Organizatorzy: Uniwersytet Mikołaja Kopernika w Toruniu Wydział Matematyki i Informatyki
Oddział Kujawsko-Pomorski Polskiego Towarzystwa Informatycznego
Centrum Kształcenia Ustawicznego TODMiDN w Toruniu

PRÓBNY EGZAMIN MATURALNY Z INFORMATYKI STYCZEŃ 2021

Arkusz I

Czas pracy: **60 minut**

Liczba punktów do uzyskania: **15**

Instrukcja dla zdającego

1. Sprawdź, czy arkusz egzaminacyjny zawiera 11 stron (zadania 1 – 3). 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 poniżej 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, pseudokodu lub języka programowania, który wybrałaś/eś na egzamin

Dane uzupełnia uczeń:

WYBRANE: ……………………………………………

(środowisko)

……………………………………………

(kompilator)

……………………………………………

(program użytkowy)

PESEL:

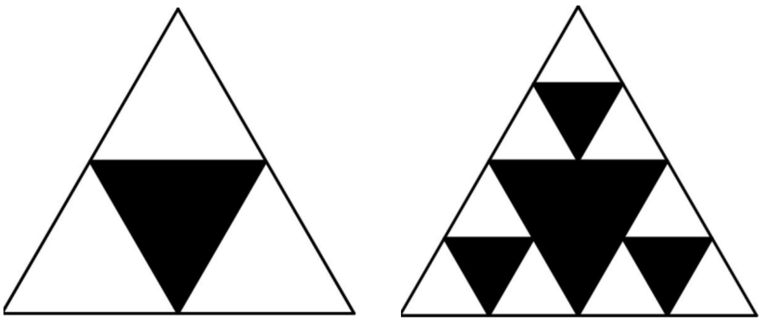
Klasa:

Zadanie 1. Test (0-5)

Oceń, czy poniższe zdania są prawdziwe. Zaznacz P, jeśli zdanie jest prawdziwe, albo F, jeśli zdanie jest fałszywe. W każdym zadaniu uzyskasz punkt, jeśli poprawnie odpowiesz na wszystkie jego części.

Zadanie 1.1. (0–1)

Poniżej jest pokazany Trójkąt Sierpińskiego stopnia 1 i otrzymany z niego Trójkąt Sierpińskiego stopnia 2. Ten drugi trójkąt ilustruje, jak powstają kolejne Trójkąty Sierpińskiego – w każdy biały trójkąt jest wpisywany odpowiednio zmniejszony trójkąt nr 1.



Ile białych i czarnych trójkątów zawiera Trójkąt Sierpińskiego stopnia *k*?

$3k$ (białe) + $3k$ (czarne)	P	F
3^2k (białe) + $1+3k$ (czarne)	P	F
3^k (białe) + $(3^k–1)/2$ (czarne)	P	F
3^k (białe) + $1+3k$ (czarne)	P	F

Zadanie 1.2. (0–1)

Ciąg liczb 10, 0, 3, 7, 8, 5, 4 poddano sortowaniu przez proste wybieranie w porządku niemalejącym. Poniżej widoczne są postaci tego ciągu po dwóch kolejnych etapach tego sortowania.

0, 10, 3, 7, 8, 5, 4

0, 3, 10, 7, 8, 5, 4

Wskaż kolejne wyrazy tego ciągu po trzecim etapie:

0, 3, 7, 10, 8, 5, 4	P	F
0, 3, 4, 7, 8, 5, 10	P	F
0, 3, 4, 10, 7, 8, 5	P	F
0, 3, 4, 5, 10, 7, 8	P	F

Zadanie 1.3. (0–1)

Wskaż zdania prawdziwe i zdania fałszywe. Wyrażenie ONP zapisane w postaci (argumentami w tych wyrażeniach są pojedyncze cyfry):

3 2 + 1 * ma wartość 5	P	F
2 4 + 3 * 1 + odpowiada zapisowi (2 + 4) * 3 + 1	P	F
2 1 + 3 1 + * odpowiada zapisowi 2 + 1 + 3 * 1	P	F
2 2 * 2 + ma wartość 8	P	F

Zadanie 1.4. (0–1)

