



	UZUPEŁNIA ZDAJĄCY	
KOD	PESEL	miejsce na naklejkę

EGZAMIN MATURALNY Z INFORMATYKI

POZIOM ROZSZERZONY

Część I



MIN-R1_1P-183

DATA: **7 czerwca 2018 r.**GODZINA ROZPOCZĘCIA: **14:00**

CZAS PRACY: 60 minut

LICZBA PUNKTÓW DO UZYSKANIA: 15

UZUPEŁNIA ZDAJĄCY	WYBRANE:	
	(system operacyjny)	
	(program użytkowy)	
	(środowisko programistyczne)	

Instrukcja dla zdającego

- 1. Sprawdź, czy arkusz egzaminacyjny zawiera 9 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 zadeklarowany (wybrany) przez Ciebie na egzamin system operacyjny, program użytkowy oraz środowisko programistyczne.
- 7. Jeżeli rozwiązaniem zadania lub jego części jest algorytm, to zapisz go w notacji wybranej przez siebie: listy kroków, pseudokodu lub języka programowania, który wybierasz na egzamin.
- 8. Na tej stronie oraz na karcie odpowiedzi wpisz swój numer PESEL i przyklej naklejkę z kodem.
- 9. Nie wpisuj żadnych znaków w części przeznaczonej dla egzaminatora.





Zadanie 1. Wyszukiwanie liczb (0–6)

Funkcja F(T, x) przyjmuje jako argumenty tablicę T, w której znajdują się liczby całkowite uporządkowane niemalejąco, oraz liczbę całkowitą x, którą ma wyszukać w tablicy.

Dane:

```
n — liczba elementów tablicy, n > 0 T[1..n] — n-elementowa tablica zawierająca liczby całkowite uporządkowane niemalejąco x — liczba całkowita poszukiwana w tablicy T
```

Wynik:

prawda – jeśli liczba x występuje w tablicy T, fałsz – w przeciwnym razie

```
\begin{array}{c} \textbf{funkcja} \ \ F(\texttt{T}, \texttt{x}) \\ p \leftarrow 1 \\ k \leftarrow n \\ \textbf{dop\acute{o}ki} \ p <= k \ \textbf{powtarzaj:} \\ s \leftarrow (p+k) \ \textbf{div} \ 2 \\ \textbf{je\acute{s}li} \ T[\texttt{s}] = \texttt{x} \ \textbf{to} \\ \textbf{wynikiem jest prawda} \\ \textbf{zako\acute{n}cz działanie funkcji} \\ \textbf{w przeciwnym razie} \\ \textbf{je\acute{s}li} \ T[\texttt{s}] < \texttt{x} \ \textbf{to} \ p \leftarrow \texttt{s}+1 \\ \textbf{w przeciwnym razie} \ k \leftarrow \texttt{s}-1 \\ \textbf{wynikiem jest } falsz \end{array}
```

Uwaga: zapis div oznacza dzielenie całkowite

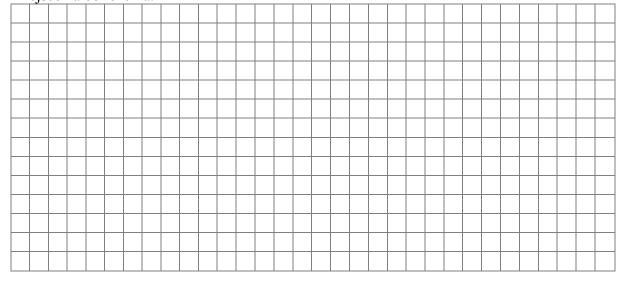
Zadanie 1.1. (0-4)

Rozważmy tablicę T = [3; 5; 7; 8; 9; 13; 33; 37; 40; 43].

A. Podaj wynik funkcji F(T, x) dla liczby x=7.

Odpowiedź:

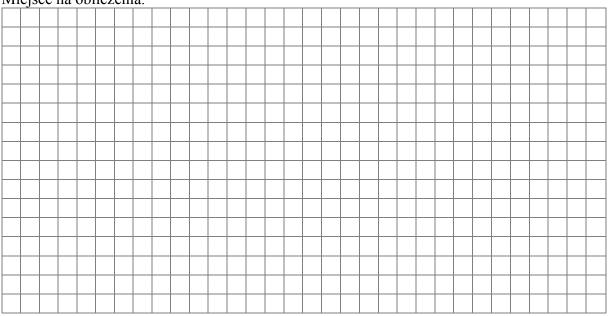
Miejsce na obliczenia.



