POMORSKA LIGA ZADANIOWA *ZDOLNI Z POMORZA*

Konkurs dla uczniów dla uczniów klas VII i VIII szkoły podstawowej województwa pomorskiego w roku szkolnym 2020/2021

Etap II – powiatowy

Przedmiot: Informatyka

Przed przystąpieniem do rozwiązywania zadań zapoznaj się z instrukcją.

INSTRUKCJA

1. Oprócz arkusza z treścią zadań otrzymujesz załączniki niezbędne do rozwiązania niektórych zadań. Ich nazwy są określone w treści zadań. Łącznie są 2 (jeden do zadania pierwszego oraz jeden do zadania drugiego). Przed przystąpieniem do rozwiązywania sprawdź, czy na pewno pobrała(e)ś te wszystkie pliki. Nie wolno używać własnych plików zamiast tych załączników wyraźnie wymienionych w treści zadania.
2. Zwróć uwagę, aby pliki zawierające rozwiązania oraz pliki z danymi testowymi (takie pliki będziesz tworzyć samodzielnie rozwiązując zadania 3,4 oraz 5) miały zawartość i nazwy takie jakie określono w treściach zadań.
3. Nie przesyłaj do oceny innych plików niż te określone w treści zadań.
4. Pliki z rozwiązaniami przesyłasz organizatorom Pomorskiej Ligi Zadaniowej zgodnie z odrębną instrukcją.
5. Przy rozwiązywaniu zadań powinno się wykorzystywać te środowiska i narzędzia programistyczne, którymi posługujesz się w szkole lub w domu. W szczególności dopuszcza się następujące środowiska:
   a) systemy operacyjne – zarówno z grupy Windows, jak i dystrybucje systemu Linux
   b) pakiety oprogramowania biurowego- Microsoft Office, ale również wersje otwarte np. Libra Office
   c) języki programowania – C/C++,C#, Free Pascal, Java, Python (kompilatory adekwatne do używanych środowisk systemu operacyjnego np. DEV, Code Block, Eclipse, GCC,G++ itp.)

d) wizualne środowiska programowania – Scratch, Logomocja lub inne mutacje LOGO.

e) nie określa się szczegółowo numerów wersji używanego oprogramowania, aby uczeń mógł je elastycznie dostosować do używanych w szkole, ale w przypadku języków programowania prosimy o dokładne podanie (np. w odrębnym pliku tekstowym) jaka wersja kompilatora (względnie jakie środowisko programistyczne) było wykorzystywane, aby adekwatnego użyć przy ocenie pracy z zastrzeżeniem punktu 6a.

6. W przypadku rozwiązań związanych z używaniem języków programowania:
   a) powinno się używać kompilatorów bez ograniczonej dostępności (np. związanej z ich komercyjnym charakterem),
   b) dopuszcza się używanie wyłącznie standardowej biblioteki (bibliotek) języka, nie jest dozwolone dołączanie zewnętrznych bibliotek np. crt, graph (poza sytuacjami wynikłymi z treści zadania np. próba realizacji rysunku wynikającego z treści zadania),
   c) nie jest dopuszczalne otwieranie przez program innych programów, plików (poza tymi z danymi wejściowymi i wyjściowymi), ani tworzenie nowych plików (np. tymczasowych) oraz tworzenie innych procesów lub wątków,
   d) błąd kompilacji przy sprawdzaniu jest traktowany jako błąd składni i sprawdzający nie ma obowiązku dalszej analizy takiego rozwiązania choć może uwzględnić poprawny zarys samego algorytmu przyznając znacząco mniejszą liczbę punktów. Podobna uwaga dotyczy pojawienia się nietypowych błędów wykonania w trakcie uruchamiania programów (np. naruszenie zasad ochrony pamięci),
   e) rozwiązania nie powinny wykorzystywać plików nagłówkowych typowych dla środowisk DOS/Windows np. conio.h lub windows.h (dotyczy języka C++),
   f) rozwiązania nie powinny naruszać bezpieczeństwa systemowego w środowisku, w którym są sprawdzane.
7. Programy nie powinny zajmować się testowaniem poprawności danych. zakłada się, że ma ona miejsce.
8. Proszę zwrócić uwagę na samodzielność rozwiązań.

