

KOMISJA Arkusz zawiera informacje prawnie chronione EGZAMINACYJNA do momentu rozpoczęcia egzaminu.



WPISUJE ZDA	AJĄCY	
KOD	PESEL	miejsce na naklejkę
		na naklejkę

### EGZAMIN MATURALNY Z INFORMATYKI

# POZIOM ROZSZERZONY

# Część I

### PRZYKŁADOWY ARKUSZ EGZAMINACYJNY

DATA: 18 grudnia 2014 r.

CZAS PRACY: 60 minut

LICZBA PUNKTÓW DO UZYSKANIA: 15

WPISUJE ZDAJĄCY	WYBRANE:	
	(środowisko)	
	(kompilator)	
	(program użytkowy)	

### Instrukcja dla zdającego

- 1. Sprawdź, czy arkusz egzaminacyjny zawiera 10 stron. Ewentualny brak zgłoś przewodniczącemu zespołu nadzorującego egzamin.
- 2. Rozwiązania i odpowiedzi zamieść w miejscu na to przeznaczonym.
- 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 podlegają ocenie.
- 6. Wpisz zadeklarowane (wybrane) przez Ciebie na egzamin środowisko komputerowe, kompilator języka programowania oraz program użytkowy.
- 7. Jeżeli rozwiązaniem zadania lub jego części jest algorytm, to zapisz go w wybranej przez siebie notacji: listy kroków lub języka programowania, który wybrałaś/eś na egzamin.
- 8. Na karcie odpowiedzi wpisz swój numer PESEL i przyklej naklejkę z kodem.
- 9. Nie wpisuj żadnych znaków w części przeznaczonej dla egzaminatora.

### Zadanie 1. Liczby Armstronga (0–5)

Liczba całkowita złożona z n cyfr jest liczbą Armstronga (narcystyczną), jeżeli jest sumą swoich cyfr podniesionych do potegi n. Na przykład:  $153=1^3+5^3+3^3=1+125+27$ .

W tym zadaniu zajmiemy się przygotowaniem algorytmu sprawdzającego, czy dana liczba jest liczbą Armstronga.

#### Zadanie 1.1.

### Sprawdź, czy liczby 6, 407, 2278 są liczbami Armstronga.

Wpisz odpowiednio P, jeśli dana liczba jest liczbą Armstronga, albo F, jeśli nią nie jest.

	P/F
6	
407	
2278	

Miejsce na obliczenia.											

#### Zadanie 1.2.

W wybranej przez siebie notacji (lista kroków, wybrany język programowania) napisz algorytm:

- umieszczający poszczególne cyfry liczby k w tablicy tab\_cyfr[] w kolejności od najmniej do najbardziej znaczącej
- zwracający liczbę cyfr jej zapisu dziesiętnego.

### Specyfikacja:

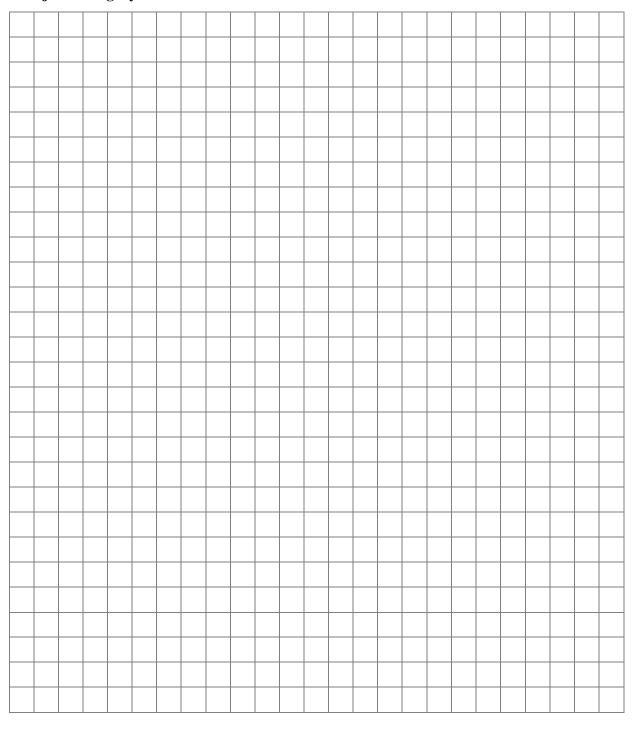
Dane:

k – liczba całkowita dodatnia.

