Міністерство освіти та науки України

Інститут «Телекомунікації, радіоелектроніки та радіотехніки»

Звіт до лабораторної роботи № 9

З дисципліни «Програмування частина 2»

Логічні та побітові операції у мові С

Виконав студент групи АП-12:

Дроф’як Святослав

Перевірив викладач:

Пелех Н.В.

Львів 2024

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

# 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 а. 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.

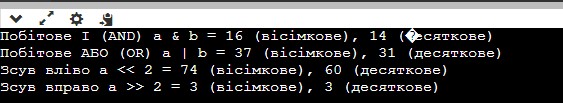
**Завдання 1**

**Реалізація:**

Програмна реалізація мовою С



Код завдання 1, приклад 1



Скріншот виконання коду, завдання 1

Програма демонструє використання побітових операцій в мові C:

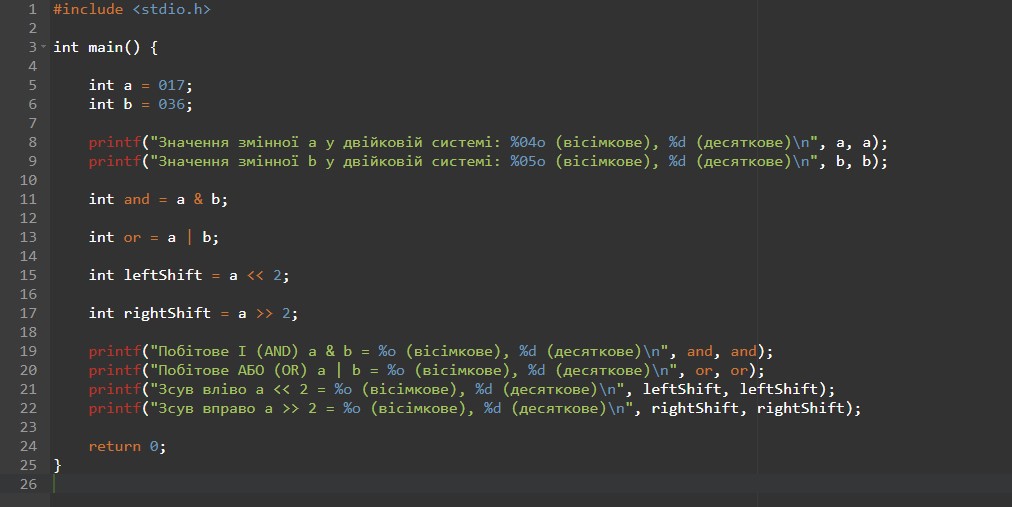
* int a = 017; - ініціалізує змінну a вісімковим значенням 017 (десяткове 15).
* int b = 036; - ініціалізує змінну b вісімковим значенням 036 (десяткове 30).

Потім виконуються наступні побітові операції:

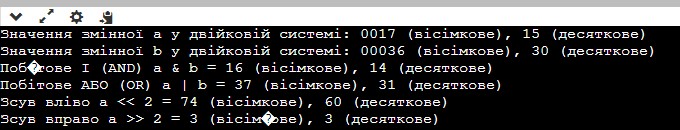
* **AND** (&): and = a & b; - бітове “І” між a та b.
* **OR** (|): or = a | b; - бітове “АБО” між a та b.
* **Left Shift** (<<): leftShift = a << 2; - зсув бітів a на дві позиції вліво.
* **Right Shift** (>>): rightShift = a >> 2; - зсув бітів a на дві позиції вправо.

Результати цих операцій виводяться у вісімковій та десятковій формі. Зокрема:

* and відображає спільні біти a та b.
* or відображає біти, які є в a або в b.
* leftShift показує результат зсуву бітів a вліво, що еквівалентно множенню.
* rightShift показує результат зсуву бітів a вправо, що еквівалентно діленню без залишку.



Код завдання 1, приклад 2



Скріншот виконання коду, завданян 1

Ця програма виконує побітові операції з двома цілими числами, представленими у вісімковій системі числення:

* int a = 017; - ініціалізує змінну a вісімковим значенням 017, що в десятковій системі дорівнює 15.
* int b = 036; - ініціалізує змінну b вісімковим значенням 036, що в десятковій системі дорівнює 30.

Далі програма виводить:

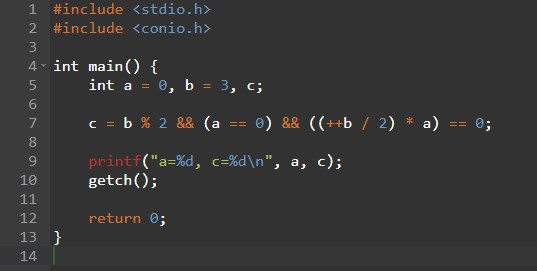
* Початкові значення a та b у вісімковій та десятковій формі.
* Результат побітової операції **AND** (&) між a та b.
* Результат побітової операції **OR** (|) між a та b.
* Результат побітового зсуву вліво (<<) для a на дві позиції.
* Результат побітового зсуву вправо (>>) для a на дві позиції.

Кожен результат виводиться у вісімковій та десятковій формі. Зсув вліво еквівалентний множенню , а зсув вправо - діленню без залишку.