Życzymy powodzenia!

Zadanie 1

W załączonym pliku ***Zalacznik-Zadanie1-psy.txt*** umieszczono dane dotyczące 1000 psów, uczestników pewnej wystawy. W kolejnych kolumnach znajdują się następujące dane każdego psa: *Lp, Rasa, Wiek, Płeć oraz Liczba medali*. Dane w każdym wierszu są oddzielone od siebie znakiem tabulacji.

Rozwiąż następujące problemy:

a) podaj numery porządkowe (LP) oraz rasę 3 psów, które zdobyły najwięcej medali,

b) oblicz jaki procent wszystkich psów stanowią owczarki (Uwaga ! Owczarki są różne np. kaukaski albo kataloński, bierzemy je wszystkie pod uwagę). Wynik podaj z dokładnością do jednego miejsca po przecinku,

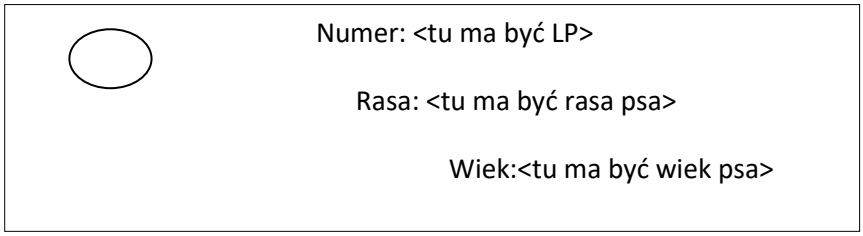
c) oblicz ile mamy samic, a ile samców,

d) oblicz średni wiek: wszystkich psów, tylko samców oraz tylko samic. Wyniki podaj z dokładnością do dwóch miejsc po przecinku,

e) pewien sponsor ufundował nagrodę dla każdego psa. Wysokość tej nagrody obliczamy wg następującej formuły: dla psów mających więcej niż 2 lata wypłacamy po 50 zł za każdy medal i dodatkowo 25 zł za każdy medal powyżej czwartego, zaś dla psów mających co najwyżej dwa lata wypłacamy po 40 zł za każdy medal. Dodatkowo oblicz sumę wszystkich nagród sponsora wypłaconych psom,

f) przedstaw na wykresie ile mamy psów, które zdobyły co najwyżej 1 medal, ile takich, które zdobyły 2 lub 3 medale, a ile tych, które zdobyły więcej niż 3 medale,

g) przygotuj szablon następującej etykiety-identyfikatora wystawowego przygotowanego przez organizatorów dla każdego psa:



W miejscu gdzie jest narysowany owal umieść zdjęcie dowolnego psa (nie musi ono naturalnie być dopasowane do rasy, chodzi o przykładowe, takie samo zdjęcie, które ma być umieszczone na identyfikatorze każdego psa). Następnie w oparciu o ten szablon utwórz 3 identyfikatory dla psów o numerach 15, 45 oraz 123.

Do oceny oddajesz plik zawierający komputerową realizację konstrukcji oraz obliczeń, na podstawie których uzyskasz rozwiązanie zadania. Nazwa tego pliku to *Zadanie1*. Jeśli tworzysz więcej plików z realizacją konstrukcji i obliczeń to nazwij je *Zadanie1a, Zadanie1b* itd. Dodatkowo wyniki dla punktów od a) do e) umieść w pliku *Zadanie1-wyniki* (w przypadku punktu e należy zapisać w tym pliku tylko sumę nagród) wyraźnie zaznaczając, , które wyniki, których pytań dotyczą. Z kolei w pliku *Zadanie1-etykiety* umieść trzy wykonane identyfikatory, o których mowa w punkcie g), a w pliku *Zadanie1-szablon* konstrukcję na podstawie, którego można utworzyć dowolne inne etykiety.
Uwaga ! Odpowiedzi liczbowe umieszczone w pliku tekstowym *Zadanie1-wyniki* oraz etykiety z pliku *Zadanie1-etykiety* nie będą mogły być uznane nawet jeśli będą poprawne, o ile nie znajdą potwierdzenia i odzwierciedlenia w zawartości pliku z komputerową realizacją obliczeń oraz w pliku *Zadanie1-szablon*.

