# Ćwiczenia 3

2.2 Niech $n$ będzie dodatnią liczbą całkowitą. Dla dodatniej liczby całkowitej $k$ powiemy, że ciąg liczb $a[1]$, …, $a[n]$ jest $k$-*dobry*, jeżeli każda inwersja $(i, j)$, $1 \leqslant i < j \leqslant n$, spełnia $j \leqslant i + k$.

(a) Zaproponuj asymptotycznie optymalny ze względu na porównania algorytm sortujący ciągi $k$-*dobre*. Uzasadnij asymptotyczną optymalność swojego algorytmu.

*Uwaga*: w tym zadaniu argumentami funkcji złożoności są $k$ i $n$.

(b) Zaproponuj wydajny czasowo i pamięciowo algorytm, który sprawdza, czy dany ciąg liczb $a[1]$, …, $a[n]$, dla zadanej dodatniej liczby całkowitej $k$, jest $k$-*dobry*. Uzasadnij poprawność swojego algorytmu i dokonaj analizy jego złożoności czasowej i pamięciowej.

2.5 Dana jest tablica $a[1..n]$ oraz liczba całkowita $k \in \{1, 2, \ldots, n\}$. Zaproponuj liniowy algorytmy przesunięcia cyklicznego elementów tablicy $a$ o $k$ pozycji w lewo.

*Przykład*: ciąg $[1, 2, 3, 4, 5]$ przesunięty cyklicznie o dwie pozycje w lewo będzie miał postać $[3, 4, 5, 1, 2]$.

2.6 Dana jest $n$-elementowa tablica $a[1..n]$ zawierająca tylko 0 i 1.

(a) Zaprojektuj wydajny algorytm sortowania $a$ stabilnie i w miejscu.

(b) Załóżmy, że $n = 2k$ i w $a$ znajduje się dokładnie $k$ zer i $k$ jedynek. Chcemy tablicę $a$ posortować tak, żeby zera i jedynki były ułożone na przemian, począwszy od zera, tj. 010101… Zaproponuj wydajny algorytm, który wykona to w miejscu i stabilnie.

# Do samodzielnej pracy

2.8 Dane są dodatnie liczby całkowite $k$, $n$, $k \leqslant n$, oraz tablica liczb całkowitych $a[1..n]$ taka, że podtablice $a[1..k]$ i $a[k+1..n]$ są uporządkowane niemalejąco. Przy założeniu, że $k = O(\sqrt{n})$ zaproponuj algorytm sortowania tablicy (scalania dwóch ciągów uporządkowanych) w miejscu i w czasie $O(n)$.

*Wskazówka*: zastosuj sortowanie przez wstawianie i skorzystaj z rozwiązania zadania 2.5.

2.9 Niech $a[1..n]$ będzie tablicą liczb całkowitych. Przyjmijmy, że $n = r^2$ dla pewnego naturalnego $r$. Zawartość tablicy $a$ traktujemy jako zapis $r$ rekordów (bloków). Każdy rekord-blok zajmuje spójny fragment tablicy od pozycji $(i-1)r+1$ do pozycji $ir$, dla pewnego $i \in [1..r]$. Kluczem w rekordzie jest ostatni element bloku. Zaproponuj sortowanie rekordów względem ich kluczy. Twój algorytm powinien działać w miejscu i w czasie liniowym. Kolejność elementów w rekordzie nie może ulec zmianie.

*Wskazówka*: zastosuj sortowanie przez wybieranie (SelectionSort).

2.10 Dane są dodatnie liczby całkowite $k \leqslant n$ oraz tablica liczb całkowitych $a[1..n]$ taka, że podtablice $a[1..k]$ i $a[k+1..n]$ są uporządkowane niemalejąco. Zaproponuj algorytm sortowania tablicy $a$ (scalania dwóch ciągów uporządkowanych) w miejscu i w czasie $O(n)$.

*Wskazówka*: Rozważmy przypadek, gdy oba scalane podciągi mają długości co najmniej $\sqrt{n}$. Idea algorytmu: podziel sortowane podciągi na bloki o długościach $\sqrt{n}$; posortuj bloki względem ich ostatnich (największych) elementów; potraktuj pierwszy blok (jego elementy) jako bufor; scalaj w pętli dwa kolejne bloki zapisując wynik w buforze i przesuwając bufor w prawo; posortuj bufor i scal go z posortowaną resztą (Knuth, Tom III).