
Problem 1 (sortowanie)
Zaimplementuj następujące algorytmy sortowania:
(a) sortowanie przez zliczanie
(b) sortowanie przez scalanie MergeSort (w wersji rekurencyjnej)
(e) sortowanie przez kopcowanie HeapSort (w wersji rekurencyjnej)
(d) sortowanie szybkie QuickSort (w wersji rekurencyjnej)
Przetestuj algorytm na wczytanym z pliku ciągu liczb z zakresu [-10000...10000].
Dane:
- W pierwszej linii pliku In0201.txt znajduje się liczba n określająca długość ciągu.
- W drugiej linii pliku znajduje się n wartości z zadanego przedziału. Liczby są separowane pojedynczą spacją.

Wyjście:
- W pierwszej linii pliku Out0201.txt znajduje się posortowany niemalejąco ciąg liczb.

Przykład
In0201.txt
18 // n
1 4 8 4 7 9 11 –1 5 3 7 8 8 3 5 5 1 9
Out02.01.txt
–1 1 1 3 3 4 4 5 5 5 7 7 8 8 8 9 9 11

Zadanie 1
(a) Podaj **rekurencyjne** rozwiązania problemów 2, 3, 4, 5, 6, 7, 8, 9, 10.
(b) Komunikacja programu z użytkownikiem powinna się odbywać za pomocą plików o podanych niżej formatach.

Problem 2 (trójkąt)
Napisz rekurencyjną funkcję, która rysuje (zaczynając od górnego poziomu) przedstawiony na rysunku trójkąt o podstawie n ($n \in NP$, $1 \le n \le 31$), zbudowany ze znaków '*'.

```
   *
  ***
 *****
*******
```

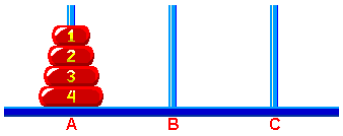
Przykład
In0202.txt
7 // n
Out02.02.txt
n=7

```
   *
  ***
 *****
*******
```

Problem 3 (ciąg Fibonacciego) (patrz [PW])
Ciąg Fibonacciego - ciąg liczb 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377..., z których każda (z wyjątkiem dwóch pierwszych) jest sumą dwóch poprzednich.
Napisz algorytm wyznaczający ciąg liczb Fibonacciego mniejszych od zadanej liczby 1<=n<=10000.

Przykład
In0203.txt
400 // n
Out02.03.txt
n=400
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377

Problem 4 (trzy paliki) (patrz [www], [PW])



Mamy trzy pionowe paliki. Na jeden z nich nałożono N krążków o różnych rozmiarach (im wyżej tym mniejsze krążki). Należy zgodnie z regułami:
1. można przekładać jednocześnie tylko 1 krążek z wierzchu palika,
2. nie można kłaść większego krążka na mniejszy,
przemieścić wszystkie krążki na drugi palik, przy pomocy trzeciego palika. Każdy ruch należy opisać dwoma cyframi (A_i -> B_i), co odpowiada przeniesieniu najwyższego krążka z palika A_i na palik B_i.
Należy więc wyznaczyć odpowiedni ciąg 2^N-1 ruchów.
Dane:
$1 \le N \le 10$; $k = 2^N-1$

Przykład
In0204.txt
4 // N
Out0204.txt
N=4
1->3, 1->2, 3->2, 1->3, 2->1, 2->3, 1->3, 1->2, 3->2, 3->1, 2->1, 3->2, 1->3, 1->2, 3->2, // A_1->B_1, A_2->B_2, ...

Problem 5 (NWD) – algorytm Euklidesa (patrz [www])
Wyznaczyć największy wspólny dzielnik NWD dwóch liczb całkowitych nieujemnych korzystając z algorytmu Euklidesa.
Idea: mając do policzenia NWD(a, b) sprawdzamy, czy b=0. Jeśli tak jest, to NWD(a, b)=a, w przeciwnym przypadku wywołujemy rekurencyjnie algorytm dla liczb b i (a%b).
Np.: NWD(54,69)= NWD(69,54)=NWD(54,15)=NWD(15,9)=NWD(9,6)=NWD(6,3)=NWD(3,0)=3

