HSR (LONDON),
HOUGHTON STREET REVIEW JOURNAL
ISSN: 2058-5519|Impact factor-15.9
http://houghtonstreetreview.com/index.php/hsr

UO‘K: 004.42:378.147

TALABALARGA KOMPYUTER XOTIRASIDA MANFIY SONLARNING IKKILIK QO‘SHIMCHA KODLARDA SAQLANISH MEXANIZMINI TUSHUNTIRISH, BITLI MANTIQIY OPERATORLARNI O‘RGATISH HAMDA MURAKKAB BITLI IFODALARNI QO‘LDA VA C# TILIDA HISOBLASH KO‘NIKMALARINI SHAKLLANTIRISH METODIKASI

YUSUPJONOVA DILSHODA
Namangan davlat universiteti 1-bosqich talabasi
dilshodayusupjonova956@gmail.com
+998 77 818 29 66
doi:https://doi.org/10.5281/zenodo.24435578

Annotatsiya. Ushbu ilmiy maqolada oliy ta’lim muassasalarining axborot texnologiyalari yo‘nalishi talabalariga kompyuter xotirasida manfiy butun sonlarning ikkilik qo‘shimcha kodda (Two’s Complement) tasvirlanish mexanizmini o‘rgatishning samarali metodikasi yoritilgan. Shuningdek, bitli mantiqiy operatorlar (&, |, ^, ~, <<, >>) ishlash tamoyili, C# dasturlash tilidagi tip kengaytirish (integral promotion) va operatorlar ustuvorligi qoidalari tahlil qilingan. Murakkab bitli ifodalarni avval qog‘ozda bosqichma-bosqich qo‘lda hisoblash va so‘ngra C# muhitida dasturiy tekshirish orqali talabalarda algoritmik fikrlash va quyi darajadagi xotira arxitekturasini chuqur tushunish ko‘nikmalarini shakllantirish yondashuvi asoslab berilgan.
Kalit so‘zlar: ikkilik qo‘shimcha kod (Two’s Complement), manfiy sonlar, kompyuter xotirasi, bitli operatorlar, bitli siljish, C# dasturlash tili, integral promotion, o‘qitish metodikasi.

МЕТОДИКА ФОРМИРОВАНИЯ У СТУДЕНТОВ НАВЫКОВ ОБЪЯСНЕНИЯ МЕХАНИЗМА ХРАНЕНИЯ ОТРИЦАТЕЛЬНЫХ ЧИСЕЛ В ПАМЯТИ КОМПЬЮТЕРА В ДВОИЧНОМ ДОПОЛНИТЕЛЬНОМ КОДЕ, ОБУЧЕНИЯ ПОБИТОВЫМ ЛОГИЧЕСКИМ ОПЕРАТОРАМ, А ТАКЖЕ ВЫЧИСЛЕНИЯ СЛОЖНЫХ ПОБИТОВЫХ ВЫРАЖЕНИЙ ВРУЧНУЮ И НА ЯЗЫКЕ C#

ЮСУПЖОНОВА ДИЛШОДА
Студентка 1-го курса Наманганского государственного университета
dilshodayusupjonova956@gmail.com
+998 77 818 29
doi:https://doi.org/10.5281/zenodo.24435578

Аннотация. В данной научной статье освещена эффективная методика обучения студентов ИТ-направлений высших учебных заведений механизму представления

HSR (LONDON),
HOUGHTON STREET REVIEW JOURNAL
ISSN: 2058-5519|Impact factor-15.9
http://houghtonstreetreview.com/index.php/hsr

отрицательных целых чисел в памяти компьютера в дополнительном двоичном коде (Two's Complement). Проанализированы принципы работы побитовых логических операторов (&, |, ^, ~, <<, >>), правила целочисленного расширения типов (integral promotion) и приоритета операторов в языке программирования C#. Обоснован подход формирования у студентов навыков пошагового ручного вычисления сложных побитовых выражений с последующей программной верификацией в среде C#.
Ключевые слова: дополнительный код (Two's Complement), отрицательные числа, память компьютера, побитовые операторы, побитовый сдвиг, язык C#, методика обучения.

