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

Timp alocat: 2 ore

1.1 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 $< a_1, a_2, \ldots, a_n >$
- Ieșire: o permutare ordonată a șirului de la intrare $< a'_1 \leq a'_2 \leq \ldots \leq a'_n >$

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**). Să verificați că ați implementat varianta eficientă pentru fiecare din algoritmii de sortare (dacă mai multe versiuni au fost prezentate)

1.2 Cerințe minimale pentru notare

Lipsa oricărei cerințe minimale (chiar și parțială) poate rezulta într-o notă mai mică prin penalizări sau refuzul de a prelua tema, rezultând în nota 0.

- *Demo*: Pregătiți un exemplu pentru exemplificarea corectitudinii fiecărui algoritm implementat. Corectitudinea fiecărui algoritm se demonstrează printr-un exemplu simplu (maxim 10 valori).
- Graficele create trebuie să fie ușor de evaluat, adică grupate și adunate prin funcțiile Profiler după cerințele temei. Tema nu va fi evaluată dacă conține o multitudine de grafice negrupate. De exemplu, analiza comparativă implică gruparea într-un singur grafic a algoritmilor comparați.
- Interpretați graficul/graficele și notați observațiile personale în antetul fișierului *main.cpp*, într-un comentariu bloc informativ.
- Nu preluăm teme care nu sunt indentate și care nu sunt organizate în funcții (de exemplu, nu prelăum teme unde tot codul este pus în main).
- *Punctajele din barem sunt corespondente unei rezolvări corecte și complete a cerinței, calitatea interpretărilor din comentariul bloc și* ***răspunsul corect dat de dumeavostră la întrebările puse de către profesor.***

1.3 Cerințe

1.3.1 Implementarea metodelor de sortare (5.5p)

- Bubble sort (1.5p)
- Insertion sort (2p)
- Selection sort (2p)

Demo: Corectitudinea algoritmului va trebui exemplificată pe date de intrare de dimensiuni mici.

1.3.2 Analiză comparativă algoritmilor pentru caz mediu statistic (1.5p – 0.5 per algoritm)

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,** asigurați-vă că folosiți **aceleași** date de intrare pentru cele 3 metode de sortare (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.

Pașii de analiză ai metodelor de sortare pentru fiecare din cele 3 cazuri (**favorabil, defavorabil, mediu)**:

- variați dimensiunea șirului de la intrare (n) între $[100 \ldots 10.000]$, cu un increment de maxim 500 (sugerăm 100).

- pentru fiecare dimensiune, generați datele de intrare adecvate pentru metoda de sortare; rulați metoda de sortare numărând operațiile (numărul de atribuiri, numărul de comparații și suma lor).

1.3.3 Analiză comparativă în caz favorabil și defavorabil (3p - câte 0.5p pentru fiecare caz a fiecărui algoritm)

Pentru fiecare caz de analiză (**favorabil, defavorabil** și **mediu),** generați grafice care compară cele 3 metode de sortare; folosiți grafice diferite pentru numărul de atribuiri, comparații și suma lor. Dacă o curbă 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. Denumiți adecvat graficele și curbele.

1.3.4 Bonus: Insertion sort prin inserție binară (0.5p)

Demo: Corectitudinea algoritmului va trebui exemplificată pe date de intrare de dimensiuni mici.

Va trebui să comparați insertion sort prin inserție binară cu toți ceilalți algoritmi (bubble, insertion, selection) pentru toate cazurile (favorabil, mediu statistic, defavorabil).

1.4 Informații adiționale

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

References

[1] Thomas H. Cormen et al. *Introduction to Algorithms*. 2nd. The MIT Press, 2001. ISBN: 0262032937.