11 punktów

Zadanie 2

W załączonym pliku *Zalacznik-Zadanie2-zmiana_warty.jpg*[1] znajdziesz zdjęcie, które było motywem przewodnim konkursu graficznego dla uczniów pewnej szkoły. Konkurs polegał na tym, żeby nanieść w tym obrazie jak najwięcej korekt (retuszy) zdjęcia oryginalnego w taki sposób, aby jury nie mogło ich łatwo odnaleźć. Przyjęto, że jeżeli jury nie odnajdzie poprawki

[1] Zdjęcie pochodzi z prywatnych zbiorów autora zadania- nie ma obawy o naruszenie praw autorskich.

w ciągu 5 minut to korekta jest uważana za zaliczoną. Wygrywało zdjęcie, na którym było najwięcej zaliczonych korekt.

Twoje zadanie polega na tym, aby utworzyć 4 kopie zdjęcia, które wygrały w konkursie z odpowiednią liczbą drobnych korekt. Na tym, którego autorka zwyciężyła w konkursie były 4 zaliczone korekty, na dwóch kolejnych 3 zaliczone korekty, a na ostatnim z tych czterech 2 zaliczone korekty.

Korekty możesz wymyśleć wg własnego uznania, ale należy kierować się następującymi regułami:

a) korekty powinny być subtelnie ukryte (bo w konkursie chodziło przecież, aby ich łatwo nie odnaleźć),

b) wśród 12 korekt, które masz do naniesienia na 4 kopiach oryginalnego zdjęcia poprawki na różnych zdjęciach mogą się powtarzać (bo autorzy pracowali samodzielnie i nie wiedzieli nic o korektach koleżanek i kolegów), ale możesz umieścić tylko jedną, identyczną korektę na wszystkich 4 zdjęciach, a inną identyczną na maksymalnie 3 zdjęciach. Wszystkie pozostały korekty muszą być różne od siebie.

Jury chce przygotować omówienie wyników konkursu w postaci prezentacji. Pomóż w realizacji tego planu i przygotuj prezentację, która spełni następujące warunki:

a) powinna zawierać slajd z klasyfikacją (imię, nazwisko, klasa) 4 uczniów i uczennic, którzy zwyciężyli w konkursie. Pierwsze miejsce zajęła naturalnie osoba (dziewczyna), której zdjęcie miało 4 zaliczone korekty, osoby z zaliczonymi 3 korektami na zdjęciu zajęły równorzędne 2 miejsce, a trzecie nagrodzone miejsce zajęła osoba z dwoma zaliczonymi korektami na swoim zdjęciu,

b) powinna zawierać na osobnych slajdach 4 zwycięskie zdjęcia wraz z opisem, gdzie znajdują się na każdym z nich zaliczone korekty (retusze) - niechcący pomożesz tym opisem osobie sprawdzającej zadanie, gdyby miała trudność w znalezieniu dobrze ukrytych retuszy Twojego autorstwa,

c) zdjęcia powinny „pojawiać się” na slajdach z użyciem efektu animacji, ale te efekty powinny być identyczne dla każdego prezentowanego zdjęcia,

d) prezentację powinno dać się przeglądać zarówno w zwykłym trybie, jak i z wykorzystaniem menu, które powinno być umieszczone na pierwszym slajdzie i z którego można przechodzić bezpośrednio do slajdów z poszczególnymi nagrodzonymi zdjęciami i opisami wykonanych na nich korekt oraz do slajdu z danymi zwycięzców konkursów. Powinna być także stworzona możliwość powrotu ze slajdów ze zdjęciami oraz z wynikami konkursu do slajdu-menu,

e) na każdym slajdzie w jego dolnej części winna być uwidoczniona aktualna data (czyli data dnia, w którym przeglądamy prezentację) oraz tytuł konkursu, który brzmiał „Poprawiamy rzeczywistość wokół nas".

