CENTRALNA KOMISJA EGZAMINACYJNA

Arkusz zawiera informacje prawnie chronione do momentu rozpoczęcia egzaminu.

WYPEŁNIA ZDAJĄCY

KOD | **PESEL**

Miejsce na naklejkę.
Sprawdź, czy kod na naklejce to **E-100**.
Jeżeli tak – przyklej naklejkę.
Jeżeli nie – zgłoś to nauczycielowi.

Egzamin maturalny ***Formuła 2015***

INFORMATYKA

Poziom rozszerzony

Część I

WYPEŁNIA ZDAJĄCY

WYBRANE:

.....
(system operacyjny)

.....
(program użytkowy)

.....
(środowisko programistyczne)

Symbol arkusza
EINP-R1-**100**-2306

DATA: **16 czerwca 2023 r.**

GODZINA ROZPOCZĘCIA: **9:00**

CZAS TRWANIA: **60 minut**

LICZBA PUNKTÓW DO UZYSKANIA: **15**

Przed rozpoczęciem pracy z arkuszem egzaminacyjnym

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

Układ graficzny
© CKE 2022

Instrukcja dla zdającego

1. Sprawdź, czy arkusz egzaminacyjny zawiera 12 stron. Ewentualny brak zgłoś przewodniczącemu zespołu nadzorującego egzamin.
2. Na pierwszej stronie oraz na karcie odpowiedzi wpisz swój numer PESEL i przyklej naklejkę z kodem.
3. Wpisz zadeklarowane (wybrane) przez Ciebie na egzamin: system operacyjny, program użytkowy oraz środowisko programistyczne.
4. Odpowiedzi i rozwiązania zapisz w miejscu na to przeznaczonym przy każdym zadaniu.
5. Pisz czytelnie. Używaj długopisu/pióra tylko z czarnym tuszem/atramentem.
6. Nie używaj korektora, a błędne zapisy wyraźnie przekreśl.
7. Pamiętaj, że zapisy w brudnopisie nie będą oceniane.
8. Nie wpisuj żadnych znaków w części przeznaczonej dla egzaminatora.
9. Możesz korzystać z kalkulatora prostego.

Zadania egzaminacyjne są wydrukowane na następnych stronach.

Zadanie 1. Liczby „przyjaciółki”

Dwie dodatnie liczby całkowite *a* i *b* nazwiemy *przyjaciółkami*, jeśli suma cyfr liczby *a* jest taka sama jak suma cyfr liczby *b*. Natomiast parę liczb *a* i *b* nazwiemy *dobrymi przyjaciółkami*, jeżeli są *przyjaciółkami* oraz pierwsza cyfra w zapisie dziesiętnym liczby *a* jest taka sama jak ostatnia cyfra w zapisie dziesiętnym liczby *b* lub odwrotnie (pierwsza cyfra w zapisie dziesiętnym liczby *b* jest taka sama jak ostatnia cyfra w zapisie dziesiętnym liczby *a*).

Przyjaciółkami są liczby 15 i 24, ale nie są *dobrymi przyjaciółkami*.
Dobrymi przyjaciółkami są natomiast liczby 124 i 61 oraz 431 i 17.

Zadanie 1.1. (0–2)

Uzupełnij poniższą tabelę.

a	***b***	**Suma cyfr liczby *a***	**Suma cyfr liczby *b***	**Przyjaciółki (PRAWDA/FAŁSZ)**	**Dobre przyjaciółki (PRAWDA/FAŁSZ)**
433	352	10	10	PRAWDA	PRAWDA
131	50				
416	84				
32	221				

Zadanie 1.2. (0–4)

Zapisz w pseudokodzie lub w wybranym języku programowania algorytm, który dla danych liczb *a* i *b* sprawdzi, czy są *dobrymi przyjaciółkami*, zgodnie z regułami podanymi wcześniej.

Uwaga: Twój algorytm może używać wyłącznie zmiennych przechowujących liczby całkowite oraz może operować wyłącznie na liczbach całkowitych. W zapisie algorytmu możesz korzystać tylko z instrukcji sterujących, operatorów arytmetycznych: dodawania, odejmowania, mnożenia, dzielenia, dzielenia całkowitego i reszty z dzielenia; operatorów logicznych, porównań i instrukcji przypisywania lub samodzielnie napisanych funkcji i procedur wykorzystujących powyższe operacje. **Zabronione** jest używanie funkcji wbudowanych oraz operatorów innych niż wymienione, dostępnych w językach programowania. Nie wolno w szczególności korzystać z żadnych funkcji zamiany z typu znakowego lub napisowego na liczbowy i odwrotnie.