Adres MAC jest 48-bitową liczbą zapisywaną heksadecymalnie (szesnastkowo). Czasami można się spotkać z określeniem, że adres MAC jest 6-bajtowy, ponieważ 1 bajt to 8 bitów, więc 6 bajtów odpowiada 48 bitom. Pierwsze 24 bity liczby oznaczają producenta karty sieciowej, pozostałe 24 bity są unikatowym identyfikatorem danego egzemplarza karty. Na przykład adres **00:0A:E6:3E:FD:E1** oznacza, że karta została wyprodukowana przez Elitegroup Computer System Co. (definicja za wikipedia.pl)

Który z przedstawionych poniżej adresów MAC w postaci 48 bitowych słów wskazuje, że pochodzi z Elitegroup Computer System Co (zgodnie z przedstawioną definicją):

00000000 00001010 11100110 0011 1110 11111101 11100001	P	F
00000000 00001010 11100111 0011 1110 11111101 11100101	P	F
10000000 00001010 11100110 0011 1110 11111101 11100001	P	F
00000000 00001010 11100110 1111 1111 11111111 11111111	P	F

Zadanie 1.5. (0–1)

Ponieważ kod ASCII jest 7-bitowy, a większość komputerów operuje na 8-bitowych bajtach, dodatkowy bit można wykorzystać na powiększenie zbioru kodowanych znaków do 256 symboli. Załóżmy, że nie mamy do dyspozycji kodów ASCII. Co najmniej ile bitowe kody są potrzebne do zakodowania pięciu symboli różnymi kodami jednakowej długości?

4	P	F
3	P	F
2	P	F
1	P	F

Wypełnia egzaminator	**Numer zadania**	**1.1**	**1.2**	**1.3**	**1.4**	**1.5**	**Suma**
	Maksymalna liczba punktów	**1**	**1**	**1**	**1**	**1**	**5**
	Uzyskana liczba punktów						

Zadanie 2. Gra (0-5)

Jaś i Małgosia postanowili zagrać w grę o liczbach. Zasady gry są proste:

- na stole znajduje się 10 różnych żetonów z wartościami od 1 do 10;
- gracze naprzemiennie wykonują po jednym ruchu: zabranie wybranego żetonu do swojej puli;
- celem gry jest takie dobieranie żetonów do swojej puli, żeby nie przekroczyć sumy punktów równej 25;
- gracz, który w dowolnym momencie uzyska sumę punktów większą od 25, natychmiast przegrywa i gra się kończy.

Ponieważ Jaś i Małgosia są początkującymi graczami, postanowili najpierw przyjrzeć się następującym strategiom gry:

- Strategia A: zawsze wybieraj żeton o maksymalnej wartości.
- Strategia B: zawsze wybieraj żeton o minimalnej wartości.

Przykład: Jaś i Małgosia używają strategii A. Jaś zaczyna jako pierwszy i bierze żeton o wartości 10. Następnie Małgosia zabiera ten o wartości 9. Teraz Jaś bierze 8, a następnie Małgosia bierze 7. Ponownie Jaś bierze 6, a Małgosia 5. Teraz, zgodnie ze strategią, Jaś bierze żeton o wartości 4, tym samym przegrywając grę, ponieważ suma jego punktów wynosi 28 (10+8+6+4).

Zadanie 2.1. (0–1)

Uzupełnij tabelkę. Jaś zawsze wykonuje pierwszy ruch. Punkty zliczamy w momencie, w którym jeden z graczy przegrywa i gra się kończy.

Strategia Jasia	Strategia Małgosi	Punkty Jasia	Punkty Małgosi	Zwycięzca
A	A	28 (10+8+6+4)	21 (9+7+5)	Małgosia
B	B			
A	B			
B	A			

Miejsce na obliczenia.

Po przyjrzeniu się efektom strategii Jaś i Małgosia zauważyli, że dla gracza rozpoczynającego rozgrywkę tylko jedna ze strategii może przynieść mu wygraną.

