOJ) ON Zwierzyna.ID_ZWIERZ = Polowania.ID_ZWIERZ
WHERE ((Polowania.DATA)>#10/15/2020# And (Polowania.DATA)<#11/15/2020#) AND
    ↪((Zwierzyna.NAZWA) Like "ges*") AND ((Polowania.LICZBA)>21));

```

**Odpowiedź:**

20 października 2020, 23 października 2020, 24 października 2020.

## Zadanie 5.3.



```
SELECT Wojewodztwa.ID_WOJ, First(Wojewodztwa.NAZWA) AS PierwszyOfNAZWA,  
      Sum(Polowania.LICZBA) AS SumaOfLICZBA  
  FROM Zwierzyna INNER JOIN (Wojewodztwa INNER JOIN Polowania ON Wojewodztwa.ID_WOJ =  
      Polowania.ID_WOJ) ON Zwierzyna.ID_ZWIERZ = Polowania.ID_ZWIERZ  
 WHERE ((Zwierzyna.TYP)="gruba")  
 GROUP BY Wojewodztwa.ID_WOJ  
 ORDER BY Sum(Polowania.LICZBA) DESC;
```

**Odpowiedź:**

Województwo	Ilość zwierzyny
zachodniopomorskie	857
dolnoslaskie	693
wielkopolskie	682
warmińsko-mazurskie	575
mazowieckie	557
kujawsko-pomorskie	538
pomorskie	513
lubuskie	499
świetokrzyskie	444
małopolskie	390
lódzkie	376
opolskie	338
śląskie	324
podkarpackie	292
lubelskie	237
podlaskie	202

## Zadanie 5.4.

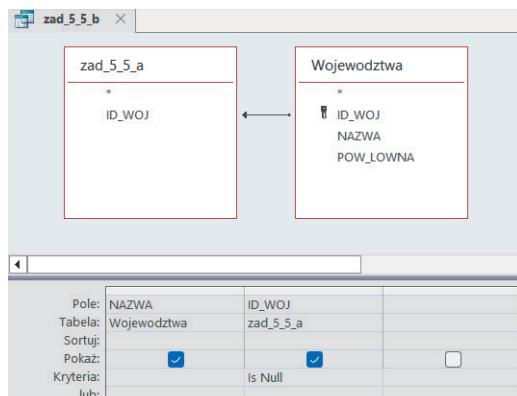
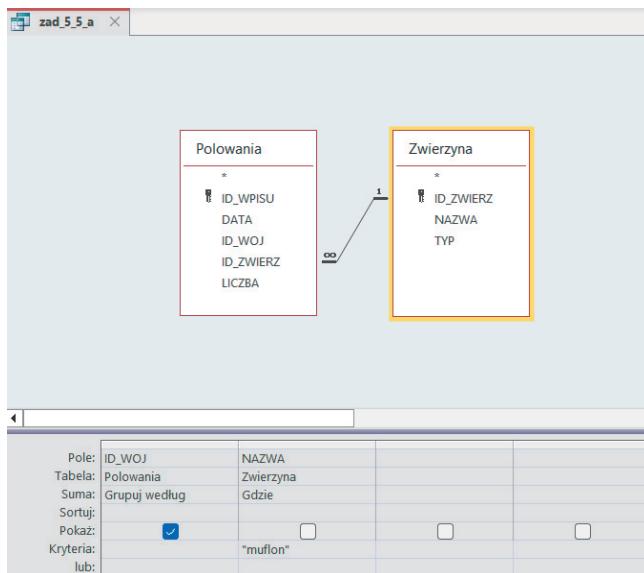
```
SELECT Avg(Polowania.LICZBA) AS Średnia_LICZBA  
  FROM Zwierzyna INNER JOIN Polowania ON Zwierzyna.ID_ZWIERZ = Polowania.ID_ZWIERZ  
 WHERE ((Zwierzyna.NAZWA)="sarna"));
```

**Odpowiedź:**

49,55

## Zadanie 5.5.

Pomocnicza kwerenda:



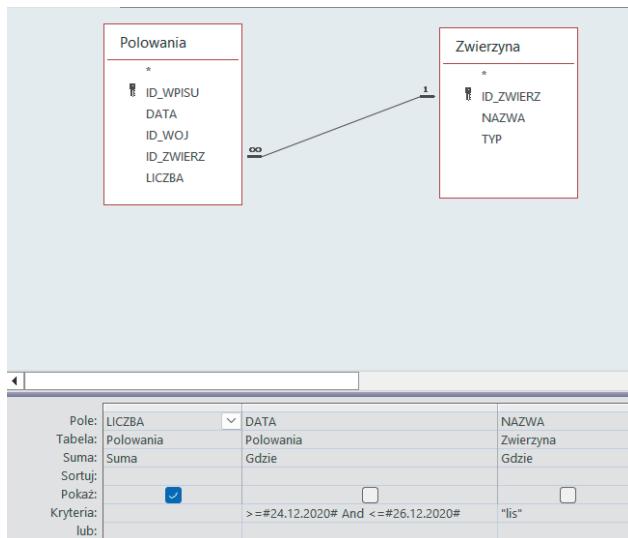
```
SELECT Polowania.ID_WOJ
FROM Zwierzyna INNER JOIN Polowania ON Zwierzyna.ID_ZWIERZ = Polowania.ID_ZWIERZ
WHERE (((Zwierzyna.NAZWA)="muflon"))
GROUP BY Polowania.ID_WOJ;
```