B. Podaj, ile razy nastąpi modyfikacja wartości zmiennej p, a ile razy zmiennej k podczas wykonywania pętli **dopóki** dla x=7 oraz dla x=43,

Zmienna	Ile razy nastąpi modyfikacja wartości zmiennej?	
p k		

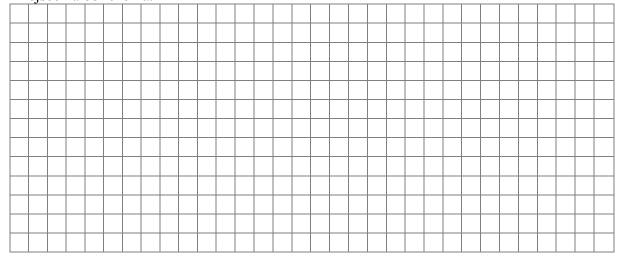
Miejsce na obliczenia.



C. Podaj kolejne wartości zmiennej s, jakie będzie ona przyjmowała dla x = 7.

Odpowiedź:

Miejsce na obliczenia.

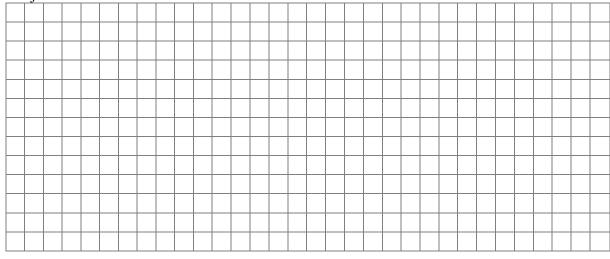


Zadanie 1.2. (0-1)

Podaj, ile razy dla n=100 jest spełniony warunek "p <= k" podczas wykonywania pętli **dopóki** w funkcji F(T, x), w sytuacji, gdy poszukiwana liczba jest większa od każdego z elementów zapisanych w tablicy T?

Odpowiedź:

Miejsce na obliczenia.



Zadanie 1.3. (0-1)

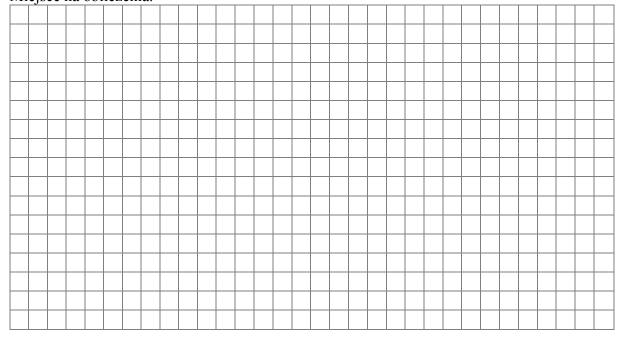
Sprawdź działanie funkcji F(T, x) dla nieuporządkowanej tablicy

$$T = [3; 5; 7; 8; 90; 13; 33; 37; 40; 43].$$

Podaj wynik działania tej funkcji dla tablicy T oraz liczby x = 43.

Odpowiedź:

Miejsce na obliczenia.



Zadanie 2. Liczby Fibonacciego (0-5)

Liczby Fibonacciego są definiowane w następujący sposób:

$$F_1 = 1, F_2 = 1,$$

$$F_n = F_{n-1} + F_{n-2} dla n = 3, 4, ...$$

Rekurencyjny algorytm, który służy do obliczania wartości F_n dla dowolnego $n \ge 1$, można zapisać następująco:

Zadanie 2.1. (0-2)

Zapisz w wybranej przez siebie notacji (w języku programowania lub w pseudokodzie) algorytm iteracyjny, który służy do obliczania wartości liczby F_n dla dowolnego $n \ge 1$. Algorytm nie może używać tablic.

Algorytm.

Zadanie 2.2. (0-1)

Aby obliczyć F_{45} , wywołano najpierw funkcję iteracyjną, a potem – rekurencyjną. Okazało się, że czas trwania obliczeń realizowanych przez funkcję rekurencyjną był długi, podczas gdy funkcja iteracyjna prawie natychmiast podała wynik. Uzasadnij długi czas działania funkcji **rekurencyjnej**.

