LABORATORIYA ISHI №5

MAVZU: ALGORITMLAR SAMARADORLIGINI BAHOLASH. SARALASH ALGORITMLARIGA DOIR MISOLLAR

Ishning maqsadi: Talabalarda algoritmlar samaradorligini baholash. saralash algoritmlariga algoritmlarni tahlil qilish ko‘nikma va malakalarini shakllantirish.

Nazariy qism:

Massivlarni saralash tushunchasi va saralash algoritmlarining baholanishi.

Saralash - tartiblash (Sorting Algorithms) deb, berilgan obyektlar ketma-ketligini ma’lum mantiqiy tartibda qayta joylashtirish jarayoniga aytiladi. Saralash bir necha ko‘rsatkichlarga bog‘liq bo‘lishi mumkin.

Umuman olganda saralashning maqsadi berilgan obyektlar to‘plamini aniq bir tartibda guruhlab chiqish jarayoni tushuniladi. Saralashning maqsadi keyinchalik, saralangan to‘plamni qidirilayotgan elementini topishdan iborat. Bu qariyb universal, fundamental jarayon. Biz bu jarayon bilan har kuni uchrashamiz telefon daftaridagi saralash, kitoblar sarlavhasida, kutubxonalarda, lug‘atlarda, pochtada va h.k. Hatto yosh bolalar ham o‘z narsalarini tartiblashga o‘rganadi.

Saralashning juda ko‘p usullari mavjud. Ular turli to‘plamlar uchun turlicha bo‘lishi mumkin.

Massivni saralash uchun ishlatiladigan usul unga berilgan xotirani ixcham holda ishlatish lozim. Boshqacha qilib aytganda, saralanayotgan massiv xuddi shu massivni o‘zida amalga oshirilishi lozim.

Saralash algoritmlari kam mashina vaqtini talab qilishi lozim. Eng yaxshi tez algoritmlar $n*\log n$ tartibidagi saralashlarni talab etadi.

Bu yerda ikkita saralashni farqlash lozim. Ichki va tashqi saralash. Ichki saralash deganda, massivlarni saralashni, tashqi saralash deganda esa, fayl elementlarini saralashni tushunamiz.

Faraz qilaylik, bizga $a_1, a_2, \ldots, a_n$ elementlar berilgan bo‘lsin, u holda massivni saralash deganda, uni elementlarini o‘rinlariga almashtirish tushuniladi.

$a_{k_1}, a_{k_2}, \ldots, a_{k_n}$ bu yerda, quyidagi tartiblashtirilgan funkstiya bajariladi (1):

$$f\left(a_{k_1}\right) \le f\left(a_{k_2}\right) \le \ldots \le f\left(a_{k_n}\right) \quad (5.1)$$

Saralash algoritmlari bajarilish tezligida xotirani effektiv ishlatilishi bo‘yicha baholash.

Vaqt – bu algoritmni tezligini xarakterlovchi asosiy parametr. Bu albatta hisoblash murakkabligi bilan bog‘liq. Vaqt bo‘yicha algoritmlarni uchta turga bo‘lish mumkin: yomon, o‘rta va yaxshi.

- **O(n*n)** – yomon.
- **O(n log n)** -o‘rta.
- **O(n)** - yaxshi.

Bu yerda n saralanadigan elementlar soni bo‘lib, u avvaldan berilgan bo‘lishi kerak.

Xotira – ba’zi bir algoritmlar saralashda qo‘shimcha xotira talab etiladi.

Odatda bunday algoritmlar O(logn) xotirani talab etiladi. Ba’zi birlari saralashda qo‘shimcha xotira talab etmaydi. Bunday algoritmlar o‘z o‘rnida (joyida) saralash deb ataladi.

Massivlarni saralash xossalari va ularning sinflari:

Turg‘unlilik (stability) – turg‘un saralash teng elementlarning o‘rinlarini o‘zaro almashtirmaydi. Agar ular bir nechta maydonlardan tashkil topgan bo‘lsa, bunday xususiyatlar juda ham foydali bo‘lishi mumkin.

