

**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 Z INFORMATYKI

## POZIOM ROZSZERZONY

### CZĘŚĆ I

DATA: **14 czerwca 2022 r.**

GODZINA ROZPOCZĘCIA: **9:00**

CZAS PRACY: **60 minut**

LICZBA PUNKTÓW DO UZYSKANIA: **15**

**WYPEŁNIA ZDAJĄCY**

**WYBRANE:**

.....  
(system operacyjny)

.....  
(program użytkowy)

.....  
(środowisko programistyczne)

#### Instrukcja dla zdającego

1. Sprawdź, czy arkusz egzaminacyjny zawiera 10 stron (zadania 1–3).  
Ewentualny brak zgłoś przewodniczącemu zespołu nadzorującego egzamin.
2. Rozwiązania i odpowiedzi zapisz w miejscu na to przeznaczonym przy każdym zadaniu.
3. Pisz czytelnie. Używaj długopisu/pióra tylko z czarnym tuszem/atramentem.
4. Nie używaj korektora, a błędne zapisy wyraźnie przekreśl.
5. Pamiętaj, że zapisy w brudnopisie nie będą oceniane.
6. Wpisz zadeklarowane (wybrane) przez Ciebie na egzamin system operacyjny, program użytkowy oraz środowisko programistyczne.
7. Na tej stronie oraz na karcie odpowiedzi wpisz swój numer PESEL i przyklej naklejkę z kodem.
8. Nie wpisuj żadnych znaków w części przeznaczonej dla egzaminatora.



EINP-R1-**100**-2206

**Zadanie 1. Liczby nudne i ciekawe**

Rozważmy operację, która dodatniej liczbie całkowitej przyporządkowuje sumę kwadratów jej cyfr w zapisie dziesiętnym. Przykładowo: liczbie 123 zostanie przyporządkowana liczba 14, ponieważ  $1^2 + 2^2 + 3^2 = 14$ . Utwórzmy teraz ciąg, którego pierwszym elementem będzie dodatnia liczba całkowita  $n$ , a każdy kolejny jego element to wynik zastosowania powyższej operacji do elementu poprzedzającego go w tym ciągu.

Jeśli w otrzymanym w ten sposób ciągu pojawi się liczba 1, to początkową liczbę  $n$  nazywamy liczbą *nudną*, w przeciwnym razie  $n$  nazywamy liczbą *ciekawą*.

**Przykład:**

Dla  $n = 13$  otrzymujemy ciąg:

$$13, 10 = 1^2 + 3^2, 1 = 1^2 + 0^2$$

Tak więc 13 jest liczbą *nudną*.

Liczba 4 jest liczbą *ciekawą*, ponieważ:

$$4, 16 = 4^2, 37 = 1^2 + 6^2, 58 = 3^2 + 7^2, 89 = 5^2 + 8^2, 145 = 8^2 + 9^2, 42 = 1^2 + 4^2 + 5^2, 20 = 4^2 + 2^2, 4 = 2^2 + 0^2, 16, 37, 58, 89, 145, 42, 20, 4, \dots \text{ itd.}$$

Czyli nigdy nie otrzymamy liczby 1.

**Zadanie 1.1. (0–2)**

Uzupełnij tabelę – wpisz TAK, jeśli podana liczba jest *nudna*, albo NIE – jeśli nie jest *nudna*.

$n$	Czy <i>nudna</i> ?
4	Nie
229	Nie
82	Tak

Miejsce na obliczenia:

$2^2 + 2^2 + 9^2 = 81 + 8 = 89$        $3^2 + 4^2 = 58$   
 $8^2 + 9^2 = 64 + 81 = 145$        $5^2 + 6^2 = 89$   
 $1^2 + 4^2 + 5^2 = 42$        $8^2 + 2^2 = 64 + 4 = 68$   
 $4^2 + 2^2 = 20$        $6^2 + 8^2 = 100$   
 $2^2 + 0^2 = 4$        $1^2 + 0^2 + 0^2 = 1$       **NUDNA**  
 $4^2 = 16$   
 $1^2 + 6^2 = 37$

### Zadanie 1.2. (0–2)

W postaci pseudokodu lub w wybranym języku programowania napisz funkcję **SumaKwCyfr**( $n$ ), która dla dodatniej liczby całkowitej  $n$  oblicza sumę kwadratów jej cyfr.