Wynik:

n – liczba cyfr (całkowita dodatnia) w zapisie dziesiętnym liczby k,
tab\_cyfr[] – tablica zawierająca kolejne cyfry zapisu dziesiętnego liczby k,
w kolejności od najmniej znaczącej do najbardziej znaczącej.

## Przykład:



	Nr zadania	1.1.	1.2.
Wypełnia	Maks. liczba pkt.	1	2
egzaminator	Uzyskana liczba pkt.		

#### Zadanie 1.3.

W wybranej przez siebie notacji (lista kroków, wybrany przez Ciebie język programowania) zapisz algorytm sprawdzający, czy zapisana w tablicy  $tab\_cyfr[]$  liczba k jest liczbą narcystyczną. W swoim algorytmie załóż, że masz do dyspozycji zarówno funkcję wyliczającą n-tq potęgę liczby a, jak i zapisane w tablicy  $tab\_cyfr[]$ , kolejne cyfry zapisu dziesiętnego liczby k.

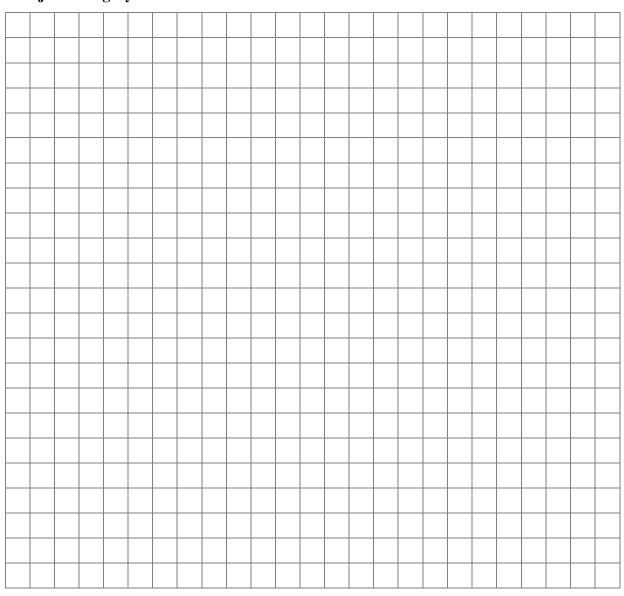
### Specyfikacja:

#### Dane:

k – liczba całkowita dodatnia,
n – liczba cyfr (całkowita dodatnia) w zapisie dziesiętnym liczby k,
tab\_cyfr[] – tablica zawierająca kolejne cyfry zapisu dziesiętnego liczby k,
w kolejności od najmniej znaczącej do najbardziej znaczącej,
potega (a, n) – funkcja zwracająca n-tą potęgę liczby a.

Wynik:

**PRAWDA** – jeśli dana liczba jest liczbą narcystyczną lub **FAŁSZ** – w przeciwnym wypadku.



### Zadanie 2. Oceń prawdziwość poniższych zdań (0-4)

Wpisz odpowiednio P, jeśli zdanie jest prawdziwe, albo F, jeśli zdanie jest fałszywe.

### Zadanie 2.1.

Niech  $a=(1001001)_2$ ,  $b=(211)_9$ ,  $c=(211)_8$ , wówczas:

	P/F
b > c	
a + b - c = 0	
$c = (89)_{16}$	

### Zadanie 2.2.

### W sieciach komputerowych:

	P/F
192.168.0.1 jest adresem pętli zwrotnej.	
w klasie adresowej A mamy 27 adresów sieci i 224 adresy hostów.	
adresy 94.254.99.1/16 oraz 94.254.168.168/16 należą do jednej podsieci.	

### Zadanie 2.3.

### Protokołami służącymi do pobierania wiadomości elektronicznych z serwera są:

	P/F
IMAP	
SMTP	
POP3	
SNMP	

### Zadanie 2.4.

### Licencja na oprogramowanie GNU GPL:

	P/F
dopuszcza wprowadzanie własnych poprawek.	
wymusza wyświetlanie reklam w czasie pracy.	
stosowana jest wyłącznie przy tworzeniu programów prototypowych, mogących działać niestabilnie.	
nie zezwala na użytkowanie zarobkowe.	