```
SELECT Wojewodztwa.NAZWA, zad_5_5_a.ID_WOJ
FROM zad_5_5_a RIGHT JOIN Wojewodztwa ON zad_5_5_a.ID_WOJ = Wojewodztwa.ID_WOJ
WHERE ((zad_5_5_a.ID_WOJ Is Null));
```

**Odpowiedź:**

lódzkie, mazowieckie.

## Zadanie 5.6.

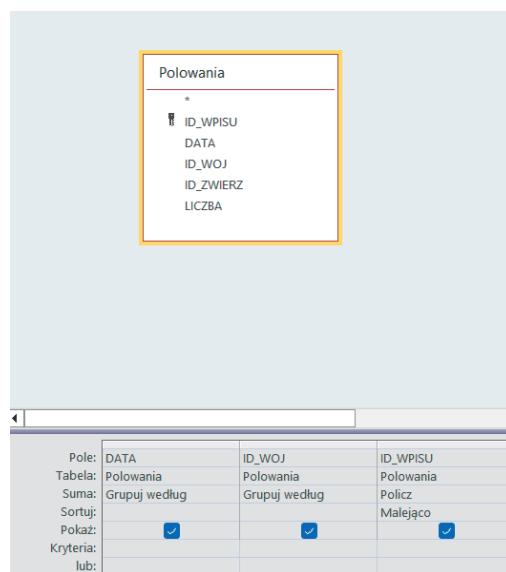


```
SELECT Sum(Polowania.LICZBA) AS SumaOfLICZBA
FROM Zwierzyna INNER JOIN Polowania ON Zwierzyna.ID_ZWIERZ = Polowania.ID_ZWIERZ
WHERE (((Polowania.DATA)>=#12/24/2020# And (Polowania.DATA)<=#12/26/2020#) AND
      ((Zwierzyna.NAZWA)="lis"));
```

**Odpowiedź:**

36

## Zadanie 5.7.



```

SELECT Polowania.DATA, Polowania.ID_WOJ, Count(Polowania.ID_WPISU) AS
    ↳PoliczOfID_WPISU
FROM Polowania
GROUP BY Polowania.DATA, Polowania.ID_WOJ
ORDER BY Count(Polowania.ID_WPISU) DESC;