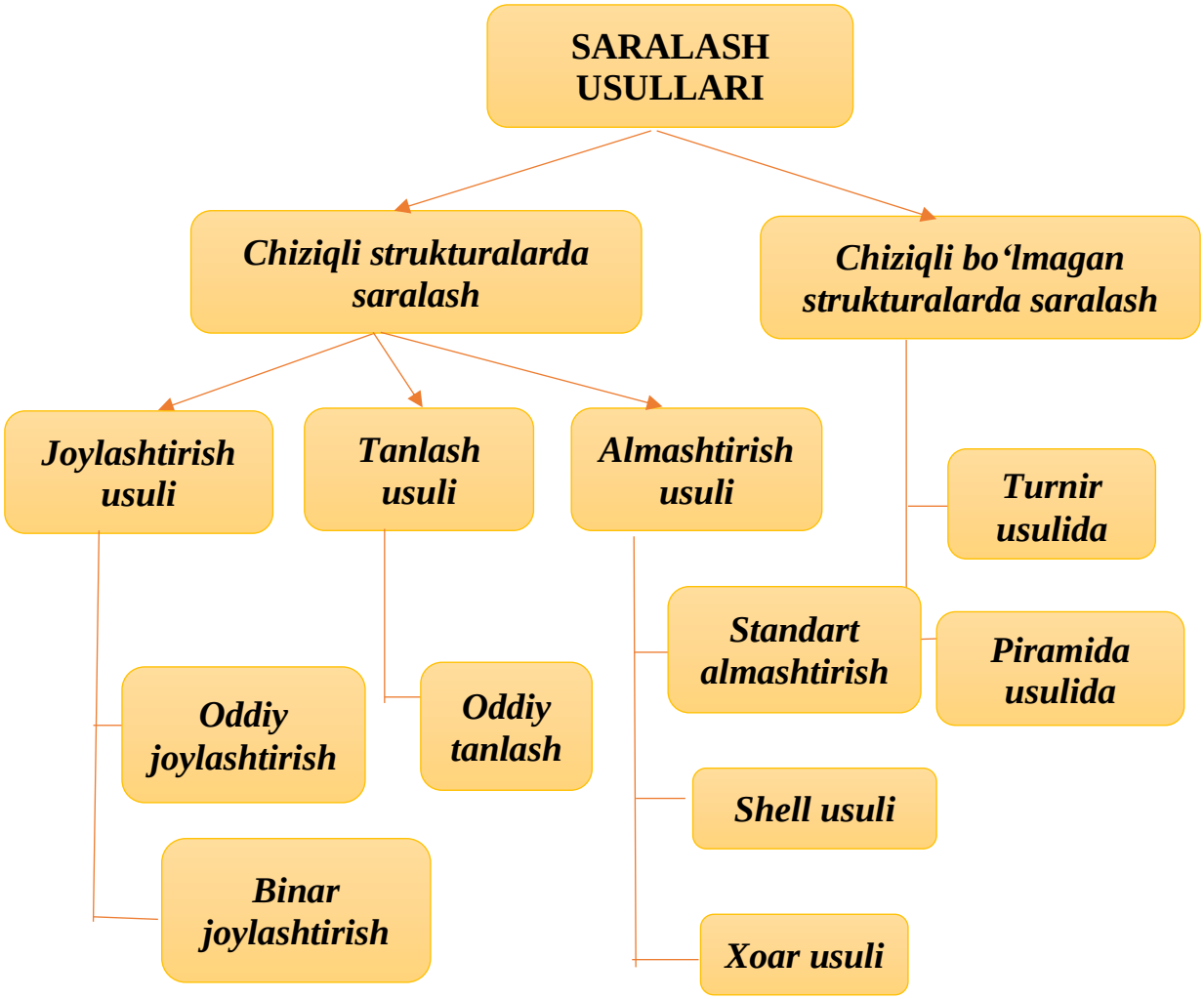
Tabiiy xulqlilik – algoritm o‘zini tabiiy holdagidek tutadi. Agar kiritiladigan ketma-ketlikdagi bu xarakteristikani xisobga olsa va yaxshi ishlasa u tabiiy xulqli deyiladi. Algoritmning asosiy xossalaridan biri unga qo‘llanish sferasi(sohasi) bilan bog‘liq.

Asosan ikkita turdagi saralash mavjud:

Ichki saralash. Boshqacha qilib aytganda, bunday saralash massivlarda saralashni amalga oshiradi. Bunda keltirilgan ketma – ketlik operativ xotirada joylashadi va bunda ixtiyoriy yacheykaga ruxsatli kirish mavjud. Asosan bu erda o‘z joyida saralash amalga oshiriladi.

Tashqi saralash. Bu yerda fayllar ustida saralash amalga oshiriladi. Albatta, bunday saralashda vaqt ko‘p ketadi, lekin o‘lcham jihatdan katta ketma-ketliklarni saralash mumkin.