Wypełnia egzaminator	Nr zadania	1.3.	2.1.	2.2.	2.3.	2.4.
	Maks. liczba pkt.	2	1	1	1	1
	Uzyskana liczba pkt.					

#### Zadanie 3. Kodowanie (0–6)

Domyślnie znak kodowany jest na 8 bitach, czyli na 1 bajcie. W ten sposób można zakodować 255 różnych znaków kodami większymi od 0. W praktyce często zdarza się, że różnych znaków w tekście jest mniej niż 255 – wtedy można przypisać do kolejnych różnych znaków kolejne liczby zapisane w systemie binarnym. Liczba wykorzystanych bitów zależy od maksymalnej liczby kodowanych znaków.

#### Przykład:

Tekst źródłowy: HANIA standardowo zajmie 5 bajtów w pamięci (1 bajt na znak):

H - 00000001; A - 00000010; N - 00000011; I - 00000100.

Ponieważ tekst zawiera tylko 4 różne znaki, do ich zakodowania kodami większymi od 0 wystarczą 3 bity, na przykład: H-001, A-010, N-011, I-100

Tak zakodowany tekst zajmuje niepełne 2 bajty. Ostatni wolny bit uzupełnimy zerem.

0	0	1	0	1	0	0	1	1	1	0	0	0	1	0	0
	Н			A			N		I		A			Wolny bit	
pierwszy bajt: 101001 <sub>bin</sub> = 41 <sub>dec</sub>								drugi bajt: 11000100 <sub>bin</sub> = 196 <sub>dec</sub>							

Tekst skompresowany będzie zawierać dwa bajty o wartościach liczbowych: 41 i 196 w reprezentacji dziesiętnej.

#### Zadanie 3.1.

Zdekoduj tekst ukryty w dwóch kolejnych bajtach o wartościach dziesiętnych 110 i 64.

Tekst zawiera kombinację trzech różnych znaków, każdy znak zakodowany na 2 bitach:  $K-01,\,A-10,\,J-11$ 

Uzupełnij pierwszy i drugi wiersz w poniższej tabeli.

Poniżej wpisz tekst zdekodowai	ıy.	

Miejsce na obliczenia.		

### Zadanie 3.2.

Zapisz algorytm (w postaci listy kroków lub w wybranym języku programowania), który dla danego łańcucha znaków zwraca liczbę różnych znaków.

### Specyfikacja:

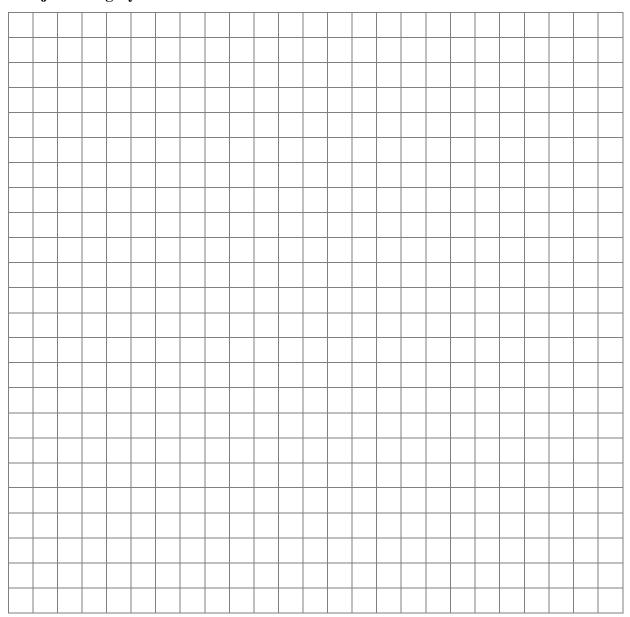
Dane:

s – źródłowy łańcuch znaków

Wynik:

r-liczba całkowita określająca liczbę różnych znaków w tekście  $\boldsymbol{s},$ 

założenie: r < 100



	Nr zadania	3.1.	3.2.
Wypełnia	Maks. liczba pkt.	1	2
egzaminator	Uzyskana liczba pkt.		

#### Zadanie 3.3.

Zapisz algorytm dekodowania tekstu (w postaci listy kroków lub w wybranym języku programowania), który pobiera tablicę bajtów tekstu skompresowanego i wyświetla źródłowy tekst.