Do oceny oddajesz pliki o nazwie *Zadanie2-Zdjecie1, Zadanie2-Zdjecie2, Zadanie2-Zdjecie3 oraz Zadanie2-Zdjecie4* zawierające zdjęcia z odpowiednio 4, 3, 3 oraz 2 korektami naniesionymi w zdjęciu oryginalnym, a także plik *Zadanie2-Prezentacja* zawierający prezentację podsumowującą szkolny konkurs graficzny wykonaną według opisanych w treści zadania założeń.

8 punktów

Zadanie 3

W pliku tekstowym opisano cyfrowo obraz stworzone z wykorzystaniem grafiki bitmapowej. Opis takiego obrazu zawiera informację o jego rozdzielczości oraz w kolejnych wierszach pliku dokładny opis każdego punktu tej bitmapy (kolejne wiersze opisują kolejne punkty bitmapy począwszy od lewego górnego rzędami od lewej do prawej do prawego dolnego). Punkt w lewym górnym rogu bitmapy ma współrzędne (0,0), a jeżeli jej rozdzielczość to m*n, to punkt w prawym dolnym rogu bitmapy ma współrzędne (m-1,n-1). Opis pojedynczego punktu to opis jego barwy zrealizowany przy pomocy modelu barw RGB. Są to trzy liczby zapisane za pomocą systemu dwójkowego, każda z nich jest wyrażona w postaci 8 bitów (czyli ich wartość dziesiętna to liczba z zakresu od 0 do 255) . Pierwsza z tych liczb podaje „natężenie" koloru czerwonego, druga zielonego, a trzecia niebieskiego dla danego piksela

definiując w ten sposób pewną barwę wypadkową. Układ 11111111 11111111 11111111 (trzy razy 255 dziesiętnie) opisuje kolor biały, a układ 00000000 00000000 00000000 (trzy razy 0 dziesiętnie) kolor czarny.

Przyjmijmy jeszcze dwie definicje:

1. Punkt stanowi **inwersją cyfrową** w stosunku do innego punktu, gdy składowe cyfrowe opisu jego barw po zsumowaniu ze składowymi cyfrowymi opisującymi barwę tego innego punktu dają trzy wartości 255 (czyli kolor biały). Np. punkt (120,130, 0) jest inwersją cyfrową w stosunku do punktu (135, 125, 255).
2. Otoczeniem punktu na bitmapie są 4 punkty położone z jego lewej, prawej, górnej i dolnej strony czyli tak jak pokazano poniżej:

x
xPx
x

P- to punkt, a x- to 4 punkty z jego otoczenia. Zauważmy jeszcze, że punkty w lewym górnym , prawym górnym, lewym dolnym i prawym dolnym rogu bitmapy maja tylko dwa punkty w swoim otoczeniu, a te które leżą w pierwszym i ostatnim wierszu oraz w pierwszej i ostatniej kolumnie bitmapy tylko 3 punkty.

Napisz program, który na podstawie opisu cyfrowego obrazu umieszczonego w pliku tekstowym rozwiąże następujące problemy:

a) odtworzy całą bitmapę (a dokładniej kolory poszczególnych jej punktów) w postaci wierszy i kolumn, przy czym w opisie tym umieści tylko 6 liter, które mogą charakteryzować poszczególne punkty tzn.

R- jeśli punkt jest w kolorze czerwonym
G- jeśli punkt jest w kolorze zielonym
B – jeśli punkt jest w kolorze niebieskim
W- jeśli punkt jest w kolorze białym
C- jeśli punkt jest w kolorze czarnym
I – jeśli punkt jest w innym kolorze niż wymienione wyżej

b) obliczy ile mamy punktów w poszczególnych barwach (tych sześciu, które wymieniono w punkcie a),

c) wypisze współrzędne (numer wiersza i kolumny) punktu o barwie C, który jest najbardziej oddalony od lewego górnego rogu bitmapy , gdzie przez odległość od tego lewego górnego rogu bitmapy rozumiemy sumę numeru wiersza i numeru kolumny danego punktu. Jeżeli nie ma w ogóle takiego punktu to jako wynik należy podać współrzędne -1, -1,