Specyfikacja:

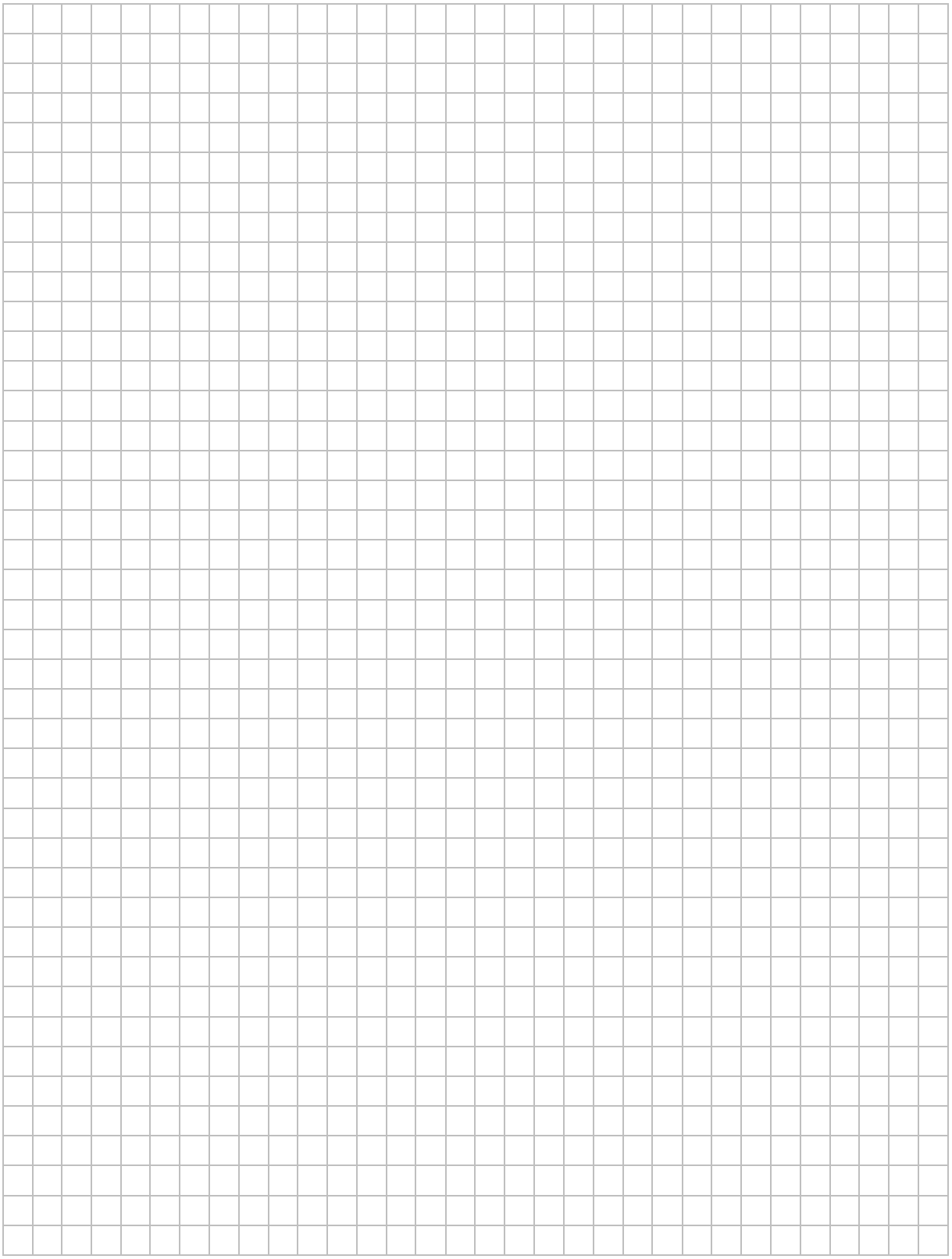
Dane:

a, *b* – dodatnie liczby całkowite

Wynik:

PRAWDA, gdy liczby *a* i *b* są dobrymi przyjaciółkami, albo FAŁSZ – w przeciwnym przypadku

Miejsce na zapis algorytmu:



Zadanie 2. Mnożenie

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

```
iloczyn(x, y):
    jeżeli y = 1
        wynikiem jest x
    w przeciwnym razie
        k ← y div 2
        z ← iloczyn(x, k)
        jeżeli y mod 2 = 0
            wynikiem jest z + z
        w przeciwnym razie
            wynikiem jest x + z + z
```

Uwaga: *x mod y* oznacza resztę z dzielenia *x* przez *y*, natomiast *x* div *y* oznacza wynik dzielenia całkowitego *x* przez *y.*

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

Przykład 1.
Dla liczb x = 9 i y = 11 algorytm wykonuje 5 dodawań. Działanie funkcji *iloczyn*(9, 11) można zilustrować w następujący sposób (w nawiasach obok wskazano liczbę wykonywanych operacji dodawania):

iloczyn(9, 11) = 9 + *z* + *z*, (dwa dodawania)
gdzie *z* = iloczyn(9, 5)

iloczyn(9, 5) = 9 + *z* + *z*, (dwa dodawania)
gdzie *z* = iloczyn(9, 2)

iloczyn(9, 2) = *z* + *z*, (jedno dodawanie)
gdzie *z* = iloczyn(9, 1)

iloczyn(9, 1) = 9

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

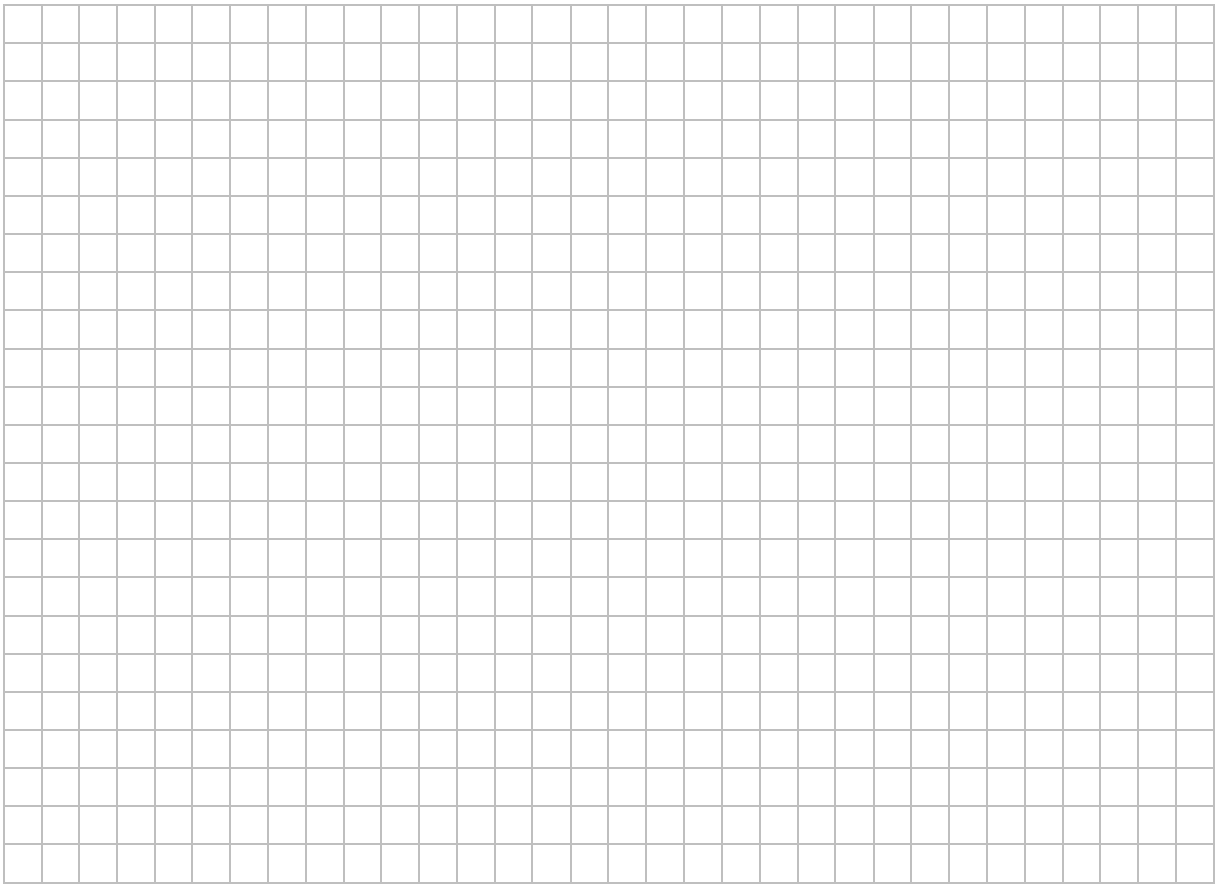
Numer wywołania	**Parametry wywołania**		**Obliczone *k*, *z***		**Wynik**
	x	***y***	***k***	***z***	
1	9	11	5	45	**99** (9+45+45)
2	9	5	2	18	**45** (9+18+18)
3	9	2	1	9	**18** (9+9)
4	9	1	–	–	**9**

