BOG'LANGAN RO'YXATLAR. SDE INTERVYULARIDA SO'RALGAN BOG'LANGAN RO'YXAT MA'LUMOTLAR TUZILMASIDAGI ENG YAXSHI 5 TA MUAMMO VA YECHIMLAR

Hamroyeva Ozoda

BuxDU Fizika-Matematika va Axborot texnologiyalari fakulteti Kompyuter ilmlari va dasturlash texnologiyasi(turlar bo'yicha) yo'nalishi 3-bosqich talabasi

E-mail: *ozodahamroyevaa@gmail.com*

Annotatsiya. Bog'langan ro'yxatlar — kompyuter fanlari va dasturlashda asosiy ma'lumotlar tuzilmalaridan biridir. Ushbu maqolada dasturiy ta'minot muhandisi (SDE) lavozimiga bo'ladigan intervyularda tez-tez uchraydigan bog'langan ro'yxatga oid beshta asosiy muammo tahlil qilinadi. Har bir masala uchun samarali algoritmik yechim, tahlil va zamonaviy yondashuvlar keltiriladi. Bu maqola talabalar, dasturchilar va ilmiy izlanishlar olib borayotganlar uchun foydalidir.

Kalit so'zlar: bog'langan ro'yxat, ma'lumotlar tuzilmasi, algoritm, SDE intervyu, rekursiya, iterator, sikl aniqlash

Аннотация. Связанные списки являются одной из фундаментальных структур данных в информатике и программировании. В этой статье анализируются пять ключевых проблем связанных списков, которые часто встречаются на собеседованиях инженеров по разработке программного обеспечения (SDE). Для каждой проблемы представлены эффективные алгоритмические решения, анализ и современные подходы. Эта статья будет полезна студентам, программистам и исследователям.

Ключевые слова: связанный список, структура данных, алгоритм, интервью SDE, рекурсия, итератор, обнаружение цикла

Abstract. Linked lists are one of the fundamental data structures in computer science and programming. This article analyzes five key linked list problems that are frequently encountered in software development engineer (SDE) interviews. For each problem, effective algorithmic solutions, analysis, and modern approaches are presented. This article is useful for students, programmers, and researchers.

Keywords: linked list, data structure, algorithm, SDE interview, recursion, iterator, cycle detection

Kirish

"The advance of technology is based on making it fit in so that you don't really even notice it, so it's part of everyday life." — Bill Gates. Tarjimasi: "Texnologiyaning rivojlanishi uni siz hatto sezmasligingiz uchun moslashtirishga asoslangan, shuning uchun u kundalik hayotning bir qismidir."

ISSN 3030-3907

Zamonaviy raqamli davrda axborotni tezkor va samarali tarzda qayta ishlash har qachongidan muhim bo‘lib bormoqda. Har kuni milliardlab foydalanuvchi ishlatadigan veb-ilovalar, mobil tizimlar, sun’iy intellekt algoritmlari va real vaqtli dasturlar – barchasi kuchli va optimallashtirilgan ma’lumotlar tuzilmalariga tayanadi. Ushbu tuzilmalar orasida bog‘langan ro‘yxatlar (Linked Lists) — o‘zining soddaligi, moslashuvchanligi va samarali ishlov berish imkoniyatlari bilan ajralib turadi. Bog‘langan ro‘yxatlar dasturchilar uchun oddiy ko‘ringandek tuyulsa-da, ularning murakkab muammolarni yechishda tutgan o‘rni beqiyosdir. Ayniqsa, dasturiy ta'minot muhandisi (SDE) bo‘lish istagidagi nomzodlar uchun bu tuzilma algoritmik fikrlashni sinovdan o‘tkazadigan asosiy vositadir. Bugungi yirik texnologik kompaniyalar — Google, Meta, Amazon va boshqa yetakchi IT gigantlari intervyularida bog‘langan ro‘yxatga oid topshiriqlar orqali nomzodning chuqur bilimi va amaliy yondashuvini sinab ko‘rishadi. Ushbu maqolada bog‘langan ro‘yxat ma’lumotlar tuzilmasiga oid SDE intervyularida eng ko‘p uchraydigan beshta muammo va ularning algoritmik yechimlari tahlil qilinadi. Bu nafaqat nazariy, balki amaliy ko‘nikmalarni rivojlantirishda ham muhim o‘rin tutadi.