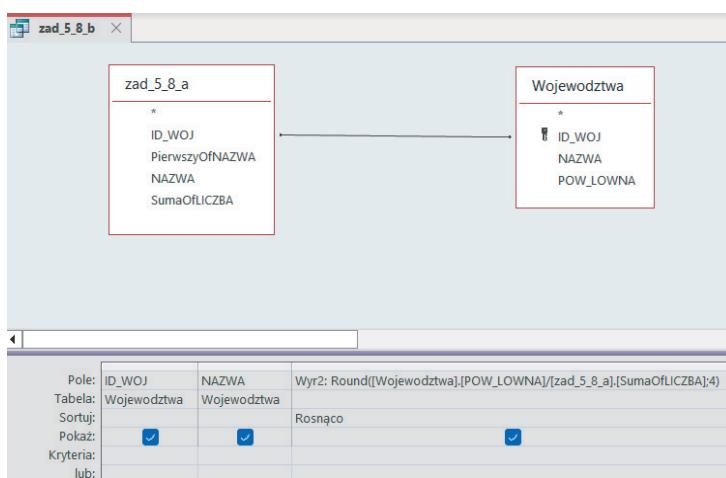
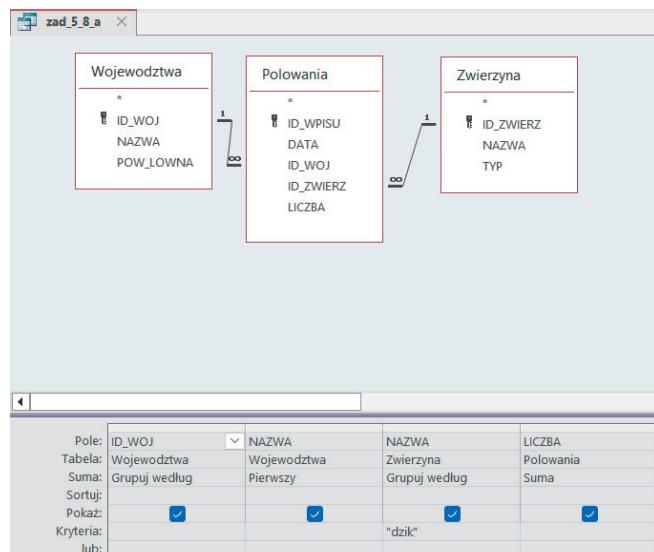
```

**Odpowiedź:**

6 grudnia 2020, 5 różnych gatunków.

**Zadanie 5.8.**

Pomocnicza kwerenda:



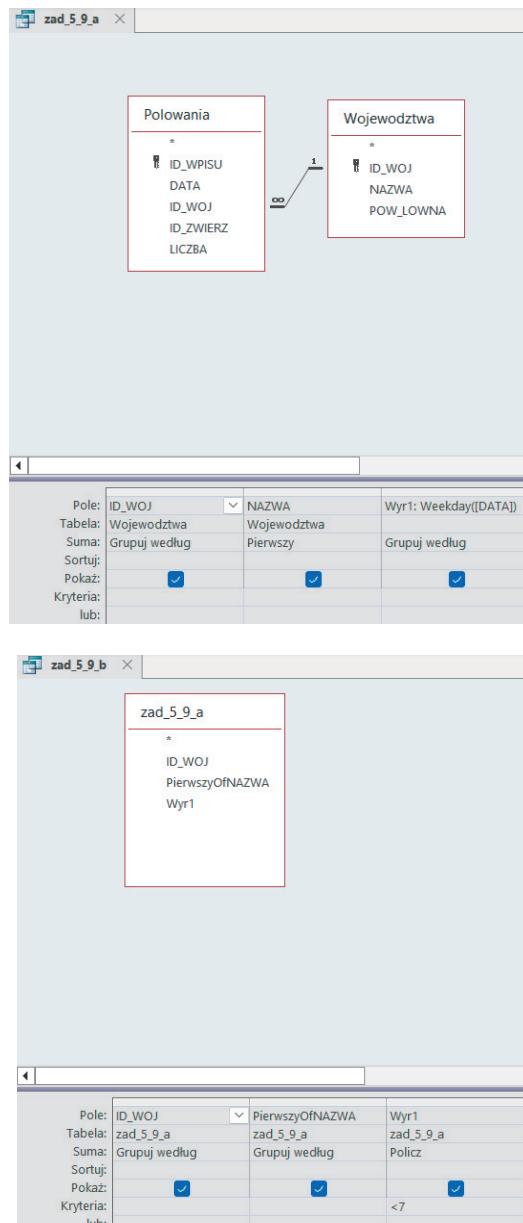
```
SELECT Wojewodztwa.ID_WOJ, First(Wojewodztwa.NAZWA) AS PierwszyOfNAZWA,  
↳ Zwierzyna.NAZWA, Sum(Polowania.LICZBA) AS SumaOfLICZBA  
FROM Zwierzyna INNER JOIN (Wojewodztwa INNER JOIN Polowania ON Wojewodztwa.ID_WOJ =  
↳ Polowania.ID_WOJ) ON Zwierzyna.ID_ZWIERZ = Polowania.ID_ZWIERZ  
GROUP BY Wojewodztwa.ID_WOJ, Zwierzyna.NAZWA  
HAVING (((Zwierzyna.NAZWA)="dzik"));  
  
