Міністерство освіти і науки України

Національний університет “Львівська Політехніка”

Лабораторна робота №9

Виконав:
Студент групи АП-11
Білий Анатолій Іванович

Прийняв:

Чайковський І.Б.

Львів – 2024

”Логічні та побітові операції у мові С”

Мета роботи: навчитися використовувати логічні та побітові операції під час програмування на мові С.

Теоретичні відомості

В програмуванні треба мати можливість не лише проводити обчислення над числовими даними, тобто робити арифметичніоперації, але й обробляти логічні дані. З логічними даними програма має справу, коли перевіряє чи виконується деяка умова.

Наприклад, крім звичайного для арифметики питання типу скільки буде 2* 2?, можливо ставити питання типу чи вірно, що 2*2 = 4? або чи вірно, що 2*2 > 5?Іншими словами, можна розглядати 2*2 = 4 та 2*2 > 5 як свого роду вирази, і казати про обчислення значень цих виразів. Значенням першого виразу є логічна істина (true) , значенням другого – логічна хиба (false).

В мові С логічні значення зображуються за допомогою цілих чисел. А саме, число 0 зображує логічну хибу, а будь-яке відмінне від нуля число зображує логічну істину.

В мові С існує три логічні операції:

1. Логічна операція І&&;
2. Логічна операція АБО ||;
3. Логічна операція НЕ ! або логічне заперечення.

Логічні операції утворюють складну (складену) умову з декількох простих (двох або більше) умов. Ці операції спрощують структуру програмного коду в декілька разів. Можна їх не використовувати, але у такому випадку кількість умов , і відповідно самого програмного коду, збільшується.

У таблиці 1 коротко охарактеризовано всі логічні операції в мові програмування С , які використовуються для побудови логічних умов.

Таблиця 1

Логічні операції С

Операції	Позначення	Умова	Короткий опис
І	&&	a==3 && b>4	Складена умова істинна, якщо істинні обидві прості умови
АБО	||	a==3 || b>4	Складена умова істинна, якщо істинна, хоча б одна з простих умов
НЕ	!	! (a==3)	Умова істинна, якщо а не дорівнює 3

Отже, логічні оператори - це оператори, які приймають в якості аргументів логічні значень (брехня або істина) і повертають логічне значення. Як і звичайні оператори, вони можуть бути одномісними (унарними, тобто приймати один аргумент), двомісними (бінарні, приймають два аргументи), тримісними і т.д.

Операція логічне І, виконується згідно таблиці істинності:

X	Y	X&&Y
0	0	0
0	1	0
1	0	0
1	1	1

Ця операція повертає 1, якщо обидва операнди ненульові, та 0 в противному випадку. Операція гарантує обчислення зліва направо, а якщо лівий операнд є 0, правий не обчислюється. Операнди не повинні бути обов'язково однакового типу, але повинні мати один з основних типів. Результат завжди типу int.

Операція логічне АБО, виконується згідно таблиці істинності:

X	Y	X||Y
0	0	0
0	1	1
1	0	1
1	1	1

Ця операція повертає 1, якщо хоча би один з операндів ненульовий, і 0 в противному випадку. Операція гарантує обчислення зліва направо, а якщо перший операнд не є 0, другий операнд не обчислюється. Тип int.

Комп'ютер виконує обчислення саме над двійковими числами, тому в мові С передбачені побітові операції, що дозволяють перевіряти і встановлювати задані розряди або множини розрядів у цілочисельних змінних.

У мові С передбачено кілька операцій для роботи з бітами, але їх не застосовують до змінних типуf float чи double:

– побітове І (&)– результатом операції є кон'юнкція побітового зображення чисел. Результатом є побітова функція І операндів. Результат обчислюється побітово – для кожного розряду операндів згідно таблиці істинності операції логічне І і записується у відповідний розряд. Операція застосовується тільки до операндів типу int;

– побітове АБО (|) – результатом операції є диз'юнкція побітового зображення чисел. Відповідна побітова функція АБО виконується для кожного розряду операндів згідно наведеної для логічного АБО таблиці істинності. Застосовується тільки до операндів типу int.;

– побітове виключне АБО (^) (XOR) – результатом операції є додавання по модулю 2 побітового зображення чисел. Дана функція виконується для кожного розряду операндів згідно наведеної таблиці істинності. Застосовується тільки до операндів типу int.