Przykład
In0205.txt
54 69
Out0205.txt
NWD(54,69)=3

Przykład 6 (przeglądanie grafu w głąb) (patrz [WL])
Sprawdź, czy zadany graf nieskierowany, bez pętli (tj. krawędzi postaci {x, x}) (reprezentowany tablicą list incydencji) jest spójny metodą przeglądania w głąb. Wypisz kolejno odwiedzone wierzchołki.
Dane:
- W pierwszej linii pliku In0206.txt podana jest liczba całkowita 1<=n<=10000 określająca ilość wierzchołków grafu.
- W kolejnych n liniach umieszczone są listy incydencji dla każdego z wierzchołków, tj. w i+1-szej linii (1<=i<=n) podane są numery wierzchołków incydujących z wierzchołkiem i-tym.

Wyjście:
- W pierwszej linii pliku Out0206.txt należy zapisać odpowiedź „graf spójny" bądź „graf niespójny".
- W kolejnej linii należy podać ciąg liczb reprezentujący kolejność odwiedzanych wierzchołków.

Przykład
In0206.txt
8
2 4 5 6
1 3 4 8
2 8
1 2 6 8

1 6
1 4 5 7 8
6 8
2 3 4 6 7
Out0206.txt
Graf spójny
1 2 3 8 4 6 5 7

Problem 7 (przeglądanie drzewa binarnego) (patrz [NW])
Dla drzewa binarnego (BST) zaimplementuj:
1. operację INSERT (wkładanie elementu) do drzewa,
2. funkcję przeglądania drzewa w porządku KLP (korzeń-lewy-prawy).

Przykład
In0207.txt
8 4 9 6 2 7 4 2 8 3
Out0207.txt
8, 4, 2, 3, 6, 7, 9

Problem 8 (Magiczna kostka)
Wypełnij kostkę sześcienną o rozmiarze $n \times n \times n$ tekstem T (podanym przez użytkownika) w następujący sposób.
Sześcian składa się z n tablic o rozmiarze $n \times n$, ustawionych jedna za drugą. Każda tablica stanowi magiczny kwadrat wypełniony (w odpowiedni sposób) literami tekstu T. Tekst T jest powielany, tzn. dla T = ABCDEFGH! generowany jest ciąg:

T = ABCDEFGH!ABCDEFGH!ABCDEFGH!...,

którego kolejne znaki są wykorzystywane w nieprzerwany sposób do wypełniania wszystkich (od 0 do $(n-1)$-szej) tablic sześcianu. Tablice o numerach parzystych (dla i = 0, 2, 4, 6, ...) uzupełniane są literami ciągu T zgodnie z podanym w przykładzie poniżej wzorcem (tj. po każdorazowym rekurencyjnym wywołaniu funkcji wypełniającej ramkę, elementy ciągu T powinny być umieszczone na obwodzie kwadratu w następującym porządku: górna krawędź (od lewej do prawej), prawa krawędź (od góry do dołu), dolna krawędź (od prawej do lewej), lewa krawędź (od dołu do góry)). Tablice o numerach nieparzystych (tj. dla i = 1, 3, 5, ...) są wypełniane przy wykorzystaniu tej samej drogi, ale kolejne litery ciągu umieszczane są na niej w odwrotnej kolejności, tj. zaczynamy wypełnianie kwadratu od jego środka (patrz przykład poniżej). Wszystkie kwadraty należy wypełnić literami w taki sposób, aby ciąg T nie został przerwany w żadnym miejscu (tj. jeśli generując tablicę 0, zakończyliśmy na znaku G, to tablicę 1 rozpoczynamy zapełniać od znaku H oraz litery H i G muszą sąsiadować ze sobą w sześcianie).
Przykład:

```
T = ABCDEFGH!, n = 5

Tablica 0       Tablica 1       Tablica 2       Tablica 3       Tablica 4
[A, B, C, D, E] [E, D, C, B, A] [F, G, H, !, A] [A, !, H, G, F] [B, C, D, E, F]
[G, H, !, A, F] [H, G, F, E, !] [C, D, E, F, B] [D, C, B, A, E] [H, !, A, B, G]
[F, F, G, B, G] [!, !, H, D, H] [B, B, C, G, C] [E, E, D, !, D] [G, G, H, C, H]
[E, E, D, C, H] [A, A, B, C, G] [A, A, !, H, D] [F, F, G, H, C] [F, F, E, D, !]
[D, C, B, A, !] [B, C, D, E, F] [!, H, G, F, E] [G, H, !, A, B] [E, D, C, B, A]
```

Dane:
- W pierwszej linii pliku In0208.txt podana jest liczba całkowita 1<=n<=100 określająca rozmiar kostki $n \times n \times n$.
- W drugiej linii pliku znajduje się powielany tekst T.