W algorytmie możesz wykorzystać poniższe funkcje lub ich odpowiedniki w wybranym języku programowania:

testBit (bajt, numerBitu) – zwraca wartość TRUE, jeśli w bajcie bit o podanym numerze ma wartość 1, lub FALSE, jeśli ten bit ma wartość 0.

ustawBit (bajt, numerBitu) – zwraca bajt, w którym bit o podanym numerze ustawiono na 1, a pozostałe bity nie zostały zmienione.

### Specyfikacja:

#### Dane:

v [] – tablica zawierająca kolejne bajty tekstu skompresowanego

n – liczba elementów tablicy v []

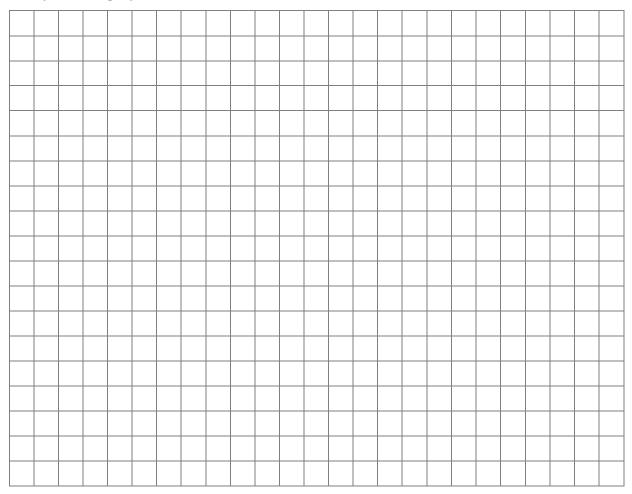
r – liczba różnych znaków w tekście

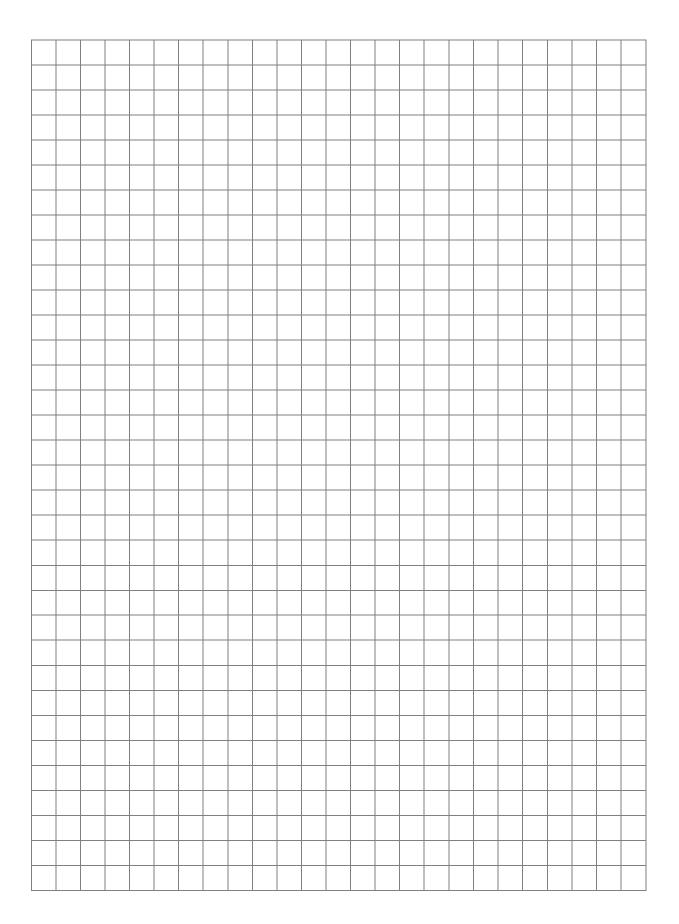
t [] – tablica przechowująca pary {znak, kod}, definiująca przyporządkowanie kodu do znaku

bity – liczba bitów przeznaczonych do przechowywania kodu jednego znaku

### Wynik:

s – źródłowy łańcuch znaków





Wypełnia egzaminator	Nr zadania	3.3.
	Maks. liczba pkt.	3
	Uzyskana liczba pkt.	

# BRUDNOPIS (nie podlega ocenie)