METHODOLOGY FOR DEVELOPING STUDENTS' SKILLS IN EXPLAINING THE MECHANISM OF STORING NEGATIVE NUMBERS IN COMPUTER MEMORY USING TWO'S COMPLEMENT BINARY REPRESENTATION, TEACHING BITWISE LOGICAL OPERATORS, AND DEVELOPING THE ABILITY TO CALCULATE COMPLEX BITWISE EXPRESSIONS MANUALLY AND USING THE C# PROGRAMMING LANGUAGE

YUSUPJONOVA DILSHODA
First-year student at Namangan State University
dilshodayusupjonova956@gmail.com
+998 77 818 29
doi:https://doi.org/10.5281/zenodo.24435578

Abstract. This scientific paper presents an effective pedagogical methodology for teaching undergraduate IT students the mechanism of storing negative integers in computer memory using Two's Complement representation. It examines the operational principles of bitwise logical and shift operators (&, |, ^, ~, <<, >>), alongside integral promotion rules and operator precedence in the C# programming language. A dual-stage learning framework—combining manual step-by-step tracing on paper with programmatic verification in C#—is substantiated to develop students' algorithmic thinking and low-level memory comprehension.
Keywords: Two's Complement, negative integers, computer memory, bitwise operators, bit shifting, C# programming language, integral promotion, teaching methodology.

1. KIRISH

Zamonaviy dasturiy ta'minot muhandisligi yuqori darajadagi abstraksiyalarga tayangan bo'lsa-da, tizimli dasturlash, kriptografiya, tarmoq protokollari, kompyuter grafikasi, o'rnatilgan tizimlar (embedded systems) va ma'lumotlarni siqish algoritmlarini yaratishda ma'lumotlarning xotirada bitlar darajasida qanday saqlanishini bilish hal qiluvchi ahamiyat kasb etadi. Oliy ta'lim muassasalarida «Dasturlash asoslari», «Kompyuter arxitekturasi» hamda «Algoritmlar va ma'lumotlar tuzilmalari»

HSR (LONDON),
HOUGHTON STREET REVIEW JOURNAL
ISSN: 2058-5519|Impact factor-15.9
http://houghtonstreetreview.com/index.php/hsr

fanlarini o‘qitish tajribasi shuni ko‘rsatadiki, talabalar o‘nlik sanoq sistemasidagi mantiqiy amallarni oson o‘zlashtirganlari holda, manfiy sonlarning ikkilik qo‘shimcha kodda (Two’s Complement) saqlanishi va ularga nisbatan bitli operatorlarning qo‘llanilishida jiddiy kognitiv qiyinchiliklarga duch keladilar.

Ko‘pchilik talabalar manfiy sonni kompyuter xotirasida oddiygina ishora biti (1) va sonning absolyut qiymati (to‘g‘ri kod) ko‘rinishida saqlanadi deb yanglish tasavvur qiladilar. Natijada, C# kabi qat’iy tiplashtirilgan zamonaviy dasturlash tillarida bitli inversiya (~), arifmetik o‘ngga siljish (>>) yoki aralash bitli ifodalar natijasini oldindan bashorat qila olmaydilar. Mazkur maqolaning maqsadi — talabalarga manfiy sonlarning ikkilik qo‘shimcha kodda saqlanishining matematik va apparat asoslarini sodda tushuntirish, bitli mantiqiy operatorlarning ishlash mexanizmini tizimlashtirish hamda murakkab bitli ifodalarni qo‘lda va C# dasturlash tilida hisoblash ko‘nikmalarini shakllantirishning bosqichma-bosqich metodikasini ishlab chiqishdan iborat.

2. MANFIY SONLARNING KOMPYUTER XOTIRASIDA SAQLANISH MEXANIZMI

Talabalarga kompyuter arxitekturasida manfiy sonlarni tasvirlash evolyutsiyasini tushuntirish uchun uchta asosiy kodlash usulini qiyosiy tahlil qilish maqsadga muvofiq: to‘g‘ri kod (Sign-Magnitude), teskari kod (One’s Complement) va qo‘shimcha kod (Two’s Complement). Protsessorning arifmetik-mantiqiy qurilmasi (ALU) ayirish amalini alohida sxema sifatida emas, balki qo‘shish sxemasi orqali bajarishi uchun aynan ikkilik qo‘shimcha kod standart sifatida qabul qilingan.

