Lista 3, Zadanie 8

Problem

Potrzebujemy algorytmu do sortowania n liczb całkowitych z przedziału od 1 do n^2 w czasie $O(n)$.

Concept

Użyjemy tutaj algorytmu `RadixSort` do podzielenia liczb na części które sortujemy dalej przy pomocy algorytmu `CountingSort`.

Przy czym, wybór sposobu podziału jest istotny podczas naszego *„dzielenia liczb na części”*. Zależy nam na złożoności $O(n)$, więc najlepiej przejść z bazy dziesiętnej na bazę n.
Wówczas `CountingSort` zostanie użyty tylko dwa razy, jako że mamy tylko n^2 liczb do posortowania o postaci $(\overline{ab})_n$, gdzie a, b to cyfry liczby w bazie n.

Oczywiście tutaj pojawia się problem liczby n^2 która wykracza poza ten zakres, ponieważ ma postać $(100)_n$.
Rozwiązaniem może być wprowadzenie specjalnego przypadku dla tej liczby, jako że jest to maximum rozważanego przedziału. Możemy wtedy instancje takich liczb przesuwać na sam koniec posortowanej listy.
Innym rozwiązaniem tutaj może być przesunięcie wszystkich elementów o 1 lokalnie, na czas sortowania. Wówczas mielibyśmy liczby z zakresu $[0, n^2 - 1]$ które jak najbardziej da się zapisać w formacie $(\overline{ab})_n$.

W rozwiązaniu rozważymy algorytm przesuwający liczby o 1 żeby zmieściły się w zakresie $[0, n^2 - 1]$.

Rozwiązanie

Pseudokod

`CustomSort`:
`input` : A - lista n liczb z przedziału $[1, n^2]$
`output`: posortowana lista A'

1. $e = 1$
2. `while` $e < n^2$
    1. $count[n]$
    2. $i = 0$
    3. `while` $i < n$
        1. $count \left[\lfloor \frac{A[i]-1}{e} \rfloor \mod n \right]$++
        2. i++
    4. $i = 1$
    5. `while` $i < n$
        1. $count[i] = count[i-1]$
        2. i++
    6. $tmp[n]$
    7. $i = n - 1$
    8. `while` $i \geq 0$
        1. $tmp \left[count \left[\lfloor \frac{A[i]-1}{e} \rfloor \mod n \right] - 1 \right] = A[i]$
        2. $count \left[\lfloor \frac{A[i]-1}{e} \rfloor \mod n \right]$--
        3. i--
    9. $i = 0$
    10. `while` $i < n$
    11. $A[i] = tmp[i]$
    12. i++
    13. $e = e \cdot n$

Opis pseudokodu

- e jest eksponentą liczby n
- Punkt 2. to pętla która zawsze uruchomi się dwa razy (dla naszego przedziału liczb $[1, n^2]$).
- Punkty od 2.1 do 2.8 to implementacja funkcji `CountSort`, która sortuje poszczególne liczby po ich cyfrach w bazie n.
- Uwaga: lokalnie (w liście $count$) następuje przesunięcie o 1 z powodów opisanych w koncepcie problemu.