ADABIYOTLAR TAHLILI

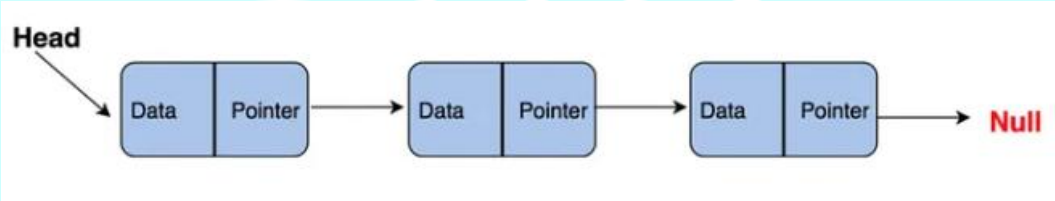
Nick Parlantening “Linked List Basics” qo’llanmasida tushuntirishlar, chizmalar, namuna kodi va mashqlar aralashmasi bilan bog'langan ro'yxatlarni yaratishning asosiy tuzilmalari va usullarini taqdim etilgan. Agar o’rganuvchi bog'langan ro'yxatlarni tushunmoqchi bo'lsa yoki ko'rsatgichni talab qiladigan kodning real, qo'llaniladigan misolini ko'rmoqchi bo'lsa, ushbu material foydali hisoblanadi [1]. Raaghav Ravishankar, Sandeep Kulkarni, Sathya Peri, Gokarna Sharmalarning “Distributing Context-Aware Shared Memory Data Structures: A Case Study on Singly-Linked Lists” maqolasida kontekstdan xabardor umumiy xotira ma'lumotlar tuzilmasini bir nechta mashinalarda ishlaydigan taqsimlangan ma'lumotlar tuzilmasi sifatida amalga oshirish uchun qismlarga ajratish o’rgatilgan. Kontekstdan xabardor ma'lumotlar tuzilmalari deganda, biz operatsiya natijasi nafaqat umumiy ma'lumotlarning qiymatiga , balki xuddi shu mijoz tomonidan bajarilgan oldingi operatsiyalarga ham bog'liqligi nazarda tutilgan. Taqsimlangan ma'lumotlar tuzilmalarini loyihalash bo'yicha katta ishlar olib borilayotgan bo'lsa-da, tarqatilgan kontekstdan xabardor ma'lumotlar tuzilmalarini loyihalashga unchalik e'tibor berilmaydi. Kontekstdan xabardor ma'lumotlar strukturasini misol tariqasida o'rganish sifatida maqolada umumiy xotira kontekstidan xabardor blokirovkasiz yakka-bog'langan ro'yxatni tarqatilgan blokirovkasiz kontekstdan xabardor yakka-bog'langan ro'yxatga aylantirilgan . Unda bog'langan ro'yxatning mijoz tomonidan ko'rinadigan operatsiyalarining ushbu xususiyatlarini saqlaydigan ikkita protokol taqdim etilgan. Ikkala protokolda

ISSN 3030-3907

ham mijozga ko'rinadigan operatsiyalar blokirovkasiz qoladi. Bundan tashqari, maqoladagi transformatsiya yondashuvi hech qanday apparatning eski versiyalaridan foydalanilmaydi(bitta so'zni solishtirish va almashtirish operatsiyasidan tashqari). Shuni ta'kidlaymizki, maqoladagi o'zgartirish umumiy bo'lib, blokirovkasiz boshqa kontekstli ma'lumotlar tuzilmalari uchun ishlatilishi mumkin, ular alohida bog'langan ro'yxatlardan tuzilishi mumkin [2].