2.1. To‘g‘ri va teskari kodlarning kamchiliklari hamda qo‘shimcha kod ustunligi

8 bitli (1 baytli, C# tilidagi sbyte tipi) xotira katagi misolida qaraydigan bo‘lsak, eng chapdagi (7-indeksli, MSB — Most Significant Bit) bit ishora biti hisoblanadi: 0 — musbat sonni, 1 — manfiy sonni anglatadi. To‘g‘ri kodda +5 soni 00000101, -5 soni esa 10000101 ko‘rinishida yoziladi. Biroq to‘g‘ri va teskari kodlarda nol sonining ikki xil ko‘rinishi (+0 va -0) mavjud bo‘lib, bu mantiqiy taqqoslash va arifmetik qo‘shishda apparat xatoliklariga olib keladi. Ikkilik qo‘shimcha kod ushbu muammoni bartaraf etadi va N bitli xotira katagida [-2^(N-1); 2^(N-1) - 1] oraliqdagi sonlarni (masalan, 8 bitda -128 dan +127 gacha) yagona va uzluksiz tarzda ifodalaydi.

2.2. Manfiy sonni ikkilik qo‘shimcha kodga o‘tkazishning ikki usuli

Talabalarga manfiy -X sonining N bitli ikkilik qo‘shimcha kodini topishning ikki xil algoritmini o‘rgatish ularning mavzuni chuqur anglashiga yordam beradi:

Algoritmik yondashuv. 1-usul (Inversiya va birlik qo‘shish algoritmi): Avval musbat |X| sonining N bitli ikkilik ko‘rinishi yoziladi. So‘ngra barcha bitlar teskarisiga almashtiriladi (0 -> 1, 1 -> 0, ya’ni teskari kod olinadi) va hosil bo‘lgan natijaga 1 qo‘shiladi: -X = (~|X|) + 1.

Pozitsion vazn yondashuvi. 2-usul (Matematik vaznlar yondashuvi): N bitli qo‘shimcha kodda eng katta (ishora) bitning vazni musbat +2^(N-1) emas, balki manfiy -2^(N-1) ga teng deb qaraladi. Masalan, 8 bitli tizimda bitlarning o‘rin qiymatlari: -128, 64, 32, 16, 8, 4, 2, 1 ko‘rinishida bo‘ladi. Bu usul qo‘shimcha koddan o‘nlik sanoq sistemasiga tezkor o‘tishda nihoyatda qulaydir.

1-jadval. 8 bitli xotira katagida (sbyte) musbat va manfiy sonlarning tasvirlanishi

HSR (LONDON),
HOUGHTON STREET REVIEW JOURNAL
ISSN: 2058-5519|Impact factor-15.9
http://houghtonstreetreview.com/index.php/hsr

O‘nlik son	To‘g‘ri kod (Sign-Magnitude)	Teskari kod (One’s Complement)	Qo‘shimcha kod (Two’s Complement)	Pozitsion vaznlar yig‘indisi (-128, 64, 32, 16, 8, 4, 2, 1)
+45	0010 1101	0010 1101	0010 1101	32 + 8 + 4 + 1 = +45
0	0000 0000	0000 0000	0000 0000	0
-1	1000 0001	1111 1110	1111 1111	-128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = -1
-45	1010 1101	1101 0010	1101 0011	-128 + 64 + 16 + 2 + 1 = -45
-128	Mavjud emas	Mavjud emas	1000 0000	-128 + 0 = -128

Metodik eslatma (O‘qituvchilar uchun):
Talabalarga qo‘shimcha kodning to‘g‘riligini tekshirish uchun «odometr (hisoblagich) analogiyasi»ni ko‘rsatish samarali natija beradi: 0000 0000 holatidan 1 ayrilganda, barcha razryadlardan qarz olinib 1111 1111 (-1) hosil bo‘ladi. Aksincha, 1111 1111 ga 1 qo‘shilganda 9-bitga o‘tish (carry-out) tashlab yuboriladi va xotirada yana 0000 0000 (0) qoladi.