X	Y	X^Y
0	0	0
0	1	1
1	0	1
1	1	0

В інших мовах програмування символ ^ застосовують для виконання операції піднесення до степеня. В мові С піднесення до другого або третього степенів зручно виконувати простим перемноженням. В інших випадках для піднесення числа x до степеня y слід використовувати вбудовану функцію pow(x,y);

Прості висловлювання мовами програмування можна записати у вигляді логічних виразів із використанням операцій порівняння.

У таблиці 2 представлено значення операцій порівняння.

Таблиця 2

Операції порівняння

Операція	**Значення**
<	менше
<=	менше або рівне
==	перевірка на рівність
>=	більше або рівне
>	більше
!=	перевірка на нерівність

До бітових операцій також відносять бітові зсуви. При зсуві значення бітів копіюються в сусідні за напрямом зсуву.

Розрізняють зсув вліво (в напрямку від молодшого біта до старшого) і вправо (в напрямку від старшого біта до молодшого).

– зсув ліворуч(<<) – зсуває ліворуч побітове зображення лівого операнда на кількість розрядів, указану як правий операнд (праворуч дописуються нулі)

Наприклад, якщо: y= 11001001000000000000000000010001, після чого використовується вираз: y=x<<4, то значення y буде наступне: 10010000000000000000000100010000. Операція зсуву вліво відкидає старші біти, які знаходяться за межами діапазону типу результату, і задає позиції порожніх бітів низького порядку, рівні нулю;

– зсув праворуч(>>) – зсуває праворуч побітове зображення лівого операнда на кількість розрядів, указану як правий операнд (ліворуч дописується копія знакового біта чи 0);

– доповнення (~) – результатом операції є побітове заперечення операнда. Дана операція дає доповнення до цілого; це означає, що кожен біт зі значенням 1 отримує значення 0, і навпаки.

У мові CI операції з вищими пріоритетами обчислюються першими. У таблиці 3 представлено пріоритет операцій.

Асоціативність – напрямок виконання операцій у разі, якщо операції мають однаковий пріоритет.

Таблиця 3

Пріоритет операцій в С

Пріоритет	Операція	Асоціативність	Опис
1	::	зліва направо	унарна операція дозволу області дії
	[ ]		операція індексування
	()		круглі скобки
	.		звернення до члена структури або класу
	->		звернення до члена структури або класу через покажчик
2	++	зліва направо	постфіксний інкремент
	- -		постфіксний декремент
3	++	справа наліво	префікс ний інкремент
	- -		префіксний декремент
4	*	зліва направо	множення
	/		ділення
	%		залишок від ділення
5	+	зліва направо	додавання
	–		віднімання
6	>>	зліва направо	зсув вправо
	<<		зрушення вліво
7	<	зліва направо	менше
	<=		менше або дорівнює
	>		більше
	>=		більше або дорівнює
8	==	зліва направо	дорівнює
	!=		не дорівнює
9	&&	зліва направо	Логічне І
10	||	зліва направо	Логічне АБО
11	?:	справа наліво	умовна операція (тернарного операція)
12	=	справа наліво	присвоювання
	*=		множення з привласненням
	/=		поділ з привласненням
	%=		залишок від ділення з привласненням
	+=		додавання з привласненням
	-=		віднімання з привласненням
13	,	зліва направо	кома

Хід роботи:

1. Написати програму на мові С, яка здійснює такі побітові операції як побітове І, побітове АБО, зсув вліво на 2, зсув вправо на 2. Дані операції застосувати до змінних: a= 017, b=036 (змінні представлені у вісімковій системі числення). Операцію зсуву застосувати тільки до змінної a. Скрін коду програми та результати її виконання представити у звіті.

2. Здійснити вручну виконання операцій з пункту 1. Для цього здійснити переведення значень змінних a= 017, b=036 з вісімкової у двійкову систему числення та виконати необхідні операції згідно п.1. Отримані результати представити у звіті, та порівняти їх з результати програми з пункту 1.