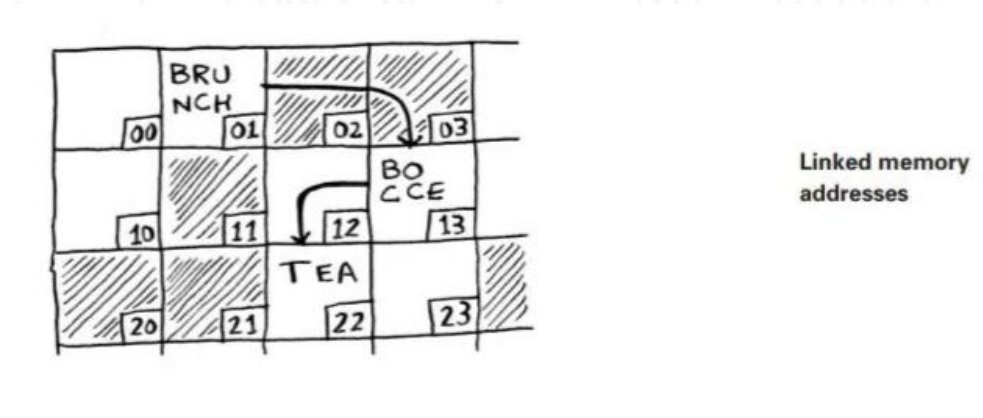
TADQIQOT METODOLOGIYASI

Dasturlash asoslari, programmalash asoslari va boshqa shu kabi fanlarda keng qo’llaniladigan mavzu ya’ni LinkedList (Bog’langan ro’yxat)- mavzusini ko’rib chiqamiz. Tabiiyki hammada eng birinchi bo’lib Linkedlist nima u bizga nimaga kerak degan savol tug’iladi. Bu savolga javobni mavzu davomida tushuntirishga harakat qilamiz. Linked list ham chiziqli ma’lumot tuzilmasi bo’lib, ishlatilishiga ko’ra arraydan keyingi o’ringa turadi. Bir qarashda linked list arrayga o’xshab ketadi. Lekin, xotirada elementlar joylashishi, ichki tuzilishi va asosiy operatsiyalar qanday tarzda amal oshirilishiga qarab arraydan anchagina farq qiladi. Linked list — bu tugunlardan iborat chiziqli tuzilma hisoblanadi, har bir tugun o’zida ma’lumot (element)ni va keyingi (ba’zida oldingi ham) tugun adressini ko’rsatuvchi ko’rsatkichni saqlaydi.



1-rasm. LinkedList strukturasi.

Bunda biz boshlang’ich tugun adressini bilgan holda qolgan tugunlarga o’tishimiz mumkin bo’ladi. Linked listda ma’lumotlar xotirada ketma-ket joylashmaydi, aksincha, xotiraning qayeri bo’sh bo’lsa shu yerga ma’lumot yozib ketilaveradi.



2-rasm. LinkedListning xotiradan joy olish strukturasi

LinkedList massiv kabi chiziqli ma'lumotlar strukturasi(MS) bo'lib, lekin undagi elementlarning o'rni (indeksi) mavjud emas. Elementlar o'zaro