d) oceni czy punkty o barwie I są rozmieszczone symetrycznie względem poziomej linii dzielącej bitmapę na dwie części o równej liczbie wierszy w przypadku bitmapy o parzystej liczbie wierszy lub względem środkowego wiersza w przypadku bitmapy zawierającej nieparzysta liczbę wierszy (Uwaga ! Brak w bitmapie punktów I nie oznacza symetrii),

e) obliczy ile mamy w otoczeniu każdego punktu bitmapy punktów będących jego inwersją cyfrową, a potem zsumuje te wyniki dla wszystkich punktów bitmapy i poda tylko tę zsumowaną wartość.

Dane wejściowe:

Plik ***piksele.txt*** zawierający w swoim pierwszym wierszu oddzielone spacją dwie liczby naturalne **m** oraz **n**, każda większa od 2 oraz nie większa niż 20 opisujące rozdzielczość bitmapy m*n. W każdym z kolejnych m*n wierszy tego pliku znajdują się trzy liczby zapisane w systemie dwójkowym, oddzielone spacją i złożone dokładnie z 8 zer lub jedynek. Liczby te opisują w kolejności: „natężenie” koloru czerwonego, zielonego oraz niebieskiego dla danego punktu bitmapy. Opis punktów jest podawany w takiej kolejności , że w drugim wierszu pliku zdefiniowano barwy punktu w lewym, górnym rogu bitmapy, w trzecim punktu na prawo od niego, w wierszu n+1 mamy opis barw ostatniego punktu z pierwszego wiersza bitmapy. Dalej od wiersza n+2 w pliku rozpoczyna się opis drugiego wiersza bitmapy, a jest w nim dokładnie opisany skrajnie lewy punkt tego drugiego wiersza bitmapy itd. są opisywane kolejne punkty kolejnych wierszy bitmapy zawsze od lewej do prawej. W ostatnim m*n+1 wierszu pliku znajdziemy cyfrowy opis barw prawego dolnego punktu bitmapy.

Dane wyjściowe:

Umieszczone w pliku ***bitmapa. txt*** m+9 wierszy w których kolejno powinny się znaleźć:

- w każdym z pierwszych m wierszy po n znaków oddzielonych spacją, przy czym każdy znak może być tylko jedną z liter: R,G,B, W,C oraz I. Ten układ m wierszy na n kolumn stanowi opis barw punktów bitmapy począwszy od jej lewego górnego roku, a skończywszy na rogu prawym dolnym. Poszczególne litery opisują barwy kolejnych punktów wg schematu podanego w treści zadania,
- w wierszach od m+1 do m+6 znajduje się po jednej liczbie w każdym. Oznaczają one kolejno: liczbę punktów w kolorze R w bitmapie, liczbę punktów w kolorze G w bitmapie, liczbę punktów w kolorze B w bitmapie, liczbę punktów w kolorze W umieszczonych w bitmapie, liczbę punktów w kolorze C w bitmapie oraz liczbę punktów w kolorze I w bitmapie. Znaczenie poszczególnych liter odpowiadających różnym kolorom punktów podano w treści zadania,
- w wierszu m+7 znajdują się dwie liczby oddzielone spacją- jest to numer wiersza i kolumny punktu o barwie C, który ma największą sumę numeru wiersza i kolumny czyli jest najbardziej oddalony od lewego górnego rogu bitmapy, jeżeli nie było w ogóle punktów typu C to w tym wierszu należy zapisać współrzędne -1, -1,
- w wierszu m+8 znajduje się słowo TAK lub NIE. TAK, jeżeli punkty w kolorze I są w bitmapie rozmieszczone symetrycznie względem linii dzielącej bitmapę na część górną i dolną (przy parzystej liczbie wierszy bitmapy) , albo symetrycznie względem środkowego wiersza (przy nieparzystej liczbie wierszy bitmapy). Słowo NIE powinno pojawić się wtedy, gdy opisane symetryczne ułożenie punktów w kolorze I nie ma miejsca
- w ostatnim m+9 wierszu pliku znajduje się jedna liczba będąca sumą punktów z całej bitmapy, które w otoczeniu każdego punktu stanowią jego cyfrową inwersję (definicję otoczenia oraz inwersji cyfrowej podano w treści zadania).