Odpowiedź:	 	

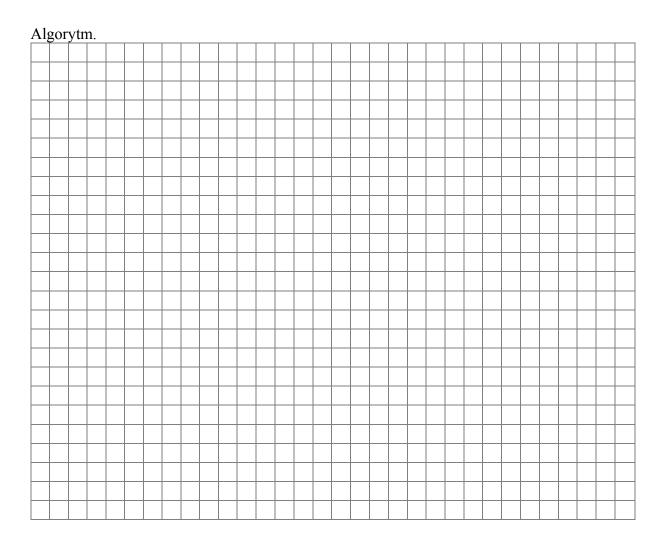
Zadanie 2.3. (0-2)

Aby przyśpieszyć rekurencyjne obliczanie wartości n-tego wyrazu ciągu Fibonacciego, można skorzystać z następujących wzorów, prawdziwych dla dowolnego całkowitego $k \ge 2$:

$$F_{2k}$$
= (F_{k+1})² - (F_{k-1})²

$$F_{2k-1} = (F_k)^2 + (F_{k-1})^2$$

Zapisz w wybranej przez siebie notacji (w postaci listy kroków, w języku programowania lub w pseudokodzie) algorytm **rekurencyjny**, który służy do obliczania wartości liczby F_n dla dowolnego $n \ge 1$ i korzysta z tych wzorów.



Zadanie 3. Test (0-4)

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)

W tabeli *T* zapisano wiele rekordów danych zawierających informacje o zawodnikach. Pola rekordu to: id, nazwisko, imie, plec, wzrost, numer_startowy, punkty, id klubu.

Polecenie SQL obliczające sumę punktów zawodników z klubu o id_klubu równym liczbie 100, może mieć postać:

1	select sum(punkty) as suma	P	F
	from T where id_klubu=100;		
,	select avg(punkty)	D	IF
_ Z	from T where id=100;	Г	Г
2	select punkty as suma	D	IF
3	from T where id_klubu=100;	r	Г
	select sum(punkty)	D	IF
	from T where id_klubu=100;	r	Г

Zadanie 3.2. (0–1)

Które zdania dotyczące struktury danych zwanej stosem są prawdziwe?

1	Elementy stosu są zdejmowane w odwrotnej kolejności niż kolejność ich wkładania na stos.	P	F
2	Tylko pierwszy dodany element jest zawsze dostępny na stosie.	P	F
3	Stos może być używany m.in. przy obliczaniu wartości wyrażeń zapisanych w Odwrotnej Notacji Polskiej (ONP).	P	F
4	Tylko ostatnio dodany element jest zawsze dostępny na stosie.	P	F

Zadanie 3.3. (0–1)

Do jednoznacznego zakodowania znaków pięcioelementowego alfabetu wystarczą/y:

1	2 bity.	P	F
2	3 bity.	P	F
3	5 bitów.	P	F
4	8 bitów.	P	F

Zadanie 3.4. (0–1)

Dana jest funkcja rekurencyjna *Rek*, której argumentem jest nieujemna liczba całkowita *n*.

funkcja Rek(n)

jeśli (n>0) to wykonaj kolejno dwie instrukcje:

- 1. wywołaj Rek dla argumentu n-1
- 2. wypisz n

Jeśli wywołamy ją dla *n* równego 5, to:

1	Zero będzie wypisane.	P	F
2	Największą wypisaną liczbą będzie 5.	P	F
3	Zostanie wypisanych 5 liczb.	P	F
4	Liczby zostaną wypisane w kolejności malejącej.	P	F

BRUDNOPIS (nie podlega ocenie)