ISSN 3030-3907

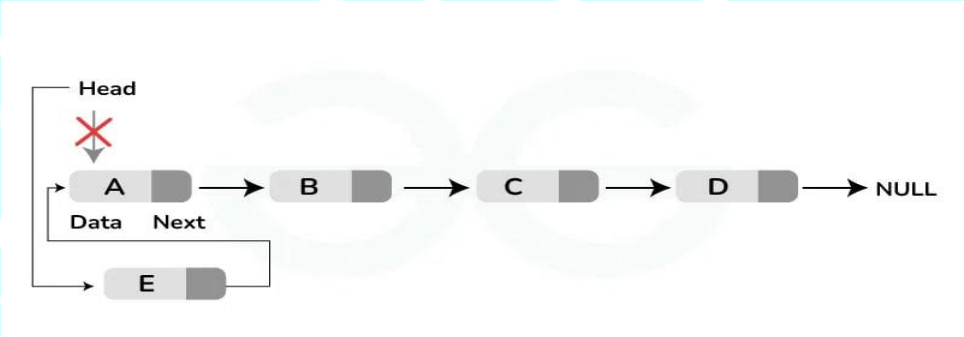
ko'rsatkich(yo'llanma) yoki "link"lar bilan bo'glanadi. So'zni Linkedlist tashkil etuvchisi sodda ma’lumotlar strukturasi "node"dan boshlaymiz. Tushunish oson bo’lishu uchun xalqa deb olishimiz mumkin. Ushbu MS ko'plab boshqa katta MS lar uchun qurilishi materiali vazifasini bajarib beradi. Xalqa o'zida qandaydir obyektni va keyingi xalqaga yo'llanmani saqlaydi. Xalqada saqlanayotgan elementlarning tipi har doim bir bo'lishi lozim. Bir necha xalqalarni birlashtirgan holda zanjirni hosil qilamiz. Demak, zanjir bu bir dan ortiq xalqalarning to'plami bo'lib, ular o'zaro faqat birgina link bilan bog'lanadi.

Bog'langan ro'yxat - ketma-ketlikdagi tugunlar guruhidan iborat bo'lgan juda tez-tez ishlatiladigan chiziqli ma'lumotlar tuzilmasi. Har bir tugun o'z ma'lumotlariga va keyingi tugunning manziliga ega, shuning uchun zanjirga o'xshash tuzilma hosil qiladi. Bog'langan ro'yxatlar daraxtlar va grafiklarni yaratish uchun ishlatiladi. Bog'langan ro'yxatlarning afzalliklari: Ular dinamik xarakterga ega bo'lib, kerak bo'lganda xotirani ajratadi. Qo'shish va o'chirish operatsiyalari osongina amalga oshirilishi mumkin. Stacks va navbatlar osongina bajarilishi mumkin. Bog'langan ro'yxat kirish vaqtini qisqartiradi. Bog'langan ro'yxatlarning kamchiliklari: Xotira behuda sarflanadi, chunki ko'rsatkichlar saqlash uchun qo'shimcha xotira talab qiladi. Hech qanday elementga tasodifiy kirish mumkin emas; U har bir tugunga ketma-ket kirishi kerak. Bog'langan ro'yxatda teskari o'tish qiyin. Bog'langan ro'yxat - bu chiziqli ma'lumotlar strukturasi bo'lib, unda elementlar qo'shni xotira joylarida saqlanmaydi. Bog'langan ro'yxatdagi elementlar ko'rsatkichlar yordamida bog'lanadi. Oddiy so'z bilan aytganda, bog'langan ro'yxat har bir tugun ma'lumotlar maydonini va ro'yxatdagi keyingi tugunga havolani o'z ichiga olgan tugunlardan iborat. Bog'langan ro'yxatlardan foydalanish: Bog'langan ro'yxatlar steklarni, navbatlarni, grafiklarni va boshqalarni amalga oshirish uchun ishlatiladi. Bog'langan ro'yxatlar ro'yxatning boshiga va oxiriga elementlarni kiritish imkonini beradi. Bog'langan ro'yxatlarda biz o'lchamni oldindan bilishimiz shart emas. Bog'langan ro'yxatlar turlari: birinchisi bu yagona bog'langan ro'yxat, ikkinchisi ikki marta bog'langan ro'yxat va uchinchisi doiraviy bog'langan ro'yxat.