Zadanie 2.2. (0–1)

W wybranej przez siebie notacji (pseudokod, lista kroków, kod w wybranym języku programowania) napisz dwie funkcje: **stratA(zetony, n)** oraz **stratB(zetony, n)**, które dla podanej tablicy aktualnie pozostałych na planszy żetonów oraz ich ilości **n** zwrócą wartość kolejnego żetonu, który należy zabrać zgodnie ze strategią A dla funkcji **stratA**, lub zgodnie ze strategią B dla funkcji **stratB**.

Specyfikacja:

Dane: zetony – tablica liczb całkowitych zawierająca wartości żetonów aktualnie pozostałych na planszy

n – liczba żetonów w tablicy

Wynik: z – wartość kolejnego żetonu do zabrania, zgodnie ze strategią

Algorytm

Zadanie 2.3. (0–3)

Wykorzystując napisane w punkcie 2.2 funkcje napisz program, w wybranej przez siebie notacji (pseudokod, lista kroków, kod w wybranym języku programowania), który dla podanych przyjętych strategii obu graczy obliczy, który z nich jest zwycięzcą.

Uwaga:

W programie możesz użyć funkcji **usun(tab, w)**, która z tablicy **tab** usuwa element o wartości **w.**

Specyfikacja:

Dane: Strategia Jasia: znak ‘A’ lub znak ‘B’

Strategia Małgosi: znak ‘A’ lub znak ‘B’

Wynik: Zwycięzca: tekst „Jaś” lub „Małgosia”

Algorytm

Wypełnia egzaminator	Numer zadania	2.1	2.2	2.3
	Maksymalna liczba punktów	1	1	3
	Uzyskana liczba punktów			

Zadanie 3. Tablica (0-6)

Dana jest tablica o N wierszach i M kolumnach liczb całkowitych $a_{i,j}$, gdzie i oznacza numer wiersza, zaś j numer kolumny. Każdy z N wierszy tablicy jest posortowany niemalejąco, tzn. dla dowolnego i

$$a_{i,1} \leq a_{i,2} \leq \ldots \leq a_{i,M}.$$

Zadanie 3.1. (0–1)

Uzupełnij specyfikację przedstawionego poniżej algorytmu.

Dane:

N, M – liczby całkowite, dodatnie
$A = [a_{i,j}]$ – tablica $N{\cdot}M$ liczb całkowitych, nieujemnych o N wierszach i M kolumnach

Wynik:

x – ……………………………………………

Algorytm:

Krok 1. Rozpocznij algorytm.
Krok 2. x ← 0
Krok 3. Dla i ← 1, 2, …, N wykonuj Kroki 4-6:
  Krok 4. Jeśli $a_{i,1}$ **mod** 2 = 0 to wykonaj Kroki 5-6:
    Krok 5. Dla j ← 1, 2, …, M wykonuj Krok 6:
      Krok 6. x ← x + 1
Krok 7. x ← x **div** M
Krok 8. Wypisz x jako wynik i zakończ algorytm.

Uwaga: W tym algorytmie **div** oznacza dzielenie całkowite, **mod** oznacza resztę z dzielenia.

Miejsce na obliczenia

Zadanie 3.2. (0–2)

Złożoność opisanego powyżej algorytmu to $O(NM)$. Zapisz algorytm, w wybranej przez siebie notacji, zgodny z tą samą specyfikacją, lecz działający w czasie $O(N)$.

Algorytm

Zadanie 3.3. (0–3)

Załóżmy dodatkowo, że liczby w każdym wierszu są parami różne. Zaproponuj algorytm, działający w czasie $O(N \log M)$, rozwiązujący problem zadany poniższą specyfikacją. Za algorytm rozwiązujący poniższy problem, lecz o większej złożoności, otrzymasz mniejszą liczbę punktów.

Dane:

N, M – liczby całkowite, dodatnie
$A=[a_{i,j}]$ – tablica $N \cdot M$ liczb całkowitych, nieujemnych o *N* wierszach i *M* kolumnach o posortowanych rosnąco wierszach
x – liczba całkowita, nieujemna

Wynik:

L – liczba wystąpień *x* w tablicy *A*

Algorytm

Wypełnia egzaminator	Numer zadania	3.1	3.2	3.3
	Maksymalna liczba punktów	1	2	3
	Uzyskana liczba punktów			

BRUDNOPIS ***(nie podlega ocenie)***