Zadanie 2.1. (0–2)

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

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

Miejsce na obliczenia:

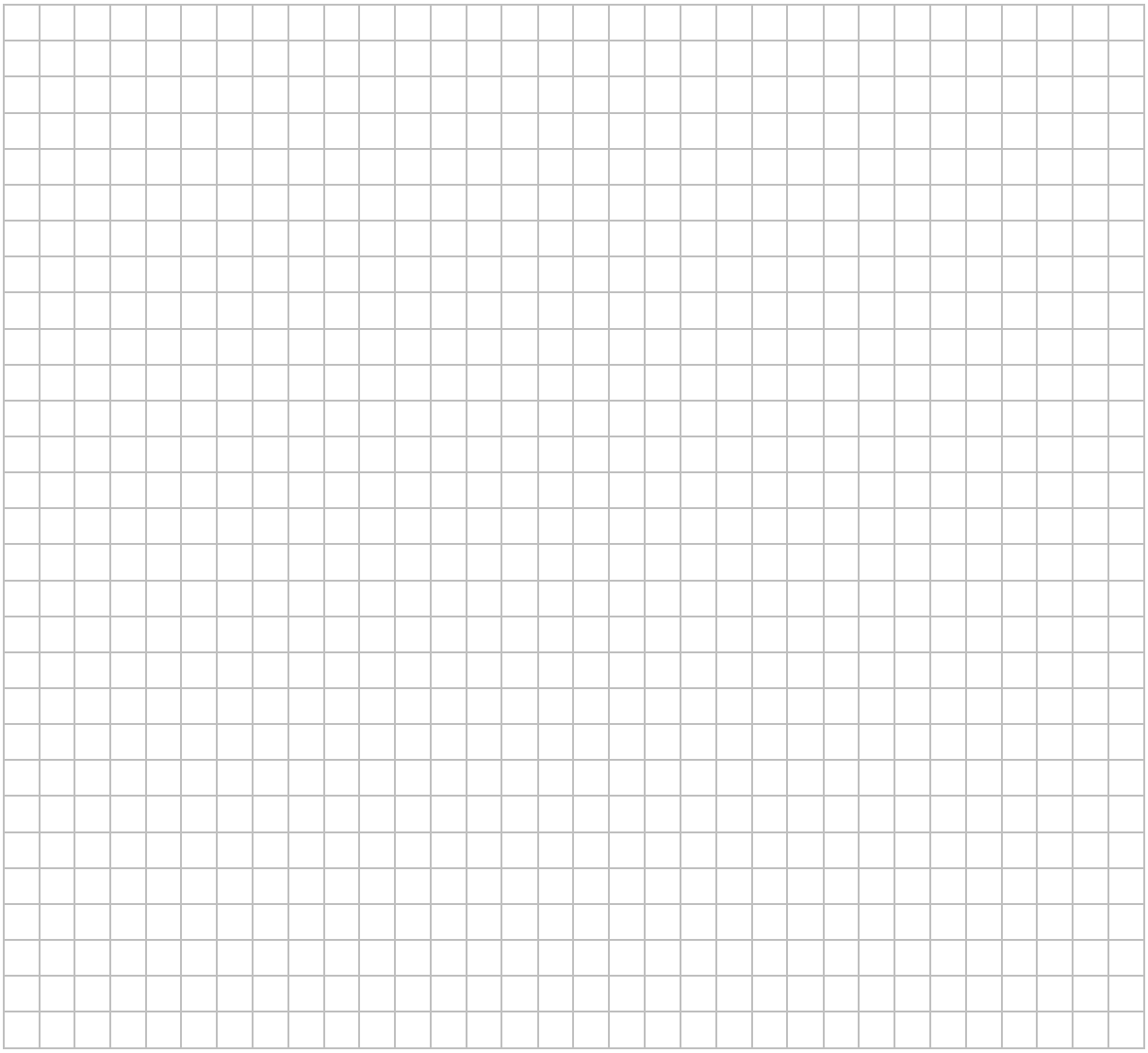


Zadanie 2.2. (0–2)