#### Przykład:

Dla  $n = 123$  wynikiem **SumaKwCyfr**(123) jest liczba  $1^2 + 2^2 + 3^2 = 14$ .

**Uwaga:** W zapisie funkcji 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 dostępnych w językach programowania.

#### Specyfikacja:

Dane:

$n$  – dodatnia liczba całkowita

Wynik:

dodatnia liczba całkowita – suma kwadratów cyfr liczby  $n$  w zapisie dziesiętnym

Miejsce na zapis funkcji:

```
def SumaKwCyfr(n):  
    s = 0  
    while n > 0:  
        x = n % 10  
        n = n // 10      # dzielenie całkowite  
        s = s + x * x  
    return s
```

**Zadanie 1.3. (0–3)**

W postaci pseudokodu lub w wybranym języku programowania napisz algorytm, który dla danej dodatniej liczby całkowitej  $n < 1000$  sprawdza, czy liczba  $n$  jest *ciekawa* czy *nudna*.

**Uwaga:**

- w algorytmie możesz użyć funkcji **SumaKwCyfr**( $n$ ) z poprzedniego zadania
- możesz skorzystać z faktu, że suma kwadratów cyfr liczby trzycyfrowej jest nie większa niż  $9^2 + 9^2 + 9^2 = 243$

**Uwaga:** W zapisie możesz korzystać tylko z instrukcji sterujących, operatorów arytmetycznych: dodawania, odejmowania, mnożenia, dzielenia, dzielenia całkowitego i reszty z dzielenia; operatorów logicznych, porównań, odwoływania się do pojedynczych elementów tablicy i instrukcji przypisywania lub samodzielnie napisanych funkcji i procedur wykorzystujących powyższe operacje. **Zabronione** jest używanie funkcji wbudowanych dostępnych w językach programowania.

**Specyfikacja:**

Dane:

$n$  – dodatnia liczba całkowita mniejsza od 1 000

Wynik:

Prawda – gdy liczba jest *nudna*, albo Fałsz – gdy jest *ciekawa* (nie jest *nudna*)

Miejsce na zapis algorytmu:

czy - nudna ( $n$ ):

$k = 1$

poprzednie[1] =  $n$

dopóki  $n \neq 1$ :

$n = \text{SumaKwCyfr}(n)$

dla każdego  $i$  od 1 do  $k$ :

jeżeli poprzednie[ $i$ ] =  $n$ :

zwróć Fałsz

$k = k + 1$

zwróć Prawda

**Zadanie 2. Funkcja Koduj**

Dana jest funkcja **Koduj**( $n$ ), która dla zadanej dodatniej liczby całkowitej  $n$  oblicza pewien jej kod – słowo puste lub słowo zbudowane tylko z wielkich liter A lub B.

**Specyfikacja**

Dane:

$n$  – dodatnia liczba całkowita

Wynik:

Kod liczby  $n$  – słowo puste lub słowo zbudowane z wielkich liter A lub B

**Funkcja** Koduj( $n$ ):

**jeżeli**  $n = 1$

wynikiem jest `"`

**w przeciwnym wypadku**

$k \leftarrow n \text{ div } 2$

**jeżeli**  $k \bmod 2 = 0$

wynikiem jest `Koduj( $k$ ) + 'A'`

**w przeciwnym wypadku**

wynikiem jest `'B' + Koduj( $k$ )`

**Uwaga:**

- *div* jest operatorem oznaczającym część całkowitą z dzielenia
- *mod* jest operatorem oznaczającym resztę z dzielenia
- *słowem* nazywamy dowolny ciąg znaków
- `"` oznacza słowo puste (bez liter)
- `+` jest operatorem łączącym znak i słowo lub dwa słowa w jedno słowo.