Przykład

Jeżeli w pliku *piksele.txt* mamy następujące dane:

3 3
00000000 00000000 11111111
11111111 11111111 00000000
10100001 11100011 10101010
11111111 00000000 00000000
00000000 11111111 00000000
00000000 00000000 00000000
11111111 11111111 11111111
10001111 10101001 11111100
10010001 01100011 10000011

to w pliku *bitmapa.txt* powinny się znaleźć następujące dane:

B I I
R G C
W I I
1
1
1
1
1
4
1 2
TAK
2

Krótkie objaśnienie do ostatniego wyniku: tylko w pierwszym wierszu mamy punkty stanowiące wzajemnie swoją cyfrową inwersję- punkt o kolorze B jest inwersją punktu o kolorze I (tego po jego prawej stronie) i na odwrót.

Do oceny oddajesz plik zawierający kod źródłowy napisanego przez Ciebie programu. Nazwa tego pliku to *Zadanie3.*

\ **11 punktów**

Zadanie 4

I. Poniżej podano wzór na obliczanie wartości pewnego ciągu rekurencyjnego:

$$a_n = \begin{cases} 2 * a_{n-1} + n & gdy\ n > 1 \\ 2, & gdy\ n = 1 \end{cases}$$

Oto kilka pierwszych wyrazów ciągu otrzymanego z powyższego wzoru: 2,6, 15, 34, 73,…
Spróbuj sobie dla porównania wypisać jeszcze kilka kolejnych wyrazów tego ciągu, a następnie napisz program, który dla pewnego naturalnego k (k>1) podanego z klawiatury obliczy sumę k pierwszych wyrazów tego ciągu i wypisze ją na ekranie, **ale nie wykorzystuje do obliczania wartości sumowanych wyrazów ciągu rekurencji, ale iterację.**

Do oceny oddajesz pliki *Zadanie4-I* zawierający kod napisanego przez Ciebie programu.

II. W tej części mamy inny ciąg rekurencyjny:

$$a_n = \begin{cases} a_{n-1} + a_{n-2} + a_{n-3} & gdy\ n > 3 \\ 1 & gdy\ n = 1 \\ 2 & gdy\ \ n = 2 \\ 3 & gdy\ n = 3 \end{cases}$$

Kilka jego pierwszych wyrazów to: 1,2,3,6,11,20,37…

Twoje zadanie polegać będzie na napisaniu programu, w którym znajdzie się funkcja rekurencyjna obliczająca wyrazy opisanego ciągu, a także który, dla zapisanego w pliku tekstowym dowolnego, złożonego z n wyrazów ciągu obliczy ile jego wyrazów jest jednocześnie wyrazami zdefiniowanego powyżej ciągu rekurencyjnego.

Dane wejściowe:

W pierwszym wierszu pliku *dane.txt* liczba naturalna **n (**zakres od 1 do 25) określająca liczbę kolejnych wierszy, w których zapisano (po jednej w wierszu) liczby naturalne będące elementami pewnego ciągu. Liczby te są nie większe od 30 000.

Dane wyjściowe:

Zapisana w pliku *rekur.txt* jedna liczba, będąca ilością elementów ciągu z pliku *dane.txt*, które są wyrazami opisanego w treści zadania ciągu rekurencyjnego.

Przykład

Jeżeli w pliku *dane.txt* mamy następujące liczby:

6
37
2
12
68
125
17

to w pliku *rekur.txt* powinna być zapisana liczba

4

Wyjaśnienie do przykładu: Do opisanego ciągu rekurencyjnego z ciągu zapisanego w *dane.txt* należą liczby 2, 37, 68, a także 125 (proszę pamiętać, że wypisaliśmy tylko kilka pierwszych wyrazów, które należą do ciągu rekurencyjnego. Formuła, wg której tworzone są kolejne wyrazy definiuje wśród kolejnych niewypisanych wyżej wartości także liczbę 125).