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

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

Miejsce na obliczenia:



Zadanie 2.3. (0–2)

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

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

Specyfikacja:

Dane:

x, y – liczby całkowite dodatnie

Wynik:

z – wartość iloczynu *x*y*

Algorytm:

z ← _____

dopóki _____ wykonuj

jeżeli *y* mod 2 = 1

z ← *z* + *x*

x ← *x* + *x*

y ← _____

Zadanie 3. Test

Oceń prawdziwość podanych zdań. Zaznacz **P**, jeśli zdanie jest prawdziwe, albo **F** – jeśli jest fałszywe.
W każdym zadaniu punkt uzyskasz tylko za komplet poprawnych odpowiedzi.

Zadanie 3.1. (0–1)

1.	$101_{16} = 11001011_2$	**P**	**F**
2.	$101_{16} = 401_8$	**P**	**F**
3.	$401_8 = 100000001_2$	**P**	**F**
4.	$101_8 = 41_{16}$	**P**	**F**

Informacja do zadań 3.2. i 3.3.

W bazie danych znajdują się tabele *zwierzeta* oraz *osoby.*
Tabela *zwierzeta* zawiera dane: identyfikator zwierzęcia (*id* – klucz podstawowy), gatunek (*gatunek*), imię zwierzęcia (*imie*) oraz identyfikator właściciela zwierzęcia (*id_osoby*).
Tabela *osoby*, z danymi właścicieli zwierząt, zawiera dane: identyfikator osoby (*id_osoby* – klucz podstawowy), imię i nazwisko osoby (*imie*, *nazwisko*).

Poniżej podano kilka przykładowych danych z obu tabel.

Tabela *zwierzeta*

id	gatunek	imie	id_osoby
1	pies	Azor	1
2	kot	Mruczek	2
3	kot	Kicia	3
4	pies	Pimpek	2

Tabela *osoby*

id_osoby	imie	nazwisko
1	Jan	Nowak
2	Adam	Kowalski
3	Anna	Nowakowska

Zadanie 3.2. (0–1)

1.	W wyniku zapytania: SELECT DISTINCT nazwisko FROM osoby; otrzymamy listę różnych nazwisk osób z tabeli *osoby* (nazwiska nie będą się powtarzać).	**P**	**F**
2.	W wyniku zapytania: SELECT nazwisko FROM osoby WHERE nazwisko LIKE "A"; Otrzymamy listę nazwisk zaczynających się od litery A.	**P**	**F**
3.	W wyniku zapytania: SELECT imie FROM osoby WHERE imie BETWEEN "A" AND "C"; otrzymamy listę imion zaczynających się od litery A lub B.	**P**	**F**
4.	W wyniku zapytania: SELECT imie, nazwisko FROM osoby WHERE imie IN ("Ala","Ula"); otrzymamy listę imion i nazwisk wszystkich osób mających imiona, które w kolejności alfabetycznej znajdują się pomiędzy „Ala” a „Ula”.	**P**	**F**

Zadanie 3.3. (0–1)

1.	W wyniku zapytania: SELECT count(DISTINCT gatunek) FROM zwierzeta; otrzymamy liczbę różnych nazw gatunków zwierząt z tabeli *zwierzeta.*	**P**	**F**
2.	W wyniku zapytania: SELECT count(gatunek) FROM zwierzeta WHERE gatunek = "pies"; otrzymamy liczbę psów w tabeli *zwierzeta.*	**P**	**F**
3.	W wyniku zapytania: SELECT count(*) FROM zwierzeta INNER JOIN osoby ON zwierzeta.id_osoby = osoby.id_osoby WHERE gatunek = "pies"; otrzymamy liczby psów z podziałem na różnych właścicieli.	**P**	**F**
4.	W wyniku zapytania: SELECT nazwisko, count(*) FROM zwierzeta INNER JOIN osoby ON zwierzeta.id_osoby = osoby.id_osoby GROUP BY osoby.id_osoby; otrzymamy zestawienie z nazwiskami osób i liczbami ich zwierząt.	**P**	**F**

BRUDNOPIS *(nie podlega ocenie)*

INFORMATYKA

Poziom rozszerzony

Formuła 2015

INFORMATYKA

Poziom rozszerzony

Formuła 2015

INFORMATYKA

Poziom rozszerzony

Formuła 2015