Saralash algoritmlarini quyidagi guruhlarga ajratish mumkin [2] (5.1-rasm):



Saralash bo‘yicha algoritmlar quyidagi sinflarga bo‘linadi:

– Qo‘shimcha xotiraga ehtiyoj yoki unga ehtiyojni yo‘qligi.

– Solishtirish amalidan tashqariga chiquvchi ma’lumotlar tarkibi haqidagi bilimlarga ehtiyoj bor yoki unga ehtiyojni yo‘qligi.

Turg‘un saralash algoritmlari:

– Tanlash saralash (Selection sort) – algoritm murakkabligi $O(n^2)$

– Ko‘pikli saralash (Bubble sort) - algoritm murakkabligi $O(n^2)$

– Aralashtirish saralashi (Sheyker, Cocktail sort, bidirectional bubble sort) - algoritm murakkabligi $O(n^2)$

– O‘rniga qo‘yish saralashi (Insertion sort) - algoritm murakkabligi $O(n^2)$

– Qo‘shilish saralashi (Merge sort) - algoritm murakkabligi O(n log n)

– Ikkilik daraxti yordamida saralash (Tree sort) - algoritm murakkabligi O(n log n) qo‘shimcha O(n) xotira talab etadi.

– Timsort saralashi (Timsort) - algoritm murakkabligi O(n log n) qo‘shimcha O(n) xotira talab etadi.

– Sanash orqali saralash (Counting sort) - algoritm murakkabligi O(n+k) qo‘shimcha O(n) xotira talab etadi.

– Blokli saralash (Savatli saralash, Bucket sort) - algoritm murakkabligi O(n) qo‘shimcha O(k) xotira talab etadi.

Turg‘un bo‘lmagan saralash algoritmlari:

– Tanlash orqali saralash (Selection sort) algoritm murakkabligi $O(n^2)$.

– Shell saralash (Shell sort) algoritm murakkabligi $O(n \log^2 n)$.

– Tarash orqali saralash (Comb sort) algoritm murakkabligi O(n logn)

– Suzuvchi saralash (Smooth sort) algoritm murakkabligi O(n logn)

– Tez saralash (Quick sort) algoritm murakkabligi O(nlogn)

– Intro sort - algoritm murakkabligi O(nlogn)

– Patience sorting - algoritm murakkabligi O(nlogn)

– Stooge sort - algoritm murakkabligi $O(n^2)$

– Razryadli saralash. algoritm murakkabligi O(n+k). O(k) qo‘shimcha xotira talab etiladi.

Amaliy bo‘lmagan saralash algoritmlari:

– Bogosort - algoritm murakkabligi O(n n!).

– O‘rinlashtirish saralash - algoritm murakkabligi O(n n!).

– Ma’nosiz saralash (Stupid sort) - algoritm murakkabligi $O(n^3)$.

– Bead sort - Algoritm murakkabligi O(n) yoki O(sqrt(n)). Maxsus apparat ta’minoti talab etiladi.

LABORATORIYA ISHINI BAJARISH UCHUN NAMUNA:

Saralash asosan ro‘yxat, massiv elementlarida amalga oshiriladi.

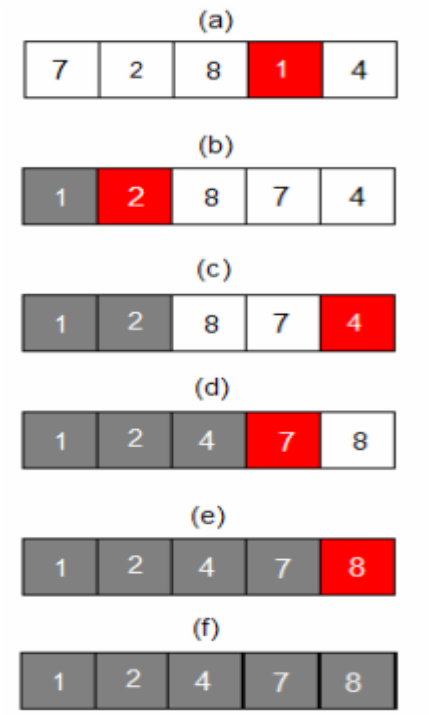
1-misol. Masalan sizning sinfingizda 5 ta o'quvchi bor. Ularni familiyasini alifbo tartibida saralash mumkin.