Do oceny oddajesz pliki *Zadanie4-II* zawierający kod napisanego przez Ciebie programu.

8 punktów

Zadanie 5

Rozważmy m ciągów liczb naturalnych (m>2), z których każdy może zawierać inna ilość liczb ,ale nie mniejszą niż 3.
Na rysunku poniżej pokazano sytuację, w której mamy m=6 ciągów. Jak widać liczba ich elementów waha się w tym przykładzie między 3, a 7.

	1	2	3	4	5	6	7
1	2	3	4				
2	1	4	1	**1**	12	15	15
3	7	5	6	**8**			
4	14	2	11	**11**	13	17	
5	6	11	12	**12**	24	32	
6	8	**8**	**9**	**57**	**100**	**101**	**102**

Załamaniem będziemy nazywać trzy ciągi o kolejnych numerach tzn. i, i+1, i+2 (1<=i<=m-2), dla których liczebność ciągu o numerze i+1 (niejako środkowego w trójce) jest mniejsza niż liczebność ciągów o numerach i oraz i+2. Dla przykładu z rysunku mamy tylko jedno załamanie utworzone przez ciągi o numerach 2, 3 i 4.

Przeplataniec to każdy ciąg składający się z **elementów wszystkich ciągów** znajdujących na danej pozycji (pod danym numerem indeksu) pod warunkiem, że ciąg jest złożony na przemian z elementu nieparzystego i parzystego lub odwrotnie. Dla przykładu z rysunku przeplatańcem jest ciąg w kolumnie 2 oraz ciąg w kolumnie 3. Nie jest nim natomiast ciąg w kolumnie 4, bo choć elementy nieparzyste i parzyste występują w nim naprzemiennie, ale brakuje w nim elementu z ciągu nr 1, który posiada jedynie 3 elementy.

Napisz program, który dla danych m ciągów o różnej liczbie elementów rozwiąże następujące problemy:

a) poda współrzędne najdłuższego podciągu rosnącego utworzonego z następujących po sobie elementów (przynajmniej dwóch) biorąc pod uwagę każdy ciąg z osobna (a więc niejako znajdź maksymalny podciąg rosnący ze względu na wiersze). Dla przykładu z rysunku jest nim podciąg będący fragmentem ciągu nr 6, którego elementy zostały pogrubione. Przez podanie współrzędnych rozumiemy określenie początku i końca tego podciągu wg zasady (numer ciągu, numer elementu). Dla przykładu z rysunku zaznaczony pogrubieniem podciąg należący do 6 ciągu należałoby wskazać następująco (6,2) – (6,7). W przypadku, gdyby istniało więcej niż jeden podciągów o poszukiwanej w tej części zadania własności jako rozwiązanie należy wskazać dowolny z nich. Z kolei w przypadku, gdyby nie było takich podciągów w ogóle powinna być wypisana informacja BRAK CIAG,

b) poda współrzędne najdłuższego podciągu rosnącego utworzonego z następujących po sobie elementów (przynajmniej dwóch) biorąc pod uwagę elementy **wszystkich ciągów** występujące pod tym samym indeksem (numerem), a więc chodzi niejako o maksymalny podciąg rosnący ze względu na umowne kolumny. Dla przykładu z rysunku jest nim podciąg będący fragmentem umownej kolumny numer 4 (czyli złożony z elementów o tym indeksie) , którego elementy zostały pogrubione. Przez podanie współrzędnych rozumiemy określenie początku i końca tego podciągu wg zasady (numer ciągu, numer elementu). Dla przykładu z rysunku zaznaczony pogrubieniem podciąg należałoby wskazać następująco (2,4) – (6,4). W przypadku, gdyby istniało więcej niż jeden podciągów o poszukiwanej w tej części zadania własności jako rozwiązanie należy wskazać dowolny z nich. Z kolei w przypadku, gdyby nie było takich podciągów powinna być wypisana informacja BRAK INDEKS. **Uwaga !**