3. BITLI MANTIQIY VA SILJISH OPERATORLARINING ISHLASH TAMOYILI

C# dasturlash tilida butun sonlar ustida bitlar darajasida amal bajarish uchun oltita asosiy operator mavjud. Mantiqiy operatorlardan (&&, ||, !) farqli o‘laroq, bitli operatorlar sonning har bir mos razryadidagi (bitidagi) 0 va 1 qiymatlar ustida mustaqil ravishda amal bajaradi.

2-jadval. C# dasturlash tilidagi bitli operatorlar va ularning funksional xususiyatlari

Operator	Nomi	Ishlash qoidasi (Har bir bit uchun)	Amaliy qo‘llanilishi
&	Bitli VA (AND)	Ikkala bit 1 bo‘lsa 1, aks holda 0	Keraksiz bitlarni o‘chirish (bit maskalash)
|	Bitli YOKI (OR)	Kamida bitta bit 1 bo‘lsa 1, aks holda 0	Ma’lum bitlarni 1 qiymatga o‘rnatish (flag set)

HSR (LONDON),
HOUGHTON STREET REVIEW JOURNAL
ISSN: 2058-5519|Impact factor-15.9
http://houghtonstreetreview.com/index.php/hsr

^	Istisno YOKI (XOR)	Bitlar har xil bo‘lsa 1, bir xil bo‘lsa 0	Bitlarni inversiyalash (toggle), shifrlash
~	Bitli inkor (NOT)	Unar operator: 0 -> 1, 1 -> 0	Barcha bitlarni to‘ldirish; ~x = -x - 1
<<	Chapga siljish	Bitlarni k o‘rin chapga suradi, o‘ngga 0 yozadi	2^k ga ko‘paytirish (to‘lib ketish chegarasigacha)
>>	O‘ngga siljish	Bitlarni k o‘rin o‘ngga suradi; ishorali sonda chapga ishora bitini nusxalaydi	2^k ga butun bo‘lish (manfiy sonda -∞ tomonga yaxlitlaydi)

3.1. C# tilida ishorali va ishorasiz o‘ngga siljish hamda tip kengaytirish xususiyatlari

Talabalar C# tilida qo‘lda hisoblangan natija bilan dastur natijasini solishtirganda ko‘pincha ikkita muhim til xususiyatini e’tibordan chetda qoldiradilar:

Siljish turlari. 1) Arifmetik va mantiqiy o‘ngga siljish: Ishorali tiplar (sbyte, short, int, long) uchun >> operatori arifmetik siljishni bajaradi — ya’ni chap tomonda bo‘shagan o‘rinlarga 0 emas, balki eng chapdagi ishora biti (manfiy sonlar uchun 1) nusxalanadi. Ishorasiz o‘ngga siljish (chapga har doim 0 yozish) uchun esa C# 11 versiyasidan boshlab >>> operatori joriy etilgan.

Tip kengaytirish qoidasi. 2) Integral Promotion (Butun sonlarni kengaytirish): C# tilida byte, sbyte, short va ushort tiplari ustida bitli operatorlar (~, &, |, ^, <<, >>) bajarilganda, kompilyator ularni avtomatik ravishda 32 bitli int tipiga kengaytiradi. Masalan, sbyte a = 5; ifodasi uchun ~a amali 8 bitli sbyte emas, balki 32 bitli int (-6) qaytaradi va uni qayta sbyte ga o‘zlashtirish uchun aniq tip keltirish (sbyte)(~a) talab etiladi.

4. MURAKKAB BITLI IFODALARNI QO‘LDA VA C# TILIDA HISOBLASH KO‘NIKMALARINI SHAKLLANTIRISH

Talabalarda barqaror hisoblash ko‘nikmasini shakllantirish uchun «Qo‘lda modellashtirish -> Bosqichma-bosqich trassirovka -> Dasturiy verifikatsiya» uch bosqichli metodik modeli taklif etiladi. Bunda talaba avval operatorlar ustuvorligi (precedence) jadvaliga tayangan holda amallar ketma-ketligini aniqlashi shart: ~ (eng yuqori) -> <<, >> -> & -> ^ -> | (eng quyi).