Sonlar berilishi: 23, 54, 3, 22, 1, 45;

Eng kattasini boshiga o‘tkazamiz: 23, 3, 22, 1, 45, 54; (54 soni har bir son bilan solishtirilib eng katta ekani aniqlandi, 45 esa o‘z o‘rnida). Shu tartibni davom ettiramiz: 3, 22, 1, 23, 45, 54; (23 undan keyinda turuvchi eng katta son) Yuqoridagi amalni yana davom ettiramiz: 3, 1, 22, 23, 45, 54; (22 esa davomchi) Oxirgi marta almashtirishimiz quyidagi natijani beradi: 1, 3, 22, 23, 45, 54; (1 eng kichigi) Saralangan tartib quyidagi holatga keldi: 1, 3, 22, 23, 45, 54.

Selection sort – Tanlab saralash bu – oddiy tartiblash algoritmidir. Ushbu tartiblash algoritmi o‘z joyida taqqoslashga asoslangan algoritm bo‘lib, unda ro‘yxat ikki qismga bo‘linadi, tartiblangan qism chap tomonda va tartiblanmagan qism o‘ng tomonda.

2-misol. Dastlab, tartiblangan qism bo‘sh, tartiblanmagan qismi esa butun ro‘yxatdir.



Eng kichik element tartiblanmagan massivdan tanlanadi va eng chap element bilan almashtiriladi va bu element tartiblangan massivning bir qismiga aylanadi. Bu jarayon tartiblanmagan massiv chegarasini bitta element bilan o‘ngga siljitishda davom etadi.

Ushbu algoritm katta ma’lumotlar to‘plamlari uchun mos emas, chunki uning o‘rtacha va eng yomon holatlari murakkabligi $O(n^2)$, bu yerda n – elementlar soni.

3-misol. Quyidagi massivni ko‘rib chiqamiz: {64, 25, 12, 22, 11}

Birinchi o‘tish: Saralangan massivdagi birinchi o‘rin uchun butun massiv 0 dan 4 gacha bo‘lgan indeksdan ketma-ket o‘tkaziladi. Hozirgi vaqtda 64 saqlanadigan birinchi pozitsiya, butun massivni aylanib o‘tgandan so‘ng, 11 eng past qiymat ekanligi bayon bo‘ladi.

64 25 12 22 11

Shunday qilib, 64 ni 11 bilan almashtiring. Bir iteratsiyadan so‘ng massivdagi eng kam qiymat bo‘lgan 11, tartiblangan ro‘yxatning birinchi pozitsiyasida paydo bo‘ladi.

11 25 12 22 64

Ikkinchi o‘tish: 25 mavjud bo‘lgan ikkinchi pozitsiya uchun massivning qolgan qismini yana ketma-ketlikda aylantiring.

11 25 12 22 64

Ketishdan so‘ng biz 12 massivdagi ikkinchi eng past qiymat ekanligini va u massivda ikkinchi o‘rinda paydo bo‘lishi kerakligini aniqladik, shuning uchun bu qiymatlarni almashtiring.

11 12 25 22 64

Uchinchi o‘tish: Endi, uchinchi o‘rin uchun, 25 mavjud bo‘lgan joyda yana massivning qolgan qismini aylanib o‘ting va massivdagi uchinchi eng kam qiymatni toping.

11 12 25 22 64

Ketish paytida 22 uchinchi eng kam qiymat bo‘lib chiqdi va u massivda uchinchi o‘rinda paydo bo‘lishi kerak, shuning uchun 22 ni uchinchi o‘rindagi element bilan almashtiring.

11 12 22 25 64

To‘rtinchi o‘tish: Xuddi shunday, to‘rtinchi pozitsiya uchun massivning qolgan qismini kesib o‘ting va massivdagi to‘rtinchi eng kichik elementni toping. 25 4-eng past qiymat bo‘lgani uchun u to‘rtinchi o‘rinni egallaydi.

