Tema nr. 1: Analiza și Compararea Metodelor Directe de Sortare

Timp alocat: 2 ore

Implementare

Se cere implementarea **corectă** și **eficientă** a 3 metode directe de sortare (sortarea bulelor, sortarea prin inserție – folosind inserție liniară sau binară, și sortarea prin selecție)

Intrare: un șir de numere $< a_1, a_2, ..., a_n >$

Ieșire: o permutare ordonată a șirului de la intrare $< a_1^{'} \leq a_2^{'} \leq ... \leq a_n^{'} >$

Toate informațiile necesare și pseudo-codul se găsesc în notițele de la seminarul nr. 1 (sortarea prin inserție este prezentată și în carte[1] – secțiunea 2.1). Pentru fiecare metodă de sortare ar trebui să selectați versiunea eficientă. În cazul bubble sort pentru punctaj maxim aferent acestei metode de sortare trebuie să implementați versiunea cu complexitate liniară în cazul favorabil.

Cerințe

- Implementarea unei metode de sortare, analiza in caz mediu statistic + demo 5p
- Implementarea celorlalte metode de sortare + demo + analiză caz mediu statistic 2p (cate 1p fiecare)
- Analiză in caz favorabil și defavorabil 3p (cate 0.5p pentru fiecare caz)
- Implementare insertion sort prin insertie binara 0.5p

Evaluare

! Înainte de a începe să lucrați pe partea de evaluare a complexitatii algoritmilor, asigurati-va că aveti un algoritm corect! Corectitudinea algoritmilor va trebui demonstrata pe un vector de dimensiuni mici (care poate să fie codat în funcția „main”).

1. Se cere compararea celor 3 algoritmi în cazurile: **favorabil (best)**, **mediu statistic (average)** și **defavorabil (worst)**. Pentru cazul **mediu** va trebui să repetați măsurătorile de **m** ori (m=5 este suficient) și să raportați media rezultatelor; de asemenea, pentru cazul **mediu,** asigurati-va că folosiți **aceleași** date de intrare pentru cele 3 metode de sortare

[1] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein. *Introduction to Algorithms*

(astfel încât compararea lor să fie corectă); identificați și generați date de intrare pentru cazurile: **favorabil** și **defavorabil**, pentru toate cele 3 metode de sortare.

2. Pașii de analiză ai metodelor de sortare pentru fiecare din cele 3 cazuri (**favorabil, defavorabil, mediu)**:
   - variati dimensiunea șirului de la intrare (n) între [100…10.000], cu un increment de maxim 500 (sugerăm 100);
   - pentru fiecare dimensiune, generati datele de intrare adecvate pentru metoda de sortare; rulați metoda de sortare numărând operațiile (numărul de atribuiri, numărul de comparații și suma lor).

   ! Doar atribuirile („=”) și comparațiile („<”,”==”,”>”,”!=”) care se fac pe datele de intrare și pe datele auxiliare corespunzătoare se iau în considerare.

3. Pentru fiecare caz de analiză (**favorabil, defavorabil** si **mediu),** generati grafice care compara cele 3 metode de sortare; folosiți grafice diferite pentru numărul de atribuiri, comparații și suma lor. Dacă o curba nu poate fi vizualizată corect din cauza că celelalte curbe au o rată mai mare de creștere (ex: o funcție liniară pare constantă atunci când este plasată în același grafic cu o funcție pătratică), atunci plasați noua curbă și pe un alt grafic.
   Denumiti adecvat graficele și curbele.

4. Interpretati graficele și scrieți observațiile în antetul fișierului „.cpp”, într-un comentariu bloc **informativ**.

5. Pregatiti un exemplu pentru exemplificarea corectitudinii fiecărui algoritm implementat

6. Nu preluăm teme care nu sunt indentate și care nu sunt organizate în funcții (de exemplu nu preluam teme unde tot codul este pus in main)

7. ***Punctajele din barem se dau pentru rezolvarea corecta și completa a cerinței, calitatea interpretărilor din comentariul bloc și răspunsul corect dat de voi la întrebările puse de către profesor.***