- Yagona bog'langan ro'yxat

U elementlar to'plamini saqlash usulidir. Massiv kabi, bu belgilar yoki butun sonlar bo'lishi mumkin. Bog'langan ro'yxatdagi har bir element tugun sifatida saqlanadi. Tugun - bu ikkita kichik element yoki qismlarning to'plami. Elementni saqlaydigan ma'lumotlar qismi va keyingi tugunga havolani saqlaydigan keyingi qism. Bog'langan ro'yxat shunday ko'plab tugunlar bir-biriga zanjirda ulanganda hosil bo'ladi. Har bir tugun tartibda mavjud bo'lgan keyingi tugunga ishora qiladi. Birinchi tugun har doim ro'yxatni aylanib o'tish uchun havola sifatida ishlatiladi va HEAD deb ataladi. Oxirgi tugun NULL ga ishora qiladi.

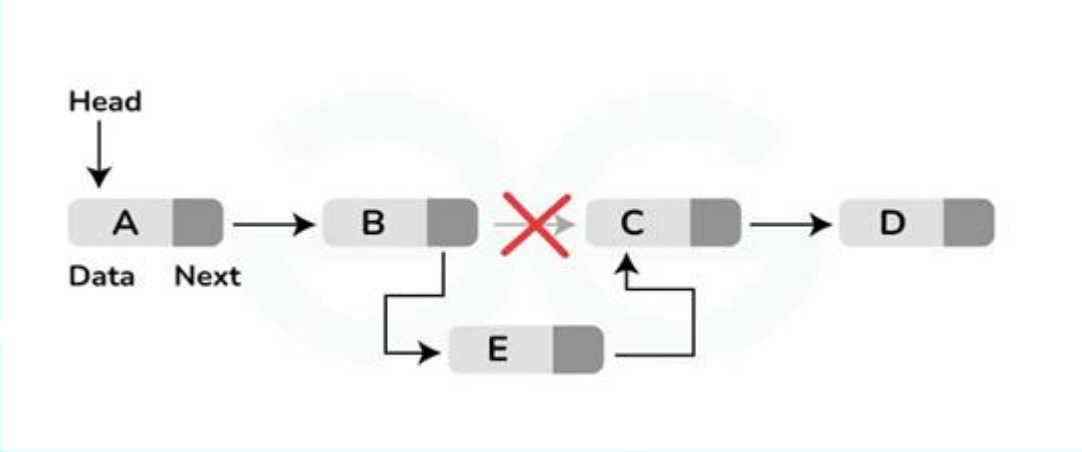
ISSN 3030-3907



3-rasm. Yagona bog'langan LinkedList

➢ Ikki marta bog'langan ro'yxat

Ikki marta bog'langan ro'yxat - bu bog'langan ro'yxatning bir turi bo'lib, unda har bir tugun o'z ma'lumotlarini saqlashdan tashqari ikkita havolaga ega. Birinchi havola ro'yxatdagi oldingi tugunga, ikkinchi havola esa ro'yxatdagi keyingi tugunga ishora qiladi. Ro'yxatning birinchi tugunida NULL ga ishora qiluvchi oldingi havola mavjud, xuddi ro'yxatning oxirgi tugunida NULLga ishora qiluvchi keyingi tugun mavjud. Ushbu ikkita havola bizga ro'yxatni orqaga va oldinga aylantirishga yordam beradi. Ammo qo'shimcha havolani saqlash qo'shimcha joy talab qiladi.



4-rasm. Ikki marta bog'langan ro'yxatdagi operatsiyalar

MUHOKAMA VA NATIJALAR

Bog‘langan ro‘yxat (linked list) kabi ma’lumotlar tuzilmalari oddiy ko‘ringan bo‘lishiga qaramay, ular bilan ishlash jarayonida yuzaga keladigan muammolar ko‘p jihatdan algoritmik aniqlik, samaradorlik va tizim resurslaridan oqilona foydalanishni talab qiladi. Ushbu maqolada ko‘rib chiqilgan beshta muammo real dasturlash holatlarida tez-tez uchraydi va har biri amaliy yechim bilan izohlangan.