3. Здійснити виконання прикладів, представлених нижче. Представити скріни коду та результати їх виконання у звіті. Пояснити отримані результати.

4. Оформити звіт.

Приклад 1

```
#include <stdio.h>
int main() {
 int a = 017;  // 017 відповідає 15 у десятковій системі
 int b = 036;  // 036 відповідає 30 у десятковій системі

 // Побітове І
 int bitwise_and = a & b;
 printf("a & b = %o\n", bitwise_and);  // %o для виводу у вісімковій системі

 // Побітове АБО
 int bitwise_or = a | b;
 printf("a | b = %o\n", bitwise_or);

 // Зсув вправо на 2 (тільки для змінної a)
 int left_shift_a = a >> 2;
 printf("a >> 2 = %o\n", left_shift_a);
}
```

```
a & b = 16
a | b = 37
a << 2 = 3
```

Приклад 2

1) Переведення значень змінних a= 017, b=036 з вісімкової у двійкову систему числення:

a = 017 (вісімкова) = 000 001 111 (двійкова)
b = 036 (вісімкова) = 011 110 (двійкова)

2)Виконання необхідних операцій:
*Побітове І (a & b):
000 001 111
& 000 011 110
000 001 110

*Побітове АБО (a | b):

000 001 111
| 000 011 110
000 011 111

*Зсув вліво на 2 (тільки для a) (a << 2):
000 001 111 << 2 = 000 111 100

*Зсув вправо на 2 (тільки для a) (a >> 2):
000 001 111 >> 2 = 000 000 011

3)Отже, результати операцій для змінних a та b:

a & b = 000 001 110 (вісімкова: 016)
a | b = 000 011 111 (вісімкова: 037)
a << 2 = 000 111 100 (вісімкова: 074)
a >> 2 = 000 000 011 (вісімкова: 003)

Приклад 3

```
#include <stdio.h>
#include<conio.h>
main() {
int a=0,b=3,c;
c=b%2 ||(a>=0)&&(++b/2*a)==0;
printf("a=%d, c=%d\n",a,c);
getch();
}
```

---

a=0, c=1

Приклад 4

```
#include <stdio.h>
#include<conio.h>
main() {
int a=1,b=0,c;
c=b%2 ||(a>=0)&&(++b*a)==0;
printf("c=%d\n",c);
getch();
}
```

---

c=0

Приклад 5

```
#include <stdio.h>
#include<conio.h>
main() {
int x=2,z,y=0;
z=(x==0)&&(y=x) ||(y>0);
printf("z=%d\n",z);
getch();
}
```

---

z=0

Контрольні питання

1. Пріоритети операцій.
   Дужки ()
   Постфіксні оператори ++ і --
   Префіксні оператори ++ і --
   Оператори множення *, ділення /, залишок від ділення %
   Оператори додавання + і віднімання -
   Оператори відношення <, <=, >, >=
   Оператори рівності ==, !=
   Логічні оператори І &&
   Логічні оператори АБО ||
   Оператор присвоєння =
   Оператори побітового І &, АБО |, XOR ^
   Оператори зсуву бітів <<, >>

2. Таблиця істинності логічного І.

Операція логічне І, виконується згідно таблиці істинності:

X	Y	X&&Y
0	0	0
0	1	0
1	0	0
1	1	1

3. Таблиця істинності логічного АБО.

Операція логічне АБО, виконується згідно таблиці істинності:

X	Y	X||Y
0	0	0
0	1	1
1	0	1
1	1	1

4. Особливості виконання побітових операцій зсуву.
   Операції зсуву вправо >> та вліво << виконують зсув бітів вказаного числа на вказану кількість позицій.
   При зсуві вправо знакове число може зберігати або втрачати свій знак в залежності від реалізації мови.

5. Таблиця істинності побітової операції XOR

X	Y	X^Y
0	0	0
0	1	1
1	0	1
1	1	0

Висновок: Під час виконання цієї лабораторної роботи я навчився використовувати логічні та побітові операції під час програмування на мові С.