**Zadanie 2.1. (0–2)**

Uzupełnij tabelę – wpisz wynik działania funkcji **Koduj**( $n$ ) dla podanych wartości  $n$ .

$n$	Wynik działania funkcji Koduj( $n$ )
1	<code>"</code>
2	<code>B</code>
12	<code>BBA</code>
33	<code>BAA A A</code>
158	<code>B B B B B A A</code>



### Zadanie 2.3. (0–2)

Podaj dwie różne dodatnie liczby całkowite, dla których funkcja **Koduj** da ten sam kod złożony z sześciu znaków.

Liczba 1: 68

Liczba 2: 66

Miejsce na obliczenia:

68  $\rightarrow$  100 0100

66  $\rightarrow$  100 0010

**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)**

Po dodaniu dwóch liczb  $101101_2$  i  $111011_2$  zapisanych w systemie binarnym otrzymamy:

1.	$1101000_2$	<input checked="" type="radio"/> P	<input type="radio"/> F
2.	$68_{16}$	<input checked="" type="radio"/> P	<input type="radio"/> F
3.	$140_8$	<input type="radio"/> P	<input checked="" type="radio"/> F
4.	$1120_4$	<input type="radio"/> P	<input checked="" type="radio"/> F

**Zadanie 3.2. (0–1)**

Poniżej przedstawiono opis dwóch tabel danych zawierających informacje o lekarzach i wizytach u tych lekarzy.

Pole *Id\_lekarza* w tabeli *Lekarze* jest połączone relacją „jeden do wielu” z polem *Id\_lekarza* w tabeli *Wizyty*.

*Lekarze*

Nazwa pola	Typ	Klucz
<i>Id_lekarza</i>	Tekst(5)	Klucz główny
<i>Imie_lekarza</i>	Tekst(50)	
<i>Nazwisko_lekarza</i>	Tekst(50)	
<i>Specjalnosc_lekarza</i>	Tekst(200)	

*Wizyty*

Nazwa pola	Typ	Klucz
<i>Id_lekarza</i>	Tekst(5)	Klucz obcy
<i>Id_pacjenta</i>	Tekst(10)	
<i>Data_wizyty</i>	Data	



1.	<p>Wynikiem zapytania:</p> <pre> SELECT Imie_lekarza, Nazwisko_lekarza, count(*) FROM Lekarze JOIN Wizyty ON Lekarze.Id_lekarza = Wizyty.Id_lekarza GROUP BY Wizyty.Id_lekarza; </pre> <p>jest zestawienie zawierające imię i nazwisko każdego lekarza oraz liczbę wizyt u tego lekarza.</p>	<b>P</b>	<b>F</b>
2.	<p>Wynikiem zapytania:</p> <pre> SELECT Specjalnosc_lekarza, count(*) FROM Lekarze JOIN Wizyty ON Lekarze.Id_lekarza = Wizyty.Id_lekarza GROUP BY Lekarze.Id_lekarza; </pre> <p>jest zestawienie zawierające nazwy specjalności oraz <u>łączne</u> liczby wizyt u lekarzy tych specjalności.</p>	<b>P</b>	<b>F</b>
3.	<p>Wynikiem zapytania:</p> <pre> SELECT Specjalnosc_lekarza, count(Specjalnosc_lekarza) FROM Lekarze JOIN Wizyty ON Lekarze.Id_lekarza = Wizyty.Id_lekarza GROUP BY Wizyty.Id_lekarza; </pre> <p>jest zestawienie zawierające nazwy specjalności oraz <u>łączne</u> liczby wizyt u lekarzy tych specjalności.</p>	<b>P</b>	<b>F</b>
4.	<p>Wynikiem zapytania:</p> <pre> SELECT Id_pacjenta, count(*) FROM Wizyty JOIN Lekarze ON Lekarze.Id_lekarza = Wizyty.Id_lekarza GROUP BY Lekarze.Id_lekarza; </pre> <p>jest zestawienie zawierające liczby wizyt pacjentów u poszczególnych lekarzy.</p>	<b>P</b>	<b>F</b>