Zadanie 3 [8 punktów]

Dane są liczby całkowite dodatnie *n, k* >, przy czym $k \leq \sqrt{n}$. W tablicy a[1..*n*] zapisano *n* liczb całkowitych o co najmniej *k* różnych wartościach. Należy zaprojektować algorytm, który stabilnie i w miejscu przemieści *k* parami różnych liczb na początek tablicy *a* i uporządkuje je rosnąco. Stabilność w tym przypadku oznacza, że spośród liczb o tej samej wartości, na początku tablicy *a* może znaleźć się tylko ta, która pojawia się w niej najwcześniej. Twój algorytm powinien działać w czasie O(*n*log *n*)

Zadanie 1 [5 punktów] Zaprojektuj optymalny algorytm pod względem pesymistycznej liczby porównań, który znajduje dwa środkowe elementy w zbiorze czterech elementów. Dowiedź poprawności swojego rozwiązania.

Zadanie 3 [5 punktów] Danych jest k uporządkowanych list o długościach będących parami różnymi potęgami dwójki. Zaproponuj wydajny algorytm scalenia tych list w jedną listę uporządkowaną. Uzasadnij poprawność swojego algorytmu i dokonaj analizy jego złożoności obliczeniowej ze względu na liczbę porównań wykonywanych podczas scalania.

Zadanie 1 [6 punktów]
Udowodnij, że do scalania dwóch ciągów uporządkowanych długości 2 i 5 potrzeba i wystarcza 5 porównań.

Zadanie 4 [7 punktów]
Powiemy, że dwa napisy są podobne wtedy i tylko wtedy, gdy zawierają jednakowe liczby wystąpień tych samych znaków. Danych jest n napisów nad alfabetem m-znakowym {1, 2, …, m}. Zaproponuj algorytm, który stwierdza, ile jest wśród nich różnych klas napisów podobnych. Twój algorytm powinien działać w czasie O(R + m), gdzie R jest sumą długości wszystkich napisów.

Zadanie 5 [7 punktów]
Dana jest 2n-elementowa tablica zawierająca n zer i n jedynek. Chcemy ją uporządkować tak, żeby zera i jedynki były ułożone na przemian, począwszy od zera, tj. 010101... Zaproponuj efektywny algorytm, który wykona to w miejscu i stabilnie (tj. kolejność zer i kolejność jedynek z wejścia muszą być zachowane).

(10 pkt) Danych jest n słów o takiej samej długości k, zbudowanych ze znaków n-elementowego, uporządkowanego alfabetu. Rozmiarem zadania w tym przypadku jest R = nk.

(4 pkt) Zaproponuj algorytm, który dla danego i, 1<= i<= k, obliczy w czasie O(R) liczbę wszystkich par słów, które różnią się tylko na i-tej pozycji.

(6 pkt) Zaproponuj algorytm, który obliczy w czasie O(R) liczbę wszystkich par słów, które różnią się tylko na dokładnie jednej pozycji.

(10 pkt) W tym zadaniu rozważamy n-elementowe ciągi k-uporządkowane (i-ty element ciągu jest nie większy od elementu i + k), 1<= k<= n.

(5 pkt) Udowodnij, że każdy algorytm sortujący przez porównania wymaga w pesymistycznym przypadku Ω(n log k) porównań do posortowania n-elementowego ciągu k-uporządkowanego.

(5 pkt) Zaproponuj algorytm sortujący takie ciągi w czasie O(n log k).

Zadanie 1 [5 punktów]

[2 punkty] Zaproponuj algorytm, optymalny ze względu na liczbę porównań, sortujący ciąg x[1], x[2], …, x[7], parami różnych liczb całkowitych oraz taki, że dla każdego i = 1, 2, 3, x[i] < x[2i] oraz x[i] < x[2i+1].

[3 punkty] Udowodnij optymalność swojego rozwiązania.

Zadanie 3 [9 punktów]

Rozważamy dynamicznie zmieniający się ciąg $A=<a_1, a_2, \ldots, a_n>$, parami różnych n liczb całkowitych. Na ciągu A dozwolona jest jedyna operacja NaPoczątek(i), $1 \leq i \leq n$, która przesuwa element a_i na początek A.

Przykład
Dla A=<4,1,3,5,2>, po wykonaniu NaPoczatek(3) dostajemy A=<3,4,1,5,2>.

Interesuje nas ciąg operacji NaPoczątek o minimalnej długości, których wykonanie posortuje A. Nazwijmy go *minimalnym ciągiem sortującym*.

a) [3 punkty] Zaprojektuj algorytm, który w czasie O(n) wyznacza pierwszy element minimalnego ciągu sortującego.

b) [2 punkty] Zaprojektuj efektywny algorytm wyznaczający cały minimalny ciąg sortujący.

c) [4 punkty] Udowodnij poprawność swoich rozwiązań.

Dokonaj analizy złożoności czasowej zaproponowanych algorytmów.