4.1. Murakkab bitli ifodani qo‘lda yechish namunasi

Amaliy topshiriq. Masala: 8 bitli ishorali butun sonlar (sbyte) uchun a = -18, b = 25 va c = -7 berilgan bo‘lsin. Quyidagi murakkab bitli ifodaning qiymatini qo‘lda hisoblang va 8 bitli qo‘shimcha kodda tasvirlang:

R = ((~a & b) | (c << 2)) ^ (b >> 1)

HSR (LONDON),
HOUGHTON STREET REVIEW JOURNAL
ISSN: 2058-5519|Impact factor-15.9
http://houghtonstreetreview.com/index.php/hsr

Yechish algoritmi:

1-qadam. Boshlang‘ich sonlarni 8 bitli ikkilik qo‘shimcha kodga o‘tkazish:

• a = -18: +18 = 0001 0010 -> inversiya: 1110 1101 -> +1 qo‘shamiz: a = 1110 1110.

• b = +25: 16 + 8 + 1 -> b = 0001 1001.

• c = -7: +7 = 0000 0111 -> inversiya: 1111 1000 -> +1 qo‘shamiz: c = 1111 1001.

2-qadam. Qavslar ichidagi unar va siljish amallarini bajarish:

• ~a (bitli inversiya): ~(1110 1110) = 0001 0001 (o‘nlikda +17).

• c << 2 (2 bit chapga siljish): 1111 1001 << 2 = 1110 0100 (chapdagi ikkita 1 chiqib ketadi, o‘ngga ikkita 0 yoziladi; o‘nlikda -28).

• b >> 1 (1 bit o‘ngga siljish): 0001 1001 >> 1 = 0000 1100 (o‘nlikda +12).

3-qadam. Bitli VA (&) hamda bitli YOKI (|) amallarini bajarish:

• ~a & b: 0001 0001 & 0001 1001 = 0001 0001 (o‘nlikda +17).

• (~a & b) | (c << 2): 0001 0001 | 1110 0100 = 1111 0101 (o‘nlikda -11).

4-qadam. Yakuniy Istisno YOKI (^) amalini bajarish:

• R = 1111 0101 ^ 0000 1100 = 1111 1001.

5-qadam. Natijani o‘nlik sanoq sistemasiga o‘tkazish:

• 1111 1001 sonining ishora biti 1 (manfiy). Pozitsion vaznlar bo‘yicha: -128 + 64 + 32 + 16 + 8 + 1 = -7 (yoki inversiya + 1: ~(1111 1001) + 1 = 0000 0110 + 1 = 0000 0111 -> -7). Demak, R = -7.

4.2. C# dasturlash tilida dasturiy tekshirish va vizualizatsiya

Talaba qo‘lda olgan natijasini tekshirishi va har bir oraliq qadamning xotiradagi ikkilik ko‘rinishini ko‘rishi uchun C# tilida Convert.ToString(value, 2).PadLeft(8, '0') metodidan foydalanish tavsiya etiladi. Quyidagi dastur kodi yuqoridagi misolning har bir bosqichini ham o‘nlik, ham 8 bitli ikkilik qo‘shimcha kodda ekranga chiqaradi:

```
using System;
class BitwiseVerification
{
   // 8 bitli qo'shimcha kodni chiroyli formatda chiqarish metodi
   static string ToBinary8(sbyte val)
   {
      string raw = Convert.ToString((byte)val, 2).PadLeft(8, '0');
      return raw.Insert(4, " "); // 4 bitlik tetradalarga ajratish
   }
   static void Main()
   {
      sbyte a = -18, b = 25, c = -7;

      // C# da bitli amallar int tipiga kengaytirilgani uchun (sbyte) ga o'giramiz
```