Podciągi w tym punkcie muszą być tworzone przez występujące po sobie w danej kolumnie elementy, jeżeli występuje tam przerwa (bo któryś z ciągów nie ma elementu o danym indeksie) to podciąg się po prostu kończy na elemencie poprzedzającym tę przerwę. Np. w kolumnie związanej z indeksem 5 nie ma jednego ciągu rosnącego 12,13,24,100, ale mamy tu dwa odrębne podciągi: jednoelementowy z liczbą 12 oraz podciąg złożony z liczb 13,24 i 100 i takie podciągi należy uwzględniać przy rozwiązywaniu tej części problemu,

c) obliczy ile mamy załamań dla tego zestawu danych (wynikiem może być naturalnie 0),

d) wypisze wszystkie numery indeksów, dla których występują przeplatańce bądź słowo BRAK, gdy nie będzie, ani jednego przeplatańca.

Dane wejściowe:

Plik tekstowy ***ciagi.txt*** zawierający w pierwszym wierszu liczbę naturalną **m,** większą od 2, ale nie większą niż 20 oznaczającą liczbę ciągów, a w kolejnych m wierszach opis kolejnych ciągów. W każdym z tych wierszy mamy jako pierwszą liczbę naturalną k (równą przynajmniej 3, ale nie większą niż 30) oznaczającą liczbę elementów ciągu opisywanego w tym wierszu, a potem oddzielonych spacją k liczb naturalnych należących do ciągu opisywanego w tym wierszu, Zakładamy, że liczby ciągu są nie większe niż 10^9.

Dane wyjściowe:

Zapisane w pliku ***wyniki.txt*** w 4 kolejnych wierszach wyniki działania programu.

W pierwszym wierszu powinna się pojawić informacja dotycząca współrzędnych najdłuższego podciągu rosnącego spośród znajdujących się w kolejnych m ciągach lub słowa BRAK CIAG, gdy nie ma podciągu rosnącego długości przynajmniej 2 w żadnym z m ciągów.

W drugim wierszu powinna się pojawić informacja dotycząca współrzędnych najdłuższego podciągu rosnącego w ciągach, które można utworzyć z elementów o takim samym indeksie (numerze kolumny) lub słowo BRAK INDEKS, gdy nie ma podciągu rosnącego o długości

przynajmniej 2 w żadnym z tak tworzonych ciągów. Przy czym brakujące w danej kolumnie elementy powodują, że elementy przed „przerwą" i za nią stanowią odrębne podciągi.

Uwaga ! Przez współrzędne podciągu dla obu wyników z pierwszych dwóch wierszy pliku *wyniki.txt* rozumiemy zapis początek podciągu -koniec podciągu ciągu, gdzie z kolei początek, a także koniec podciągu to zapisana w nawiasie para liczb (numer ciągu, nr indeksu elementu).

W trzecim wierszu pliku ***wyniki.txt*** powinna być umieszczona jedna liczba, która podaje liczbę załamań odpowiadającym danym z pliku ***ciągi.txt*** (definicja załamania została podana wyżej).

W czwartym wierszu powinien być umieszczony ciąg liczb oddzielonych spacją oznaczających numery indeksów, dla których ciągi złożone z elementów o tym właśnie indeksie (we wszystkich m ciągach stanowiących dane) tworzą przeplatańce (definicję przeplatańca podano wyżej) albo słowo BRAK, gdy nie ma żadnego przeplatańca.

Przykład (odpowiadający rysunkowi)

Wówczas w pliku ciągi.txt mamy następujące dane:

6

3 2 3 4

7 1 4 1 1 12 15 15

4 7 5 6 8

6 14 12 11 11 13 17

6 6 11 12 12 24 32

7 8 8 9 57 100 101 102

a w pliku ***wyniki.txt*** powinniśmy otrzymać następujące wyniki:

(6,2) – (6,7)

(2,4) – (6,4)

1

2 3

Do oceny oddajesz plik zawierający kod źródłowy napisanego przez Ciebie programu. Nazwa tego pliku to *Zadanie5.*

12 punktów

Razem w całym zestawie 50 punktów.