**Завдання 2**

**Реалізація:**

Програмна реалізація мовою С



Код завдання 2, приклад 1



Скріншот виконання коду, приклад 1

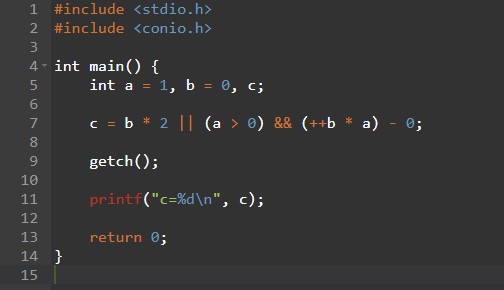
Програма виконує логічні операції та виводить значення змінних a та c:

1. Ініціалізує змінні a та b.
2. Обчислює c як логічне “І” (&&) трьох умов:

* b % 2 перевіряє, чи є b непарним.
* (a == 0) перевіряє, чи a дорівнює нулю.
* ((++b / 2) \* a) == 0 інкрементує b, ділить на 2 та множить на a, перевіряючи, чи результат дорівнює нулю.

1. Виводить значення a (яке залишається 0) та c (яке буде 1, оскільки всі умови істинні).

getch() чекає на натискання клавіші перед закриттям програми.



Код завдання 2, приклад 2

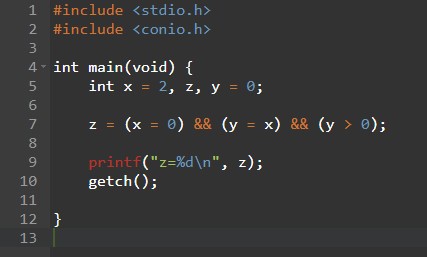


Скріншот виконання коду, приклад 2

Ця програма виконує логічні операції та виводить значення змінної c:

1. Ініціалізує змінні a та b.
2. Обчислює c як логічне “АБО” (||) між результатом множення b \* 2 (що дає 0) та логічним “І” (&&) між умовою (a > 0) (що є істинною, оскільки a дорівнює 1) та виразом (++b \* a) - 0 (що також дає 1, оскільки b інкрементується до 1 перед множенням на a).
3. getch() чекає на натискання клавіші перед виведенням результату.
4. Виводить значення c, яке буде 1, оскільки хоча б одна з умов у логічному виразі істинна.

Таким чином, програма демонструє використання логічних операторів та пріоритету операцій у мові C.



Код завдання 2, приклад 3



Скріншот виконання коду, приклад 3

Програма виконує наступні дії:

1. Ініціалізує змінні x та y.
2. Присвоює x значення 0, а потім використовує це значення для присвоєння y.
3. Обчислює логічне “І” (&&) для виразів (x = 0), (y = x) та (y > 0). Оскільки перший вираз (x = 0) є false, решта виразів не обчислюються, і z отримує значення 0.
4. Виводить значення z.
5. getch() чекає на натискання клавіші перед закриттям програми.

Таким чином, z завжди буде 0, оскільки x присвоюється 0, що робить перший вираз у логічному “І” неправдивим.

# Контрольні питання

1. Пріоритети операцій.

**Пріоритети операцій у мові програмування C**:

* Операції в дужках мають найвищий пріоритет.
* Потім виконуються унарні операції, такі як інкремент (++), декремент (–), логічне заперечення (!).
* Після цього виконуються арифметичні операції множення (\*), ділення (/) та залишок від ділення (%).
* Далі йдуть арифметичні операції додавання (+) та віднімання (-).
* Потім виконуються побітові операції зсуву (<<, >>).
* За ними слідують порівняльні операції (<, <=, >, >=).
* Після цього виконуються рівності (==, !=).
* Далі йдуть побітові AND (&), XOR (^), OR (|).

І нарешті, логічні AND (&&) та OR (||).

1. Таблиця істинності логічного І.

**Таблиця істинності логічного І (AND)**:

A | B | A AND B

0 | 0 | 0

0 | 1 | 0

1 | 0 | 0

1 | 1 | 1

1. Таблиця істинності логічного АБО.

**Таблиця істинності логічного АБО (OR)**:

A | B | A OR B

0 | 0 | 0

0 | 1 | 1

1 | 0 | 1

1 | 1 | 1

1. Особливості виконання побітових операцій зсуву.

**Особливості виконання побітових операцій зсуву**:

* Побітовий лівий оператор зсуву (<<) зрушує біти числа вліво на задану кількість позицій, заповнюючи нулями справа.
* Побітовий правий оператор зсуву (>>) зрушує біти числа вправо, заповнюючи нулями зліва для беззнакових типів або зберігаючи знак для знакових типів.

1. Таблиця істинності побітової операції XOR.

**Таблиця істинності побітової операції XOR**:

A | B | A XOR B

0 | 0 | 0

0 | 1 | 1

1 | 0 | 1

1 | 1 | 0

Висновок :

Під час лабораторної роботи я навчився використовувати логічні та побітові операції в мові С. Програма, яку я написав, правильно виконує побітові операції І, АБО, зсув вліво та вправо на 2. Порівнявши результати виконання програми та ручних розрахунків, переконався у правильності роботи коду. Виконання прикладів збільшило моє розуміння цих операцій. Лабораторна допомогла мені освоїти ці концепції для подальших проектів у мові С.