SELECT Wojewodztwa.ID_WOJ, Wojewodztwa.NAZWA,  
↳ Round([Wojewodztwa].[POW_LOWNA]/[zad_5_8_a].[SumaOfLICZBA],4) AS Wyr2  
FROM zad_5_8_a INNER JOIN Wojewodztwa ON zad_5_8_a.ID_WOJ = Wojewodztwa.ID_WOJ  
ORDER BY Round([Wojewodztwa].[POW_LOWNA]/[zad_5_8_a].[SumaOfLICZBA],4);
```

**Odpowiedź:**

Województwo	Liczba km/dzika
kujawsko-pomorskie	9,0542
swietokrzyskie	9,2254
mazowieckie	12,8502
lodzkie	13,7167
zachodniopomorskie	14,0473
slaskie	16,0262
dolnoslaskie	17,146
malopolskie	20,1111
warmińsko-mazurskie	20,8795
pomorskie	22,7381
lubuskie	26,3031
podkarpackie	29,4639
opolskie	32,3472
lubelskie	35,3763
podlaskie	43,3295
wielkopolskie	75,2386

## Zadanie 5.9.

Pomocnicza kwerenda:



```

SELECT Wojewodztwa.ID_WOJ, First(Wojewodztwa.NAZWA) AS PierwszyOfNAZWA,
       Weekday([DATA]) AS Wyr1
FROM Wojewodztwa INNER JOIN Polowania ON Wojewodztwa.ID_WOJ = Polowania.ID_WOJ
GROUP BY Wojewodztwa.ID_WOJ, Weekday([DATA]);
    
```

```

SELECT zad_5_9_a.ID_WOJ, zad_5_9_a.PierwszyOfNAZWA, Count(zad_5_9_a.Wyr1) AS
→PoliczOfWyr1
FROM zad_5_9_a
GROUP BY zad_5_9_a.ID_WOJ, zad_5_9_a.PierwszyOfNAZWA
HAVING (((Count(zad_5_9_a.Wyr1))<7));

```

**Odpowiedź:**

swietokrzyskie, lodzkie, podlaskie.

**Zadanie 5.10.**

Pomocnicze kwerendy:

**Diagram przedstawiający struktury tabel w zadaniu 5.10.a:**

```

    graph LR
        subgraph zad_5_10_a [ ]
            direction TB
            subgraph Polowania
                direction TB
                ID_WPISU[ID_WPISU]
                DATA[DATA]
                ID_WOJ[ID_WOJ]
                ID_ZWIERZ[ID_ZWIERZ]
                LICZBA[LICZBA]
            end
            subgraph Wojewodztwa
                direction TB
                ID_WOJ[! ID_WOJ]
                NAZWA[NAZWA]
                POW_LOWNA[POW_LOWNA]
            end
            Polowania -- "1 -> n" --> Wojewodztwa
        end
    