Birinchi muammo — qidiruv jarayonining vaqt murakkabligi — bu bog‘langan ro‘yxatlarda elementga indeks bo‘yicha murojaat qilish imkonining yo‘qligi bilan bog‘liq. Bunday holatda hash jadval (hashtable) yoki hash xarita (hashmap) kabi qo‘shimcha tuzilmalar orqali ma’lumotlarga tezkor kirishni (O(1))

ISSN 3030-3907

ta’minlash mumkin. Bu yondashuv tezkorlikni oshirish bilan birga, xotira sarfini ham oshiradi, bu esa amaliy muvozanatni talab qiladi.

```
class Node:
    def __init__(self, value):
        self.value = value
        self.next = None
class IndexedLinkedList:
    def __init__(self):
        self.head = None
        self.index_map = {}  # O(1) qidiruv uchun
    def append(self, value):
        new_node = Node(value)
        if not self.head:
            self.head = new_node
        else:
            temp = self.head
            while temp.next:
                temp = temp.next
            temp.next = new_node
        self.index_map[value] = new_node
    def search(self, key):
        return self.index_map.get(key, None)
# Amaliy misol
ll = IndexedLinkedList()
ll.append(10)
ll.append(20)
ll.append(30)
print(ll.search(20).value)  # O(1) qidiruv
```

Ikkinchi muammo — nolga bo‘lish (division by zero) xatoliklari — dasturiy xatolarning eng keng tarqalgan va og‘ir oqibatlarga olib keluvchi holatlaridan biridir. Ushbu muammoning yechimi sifatida har qanday bo‘lish amali bajarilishidan oldin shartli tekshiruvlar (if-statements) yordamida qiymatning nolga teng emasligini aniqlash tavsiya etiladi. Bu yondashuv dastur barqarorligini ta’minlaydi.

```
def safe_divide_list(head):
    current = head
    while current:
        if current.value != 0:
            print(100 / current.value)
```


ISSN 3030-3907

```
            else:
                print("0 ga bo'lish mumkin emas")
            current = current.next
    # Amaliy misol
    node1 = Node(5)
    node2 = Node(0)
    node3 = Node(10)
    node1.next = node2
    node2.next = node3
    safe_divide_list(node1)
```

Uchinchi muammo — ma’lumotlar tuzilmasining samaradorligini oshirish — katta hajmdagi ma’lumotlarni ishlov berishda muhim rol o‘ynaydi. AVL daraxti va Red-Black daraxtlari kabi balanslangan daraxt tuzilmalaridan foydalanish orqali ma’lumotlarni kiritish, o‘chirish va qidirish operatsiyalarining vaqt murakkabligini O(log n) ga tushirish mumkin.

```
    def insert_sorted(head, value):
        new_node = Node(value)
        if not head or value < head.value:
            new_node.next = head
            return new_node
        current = head
        while current.next and current.next.value < value:
            current = current.next
        new_node.next = current.next
        current.next = new_node
        return head
    # Amaliy misol
    head = None
    for val in [30, 10, 20, 5]:
        head = insert_sorted(head, val)
    # Chiqarish
    temp = head
    while temp:
        print(temp.value, end=" -> ")
        temp = temp.next
```

To‘rtinchi muammo — ma’lumot hajmining chegarasini bilish zarurati — algoritmik optimallashtirishda muhim ahamiyatga ega. Shu bois, QuickSort, InsertionSort kabi tartiblash algoritmlarining mos variantlarini tanlash ma’lumotlar

ISSN 3030-3907

hajmi va turiga qarab amalga oshiriladi. Bu esa dastur samaradorligini bevosita oshiradi.

```
def merge_sort(head):
    if not head or not head.next:
        return head
    # Markazni topish
    slow, fast = head, head.next
    while fast and fast.next:
        slow = slow.next
        fast = fast.next.next
    mid = slow.next
    slow.next = None
    left = merge_sort(head)
    right = merge_sort(mid)
    return merge(left, right)
def merge(left, right):
    dummy = Node(0)
    tail = dummy
    while left and right:
        if left.value < right.value:
            tail.next, left = left, left.next
        else:
            tail.next, right = right, right.next
        tail = tail.next
    tail.next = left or right
    return dummy.next
```