Wyjście:
- W pierwszej linii pliku Out0208.txt znajduje się tekst "n=..., T=..." zgodny z poniższym wzorcem.
- W kolejnych liniach umieszczone są wartości kolejnych tablic (zgodnie z poniższym wzorcem.

Przykład
In0208.txt
4 // n
ABCDEFGH! // T
Out0208.txt

```
n=4, T= ABCDEFGH!
Tablica 0       Tablica 1       Tablica 2       Tablica 3       Tablica 4
[A, B, C, D, E] [E, D, C, B, A] [F, G, H, !, A] [A, !, H, G, F] [B, C, D, E, F]
[G, H, !, A, F] [H, G, F, E, !] [C, D, E, F, B] [D, C, B, A, E] [H, !, A, B, G]
[F, F, G, B, G] [!, !, H, D, H] [B, B, C, G, C] [E, E, D, !, D] [G, G, H, C, H]
[E, E, D, C, H] [A, A, B, C, G] [A, A, !, H, D] [F, F, G, H, C] [F, F, E, D, !]
[D, C, B, A, !] [B, C, D, E, F] [!, H, G, F, E] [G, H, !, A, B] [E, D, C, B, A]
```

Problem 9 (przecięcie zbiorów)
Podaj możliwie najefektywniejszy czasowo algorytm (tj. o złożoności $O(n \lg n)$) wyznaczający przecięcie zbiorów A i B ($A \cap B$). Zakładamy, że zbiory A i B zawierają wartości całkowite z zakresu od 1 do 100000 i reprezentowane są odpowiednio tablicami A i B oraz że elementy zbioru B (tablica B) są uporządkowane rosnąco.
Dane:
- W pierwszej linii pliku In0209.txt znajduje się liczba n określająca liczność zbiorów A i B.
- W drugiej linii pliku znajduje się n wartości zbioru A oddzielonych pojedynczą spacją.
- W trzeciej linii pliku znajduje się n uporządkowanych wartości zbioru B oddzielonych pojedynczą spacją.

Wyjście:
- W pierwszej linii pliku Out0209.txt znajdują się liczby należące do przecięcia zbiorów A i B.
- W drugiej linii pliku znajduje się liczba określająca ich ilość.

Przykład
In0209.txt
12
9 **4 6** 11 24 **12 16** 49 33 16 **8 18**
1 3 **4 6** 7 **8** 9 **12 16 18** 25 27
Out0209.txt
4 6 8 12 16 18
6

Problem 10 (metoda bisekcji)
Napisz funkcję wyznaczającą (**metodą bisekcji**) przybliżony pierwiastek równania f(x)=0 w przedziale [a, b] z dokładnością E.
[Tw. Darboux] Jeśli funkcja f jest ciągła na przedziale [a,b] oraz spełnia warunek f(a)f(b)<0 (tzn. wartości funkcji na końcach przedziału mają różne znaki), to istnieje punkt $c \in (a,b)$ taki, że f(c)=0. Jeśli dodatkowo funkcja f jest ściśle monotoniczna (tzn. rosnąca albo malejąca) w [a,b], to punkt c jest jedyny.
Dane:
- W pierwszej linii pliku In0210.txt znajduje przedział [a,b], dokładność E oraz postać funkcji f(x) (którą należy wczytać ręcznie.

Wyjście:

- W pierwszej linii pliku Out0210.txt znajduje się przybliżony pierwiastek

Przykład

In0210.txt
-1 5 0.25 f(x)= x^2-2 // [a,b]=[-1,5], E=0.25
Out0210.txt
1.4375

Zestawy

Zestaw	01	02	03	04	05	06	07	08	09	10	11	12
Zad1	1a	1b	1c	1d	1a	1b	1c	1d	1a	1b	1c	1d
Zad2	2	2	2	2	4	4	4	4	5	5	5	5
Zad3	3	9	10	3	9	10	3	9	10	3	9	10
Zad4	6	6	7	7	8	8	8	8	6	6	7	7