HSR (LONDON),
HOUGHTON STREET REVIEW JOURNAL
ISSN: 2058-5519|Impact factor-15.9
http://houghtonstreetreview.com/index.php/hsr

```
        sbyte notA      = (sbyte)(~a);
        sbyte cShiftL   = (sbyte)(c << 2);
        sbyte bShiftR   = (sbyte)(b >> 1);
        sbyte andPart   = (sbyte)(notA & b);
        sbyte orPart    = (sbyte)(andPart | cShiftL);
        sbyte result    = (sbyte)(orPart ^ bShiftR)
        Console.WriteLine($"a        = {a,4} | {ToBinary8(a)}");
        Console.WriteLine($"b        = {b,4} | {ToBinary8(b)}");
        Console.WriteLine($"c        = {c,4} | {ToBinary8(c)}");
        Console.WriteLine(new string('-', 28));
        Console.WriteLine($"~a       = {notA,4} | {ToBinary8(notA)}");
        Console.WriteLine($"c << 2   = {cShiftL,4} | {ToBinary8(cShiftL)}");
        Console.WriteLine($"b >> 1   = {bShiftR,4} | {ToBinary8(bShiftR)}");
        Console.WriteLine($"~a & b   = {andPart,4} | {ToBinary8(andPart)}");
        Console.WriteLine($"(...)|(...)={orPart,4} | {ToBinary8(orPart)}");
        Console.WriteLine($"NATIJA R  = {result,4} | {ToBinary8(result)}");
    }
}
```

Dastur ishga tushirilganda konsol oynasida barcha oraliq bosqichlar qo‘lda bajarilgan hisob-kitoblar bilan 100% mos kelishini talaba bevosita ko‘radi. Ayniqsa, Convert.ToString((byte)val, 2) konstruksiyasi manfiy sbyte qiymatining xotiradagi asl 8 bitli nusxasini (32 bitli int ga kengaytirib yubormasdan) aniq ko‘rsatib berish uchun metodik jihatdan juda muhimdir.

5. METODIK TAVSIYALAR VA XULOSA

O‘quv jarayonida talabalarning bitli amallar mavzusini o‘zlashtirish samaradorligini oshirish uchun laboratoriya mashg‘ulotlarini quyidagi tizim asosida tashkil etish tavsiya etiladi:

1. Ko‘rgazmali jadval usuli: Har bir bitli ifodani yechishda 8 katakli maxsus bit-to‘rdan (bit-grid) foydalanish talabalarning razryadlar siljishida adashishini kamaytiradi.

2. Chegaraviy holatlar tahlili: Talabalarga -128 (sbyte.MinValue) sonini inkor qilish (~(-128) = 127) hamda manfiy sonlarni o‘ngga siljitishda (masalan, -1 >> 5 = -1) yuzaga keladigan invariant holatlarni mustaqil tadqiq qilish topshiriqlarini berish.

3. Amaliy muhandislik masalalari: Bitli operatorlarni mavhum matematik misollar bilan cheklamay, ularni RGB rang kodlarini ajratish, ruxsatlar (Flags enum) tizimini boshqarish va juft/toqlikni tekshirish kabi real C# dasturlash vazifalariga bog‘lash.

Xulosa qilib aytganda, manfiy sonlarning ikkilik qo‘shimcha kodda saqlanish mexanizmini ham algoritmik, ham pozitsion vazn nuqtai nazaridan tushuntirish, bitli mantiqiy operatorlarni C# tilining tiplashtirish qoidalari bilan bog‘lab o‘rgatish hamda qo‘lda trassirovka qilishni dasturiy tekshirish bilan uyg‘unlashtirish talabalarning tizimli fikrlash qobiliyatini sezilarli darajada rivojlantiradi.

HSR (LONDON),
HOUGHTON STREET REVIEW JOURNAL
ISSN: 2058-5519|Impact factor-15.9
http://houghtonstreetreview.com/index.php/hsr

FOYDALANILGAN ADABIYOTLAR RO‘YXATI

1. Tanenbaum A.S., Austin T. Structured Computer Organization. 6th Edition. — Pearson Education, 2013. — 800 p.
2. Bryant R.E., O'Hallaron D.R. Computer Systems: A Programmer's Perspective. 3rd Edition. — Pearson, 2016. — 1120 p.
3. Albahari J., Albahari B. C# 12 in a Nutshell: The Definitive Reference. — O'Reilly Media, 2023. — 1088 p.
4. Microsoft Learn Documentation. Bitwise and shift operators (C# reference). — https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/operators/bitwise-and-shift-operators
5. Aripov M.M., Begalov B.A. Axborot texnologiyalari va dasturlash asoslari. Oliy o‘quv yurtlari uchun darslik. — Toshkent: Noshir, 2019.