Bit Algo
START

Sprawy organizacyjne

- zajęcia regularnie co tydzień: poniedziałek 16:15-17:45 (3.27b), środa 17:50-19:20 (3.27b) i czwartek 17:50-19:20 (3.27a)

- https://www.facebook.com/groups/bitalgoagh/

Sortowanie i szukanie

Złożoność wyszukiwania:

Dane nieposortowane	Dane posortowane
O(n)	O(logn)

`function linear_search(A, n, T):`
`  i := 0`
`  while i < n:`
`  if A[i] == T:`
`  return i`
`  i := i + 1`
`  return unsuccessful`	

Złożoność wyszukiwania:

Dane nieposortowane	Dane posortowane
O(n)	O(logn)

`function linear_search(A, n, T):`
`  i := 0`
`  while i < n:`
`  if A[i] == T:`
`  return i`
`  i := i + 1`
`  return unsuccessful`
	`function binary_search(A, n, T):`
	`  L := 0`
	`  R := n - 1`
	`  while L <= R:`
	`  m := floor((L + R) / 2)`
	`  if A[m] < T:`
	`  L := m + 1`
	`  else if A[m] > T:`
	`  R := m - 1`
	`  else:`
	`  return m`
	`  return unsuccessful`

Złożoność sortowania:

sortowanie	złożoność średnia	złożoność pesymistyczna	stabilność
quicksort	O(nlogn)	$O(n^2)$	NIE
merge sort	O(nlogn)	O(nlogn)	TAK
heap sort	O(nlogn)	O(nlogn)	NIE
count sort	O(n + k)	O(n + k)	TAK
radix sort	O(d(n+k))	O(d(n+k))	TAK

Zadania

Zadanie 1

(KOLOKWIUM) Do szkoły chodzi 2n dzieci. Połowa z nich należy do grupy niebieskiej a połowa do grupy zielonej. Na apelu dzieci miały się ustawić podzielone na grupy, najpierw grupa zielona a potem grupa niebieska. Niestety nauczyciel zapomniał im o tym powiedzieć i teraz stoją w szeregu przypadkowo (z odstępami jednego metra między kolejnymi dziećmi). Szereg reprezentowany jest jako lista struktur typu:

```
struct Pupil {
	int group;
	Pupil* next;
};
```

gdzie group to grupa (1 to niebiescy a 0 to zieloni) a next to wskaźnik pokazujący kolejną osobę w szeregu. Proszę zaimplementować funkcję **`int distanceToIdeal( Pupil* L )`**, która oblicza najmniejszą liczbę metrów jaką dzieci muszą sumarycznie przejść, żeby te z grupy zielonej stały przed tymi z grupy niebieskiej (i żeby cały szereg wciąż stał w tym samym miejscu). Funkcja powinna być poprawna oraz możliwie jak najszybsza. W zadaniu nie wymaga się sortowania listy.

Zadanie 2

(KOLOKWIUM) Proszę zaimplementować funkcję:

```
double AverageScore(double A[], int n, int lowest, int highest);
```

Funkcja ta przyjmuje na wejściu tablicę ***n*** liczb rzeczywistych (ich rozkład nie jest znany, ale wszystkie są parami różne) i zwraca średnią wartość podanych liczb po odrzuceniu **`lowest`** najmniejszych oraz **`highest`** największych. Zaimplementowana funkcja powinna być możliwie jak najszybsza. Proszę oszacować jej złożoność czasową (oraz bardzo krótko uzasadnić to oszacowanie).

Zadanie 3

Proszę zaimplementować funkcję:

```
int findDistinct(int A[], int n);
```

Funkcja ta przyjmuje na wejściu posortowaną tablicę ***n*** liczb całkowitych, w której mogą pojawiać się duplikaty. Funkcja powinna zliczać ilość wystąpień różnych wartości bezwzględnych elementów występujących w tej tablicy.

Przykład:

Wejście: `{-1, -1, 0, 0, 1, 1, 1}`

Wyjście: `2`

Wejście: `{1, 1, 1}`

Wyjście: `1`

Zadanie 4

(KOLOKWIUM) Dana jest struktura Node opisująca listę jednokierunkową:

```
struct Node {
     Node * next;
     int value;
};
```

Proszę zaimplementować funkcję **`Node* fixSortedList( Node* L )`**, która otrzymuje na wejściu listę jednokierunkową bez wartownika. Lista ta jest prawie posortowana w tym sensie, że powstała z listy posortowanej przez zmianę jednego losowo wybranego elementu na losową wartość. Funkcja powinna przepiąć elementy listy tak, by lista stała się posortowana i zwrócić wskaźnik do głowy tej listy. Można założyć, że wszystkie liczby na liście są różne i że lista ma co najmniej dwa elementy. Funkcja powinna działać w czasie liniowym względem długości listy wejściowej.

Zadanie 5

(KOLOKWIUM) Dana jest n elementowa tablica **A** zawierająca liczby naturalne (potencjalnie bardzo duże). Wiadomo, że tablica **A** powstała w dwóch krokach. Najpierw wygenerowano losowo (z nieznanym rozkładem) **n** różnych liczb nieparzystych i posortowano je rosnąco. Następnie wybrano losowo $\lceil \log n \rceil$ elementów powstałej tablicy i zamieniono je na losowo wybrane liczby parzyste. Proszę zaproponować (bez implementacji!) algorytm sortowania tak powstałych danych. Algorytm powinien być możliwie jak najszybszy. Proszę oszacować i podać jego złożoność czasową.

Bit Algo
START