```

**Okno kwerendy dla zadania 5.10.a:**

Pole:	ID_WOJ	NAZWA	DATA
Tabela:	Wojewodztwa	Wojewodztwa	Polowania
Suma:	Grupuj według	Pierwszy	Grupuj według
Sortuj:			
Pokaz:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kryteria:			
lub:			

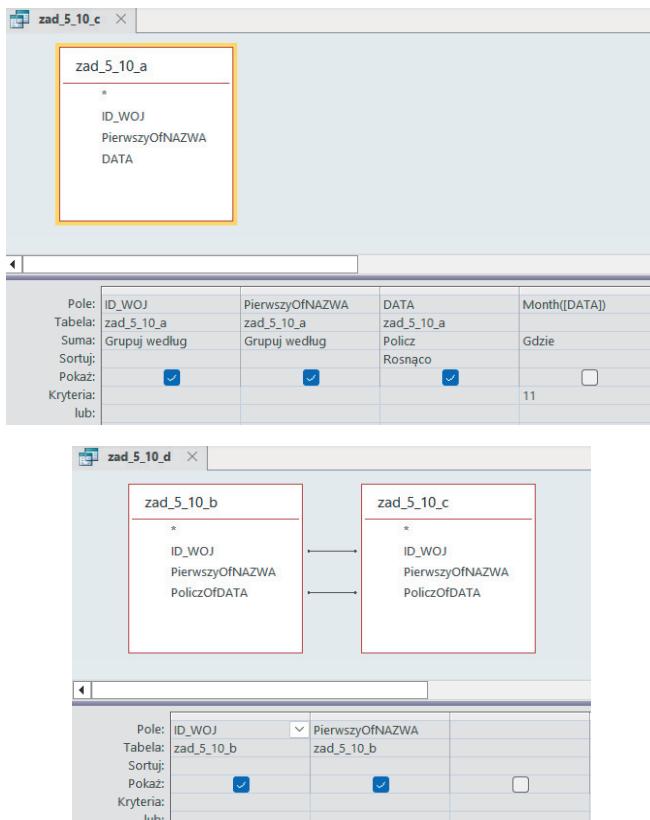
**Diagram przedstawiający struktury tabel w zadaniu 5.10.b:**

```

    graph LR
        subgraph zad_5_10_b [ ]
            direction TB
            subgraph zad_5_10_a
                direction TB
                ID_WOJ[! ID_WOJ]
                PierwszyOfNAZWA[PierwszyOfNAZWA]
                DATA[DATA]
            end
            zad_5_10_a -- "zdefiniowana ponownie" --> zad_5_10_a
        end
    
```

**Okno kwerendy dla zadania 5.10.b:**

Pole:	ID_WOJ	PierwszyOfNAZWA	DATA	Month([DATA])
Tabela:	zad_5_10_a	zad_5_10_a	zad_5_10_a	Policz
Suma:	Grupuj według	Grupuj według	Grupuj według	Gdzie
Sortuj:			Rosnąco	
Pokaz:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Kryteria:				
lub:	10			



```

SELECT Wojewodztwa.ID_WOJ, First(Wojewodztwa.NAZWA) AS PierwszyOfNAZWA,
    ↳Polowania.DATA
FROM Wojewodztwa INNER JOIN Polowania ON Wojewodztwa.ID_WOJ = Polowania.ID_WOJ
GROUP BY Wojewodztwa.ID_WOJ, Polowania.DATA;

SELECT zad_5_10_a.ID_WOJ, zad_5_10_a.PierwszyOfNAZWA, Count(zad_5_10_a.DATA) AS
    ↳PoliczOfDATA
FROM zad_5_10_a
WHERE ((Month([DATA]))=10))
GROUP BY zad_5_10_a.ID_WOJ, zad_5_10_a.PierwszyOfNAZWA
ORDER BY Count(zad_5_10_a.DATA);

SELECT zad_5_10_a.ID_WOJ, zad_5_10_a.PierwszyOfNAZWA, Count([zad_5_10_a].DATA) AS
    ↳PoliczOfDATA
FROM zad_5_10_a
WHERE ((Month([DATA]))=11))
GROUP BY zad_5_10_a.ID_WOJ, zad_5_10_a.PierwszyOfNAZWA
ORDER BY Count([zad_5_10_a].DATA);

SELECT zad_5_10_b.ID_WOJ, zad_5_10_b.PierwszyOfNAZWA
FROM zad_5_10_b INNER JOIN zad_5_10_c ON
(zad_5_10_b.PoliczOfDATA=[zad_5_10_c].PoliczOfDATA) AND
(zad_5_10_b.ID_WOJ=[zad_5_10_c].ID_WOJ);

```

### Odpowiedź:

swietokrzyskie, lubuskie, kujawsko-pomorskie.

# PROGRAM PARTNERSKI

— GRUPY HELION —



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!  
<http://program-partnerski.helion.pl>

GRUPA  
**Helion**