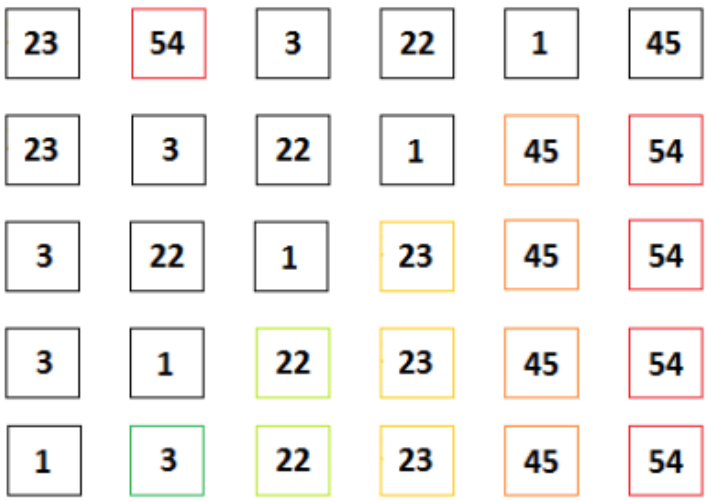
11 12 22 25 64

Beshinchi o‘tish: Nihoyat, massivda mavjud bo‘lgan eng katta qiymat avtomatik ravishda massivning oxirgi pozitsiyasiga joylashtiriladi Olingan massiv tartiblangan massivdir.

11 12 22 25 64

4-misol. Sonlar berilishi: 23, 54, 3, 22, 1, 45.

1. Eng kattasini boshiga o‘tkazamiz: 23, 3, 22, 1, 45, 54; (54 soni har bir son bilan solishtirilib eng katta ekani aniqlandi, 45 esa o‘z o‘rnida turibdi)
2. Shu tartibni davom ettiramiz: 3, 22, 1, 23, 45, 54; (23 undan keyinda turuvchi eng katta son)
3. Yuqoridagi amalni yana davom ettiramiz: 3, 1, 22, 23, 45, 54; (22 esa davomchi)
4. Oxirgi marta almashtirishimiz quyidagi natijani beradi: 1, 3, 22, 23, 45, 54; (1 eng kichigi)



LABORATORIYA ISHINI BAJARISH UCHUN TOPSHIRIQLAR:

1. $\boldsymbol{a1, a2, \dots, a20}$ ketma-ketlik berilgan. Ketma-ketlikning o‘sish tartibida toq o‘rinlarda turgan musbat elementlarni joylashtiring.

2. $\boldsymbol{a1, a2, \dots, a15}$ ketma-ketlik berilgan. Ketma-ketlikning kamayish tartibida nol bo‘lmagan elementlarini joylashtiring.

3. $\boldsymbol{x1, x2, \dots, x20}$ ketma-ketlik berilgan. Toq o‘rinda turgan elementlarni o‘sish tartibida joylashtiring, toqlarni esa kamayish tartibida joylashtiring.

4. $\boldsymbol{a1, a2, \dots, a15}$ ketma-ketlik berilgan. Uning absolyut qiymatlarini o‘sish tartibida tartiblash talab etiladi.

5. $\boldsymbol{x1, x2, \dots, x20}$ ketma-ketlik berilgan. Ketma-ketlikning manfiy elementlarini kamayish tartibida joylashtiring.

6. $\boldsymbol{a1, a2, \dots, a20}$ ketma-ketlik berilgan. Ketma-ketlikning musbat elementlarini kamayish tartibida joylashtiring.

7. $\boldsymbol{a1, a2, \dots, a15}$ ketma-ketlik berilgan. Ketma-ketlikning manfiy elementlarini o‘sish tartibida joylashtiring.

8. $\boldsymbol{a1, a2, \dots, a20}$ ketma-ketlik berilgan. Ketma-ketlikning toq elementlarini kamayish tartibida joylashtiring.

9. $\boldsymbol{x1, x2, \dots, x15}$ ketma-ketlik berilgan. Ketma-ketlikning juft musbat elementlarini o‘sish tartibida joylashtiring.

10. $\boldsymbol{x1, x2, \dots, x20}$ ketma-ketlik berilgan. Ketma-ketlikning 10 dan katta bo‘lgan elementlarini o‘sish tartibida joylashtiring.

11. $\boldsymbol{x1, x2, \dots, x20}$ ketma-ketlik berilgan. Ketma-ketlikning 10 dan kichik elementlarini kamayish tartibida joylashtiring.

12. $\boldsymbol{x1, x2, \dots, x20}$ ketma-ketlik berilgan. Ketma-ketlikning juft o‘rinda turgan juft elementlarini o‘sish tartibida joylashtiring.

13. Butun sonlar massivi berilgan. Klaviaturadan sonini kiritish.

14. Barcha elementlar turli xil bo‘lgan holda, massivning eng katta elementi va uning tartib raqamini toping.

15. $\boldsymbol{a1, a2, \dots, a15}$ ketma-ketlik berilgan. Ketma-ketlikning manfiy elementlarini kamayish tartibida joylashtiring.