Beshinchi muammo — o‘zgarmas (immutable) tuzilmalarni yaratish — bu ko‘p oqimli (multithreaded) yoki xavfsiz dasturlash muhitida dolzarbdir. Ko‘rsatkichlar (pointers) yoki const parametrlardan foydalanish orqali strukturaviy o‘zgarishlarning oldi olinadi va dastur yaxlitligi ta'minlanadi.

```
class ImmutableNode:
    def __init__(self, value, next_node=None):
        self._value = value
        self._next = next_node
    @property
    def value(self):
        return self._value
    @property
    def next(self):
```


ISSN 3030-3907

```
        return self._next
# Amaliy misol
n1 = ImmutableNode(1)
n2 = ImmutableNode(2, n1)
print(n2.value)
print(n2.next.value)
```

Muhokamadan ko‘rinib turibdiki, har bir muammo texnik jihatdan turli darajadagi yechimlarni talab qiladi. Ushbu yondashuvlar faqat bog‘langan ro‘yxatlarda emas, balki boshqa murakkab tuzilmalarda ham qo‘llanilishi mumkin. Shuningdek, bu algoritmik strategiyalar dasturiy ta’minot muhandislari uchun asosiy vositalardan biri hisoblanadi.

XULOSA VA TAKLIFLAR

Bog'langan ro'yxat - bu chiziqli ma'lumotlar strukturasi bo'lib, unda elementlar qo'shni xotira joylarida saqlanmaydi. Bog'langan ro'yxatdagi elementlar ko'rsatkichlar yordamida bog'lanadi. Oddiy so'z bilan aytganda, bog'langan ro'yxat har bir tugun ma'lumotlar maydonini va ro'yxatdagi keyingi tugunga havolani o'z ichiga olgan tugunlardan iborat. Bog'langan ro'yxatning foydalari juda ko'p bo'lishi mumkin. Bu, ma'lumotlarni tuzatish va ulashish uchun qulay usul bo'lib, sizning mijozlaringizga yangiliklar, chegirmalar va boshqa ma'lumotlarni yuborish imkoniyatini beradi. Shuningdek, sizning mijozlaringiz bilan doimiy aloqada bo'lishingizga yordam beradi va ularning istaklarini tushuntirib, ular uchun maxsus takliflarni yuborishga imkon beradi. Bu esa, mijozlaringizga eng sodda va samarali xizmatni taqdim etishga imkon beradi. Xullas qisqa qilib aytadigan bo’lsam , bog’langan ro’yxat yoki bog’lanishlar - bu juda ko’p joylarda qo’llaniladi . Linus Torvaldsning ham bog’lanishlar haqida ajoyib jumlasi bor edi : “ Yomon dasturchilar o’zlarining kodlari haqida qayg’urishadi, yaxshi dasturchilar esa ma’lumotlar tuzilmalari va ularning bog’lanishlari haqida bosh qotirishadi.” — Linus Torvald.

FOYDALANILGAN ADABIYOTLAR RO‘YXATI

[1]. Nick Parlante (2001) “Linked List Basics”

[2]. Raaghav Ravishankar, Sandeep Kulkarni, Sathya Peri, Gokarna Sharma (2025) “Distributing Context-Aware Shared Memory Data Structures: A Case Study on Singly-Linked Lists”

https://kompy.info/download/ozbekiston-respublikasi-raqamli-texnologiyalar-vazirligi-muham-v130.doc

https://www.texnoman.uz/post/linkedlist-kirish.html

https://janobmusayev.medium.com/array-va-linked-list-9617c85fd28d

ISSN 3030-3907