Procedūrinio programavimo pagrindai
12 tema

Yra daugybė algoritmų, skirtų įvairiems taikomiesiems uždaviniams spręsti. Šių užduočių esmė – susipažinti su kai kuriais algoritmais ir susipažinti su algoritmų efektyvumo vertinimo metodais.
Duomenų rikiavimas ir jų paieška – klasikiniai informatikos uždaviniai. Jiems spręsti yra naudojami įvairūs algoritmai, vieni jų labiau, kiti jų mažiau efektyvūs. Turėdamas algoritmo pseudokodą, programuotojas turi gebėti tą algoritmą įgyvendinti, ištestuoti savo parašytą kodą, o taip pat ir įvertinti jo efektyvumą.
Kiekvieną algoritmą (arba jo variantą) realizuokite kaip atskirą funkciją(-as), o funkcijoje main įvertinkite kiekvieno jų efektyvumą ir palyginkite juos tarpusavyje. Kiekvienoje iš funkcijų berikiuodami skaičiuokite, kiek kartų buvo lyginami gretimi elementai, ir kiek kartų jiems buvo vykdomos priskyrimo operacijos. Jei norite, dviem šių operacijų skaitliukams galite naudoti globalius kintamuosius. Main funkcijoje atspausdinkite lentelę, kurioje būtų galima pažiūrėti, kiek operacijų buvo atlikta kiekvienoje iš nagrinėjami algoritmo versijų.
Atliktas užduotis įkelkite į VU VMA, laikydamiesi pateiktų nurodymų.

Užduotis 1a.
Apibrėžkite funkciją, skirtą užpildyti turimą masyvą atsitiktinai sugeneruotais duomenimis. Jei turite jau parašytą realizaciją, galite ja naudotis.
Užduotis 1b.
Apibrėžkite funkciją, skirtą rikiavimo algoritmo korektiškumui tikrinti. Ši funkcija patikrina, ar masyve esantys duomenys surikiuoti reikiama tvarka.
Užduotis 1c.
Realizuokite žemiau nurodytas rikiavimo algoritmų versijas. Svarbu: pirmiausia pagaminkite joms korektišką testavimo aplinką, t.y. tokią, kurioje lyginimas būtų korektiškas, nes kiekviena iš algoritmo versijų eksperimento metu rikiuoja tą patį (nesurikiuotą, programos pradžioje sugeneruotą) duomenų masyvą. Kad sugeneruoti duomenis, naudokitės funkcija iš užduoties 1a, o surikiavę juos vienu ar kitu užduoties 2 algoritmų pasinaudokite užduoties 1b funkcija algoritmo darbo rezultatui patikrinti. Jei algoritmas duomenų neišrikiuoja – atspausdinkite tai rezultatų lentelėje. Tai reiškia, jog programuodami padarėte klaidą.

Algoritmai:
a) Burbuliuko metodas (angl. bubble sort, https://en.wikipedia.org/wiki/Bubble_sort)
b) Greitojo rikiavimo metodas (angl. quick sort, https://en.wikipedia.org/wiki/Quicksort)
c) Įterpimo metodas (angl. insertion sort, https://en.wikipedia.org/wiki/Insertion_sort)
d) Išrinkimo metodas (angl. selection sort, https://en.wikipedia.org/wiki/Selection_sort)
e) Sąlajos metodas (angl. merge sort, https://en.wikipedia.org/wiki/Merge_sort)

Užduotis 2.

a) Perrašykite testavimo aplinką taip, kad būtų atliekamas ne vienas, bet daug eksperimentų (t. y. rikiuojamas ne vienas, o daug skirtingų duomenų rinkinių), ir rezultatų lentelėje atsispindėtų visuminis palyginimas, o ne vieno duomenų rinkinio rezultatai.
b) Atlikite eksperimentus realizuotų algoritmų efektyvumui įvertinti. Eksperimento rezultatus atspausdinkite main funkcijoje, nurodydami ne tik kiek operacijų, bet ir laiko reikėjo vienam ar kitam algoritmui. Išmatuokite, kiek sekundžių algoritmas dirbo (tam pasinaudokite <time.h> aprašytomis funkcijomis)

Papildomos užduotys.

Užduotis 3.
Įvertinkite kiekvieno iš algoritmo priklausomybę nuo duomenų pobūdžio. Kiekvieną iš algoritmų patikrinkite trimis atvejais: a) rikiuodami jau surikiuotą (reikiama tvarka) masyvą, b) rikiuodami priešinga tvarka surikiuotą masyvą, c) rikiuodami atsitiktinį duomenų rinkinį (generuokite pvz. 100 skirtingų duomenų rinkinių ir fiksuokite visų eksperimentų vidurkį). Papildykite rezultatų lentelę (kiekvienam algoritmui skirkite po tris eilutes, kiekvienam iš čia paminėtų atvejų).

Užduotis 4.
Įvertinkite kiekvieno iš algoritmo priklausomybę nuo duomenų kiekio. Kiekvieną iš algoritmų patikrinkite trimis atvejais: a) rikiuodami mažą (20 elementų), b) rikiuodami vidutinį (1000 elementų) ir c) rikiuodami didelį (50000 elementų) masyvą. Vietoje vienos rezultatų lentelės, pateikite tris lenteles, po vieną kiekvienam iš skirtingų dydžių.

Užduotis 5.
Kiekvienam rikiavimo algoritmui sukurkite atskirą *object* failą (.o). Visus *object* failus įkelkite į savo sukurtas (vieną statinę, kitą dinaminę) bibliotekas.