Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Інститут телекомунікацій, радіоелектроніки та електронної техніки
кафедра «Радіоелектронні пристрої та системи»

Звіт з лабораторної роботи №3а

з дисципліни «Програмування»

Підготував:
ст. групи АП-11
Фостик В. І.

Прийняла:
Гордійчук-Бублівська О. В.

Львів – 2024

Тема:
Логічні і бітові операції та вирази мови С.

Мета:
Дослідження властивостей операцій порівняння, логічних і бітових мови програмування С.

Теоретичні відомості:
Операції порівняння – бінарні, причому обидва операнди повинні бути арифметичного типу, або вказівниками. Результат цілочисельний: 0 (хибність) або 1 (істинність). Тип результату int.
Операції рівності і нерівності відносять до цієї ж групи. Важливо правильно витримувати синтаксис знаку «логічне дорівнює» - ця операція не виконує присвоювання:
вираз == вираз
вираз != вираз
Результатом цих операцій є 0, якщо задане відношення хибне, і 1, якщо істинне. Тип результату int. Ці операції мають нижчий пріоритет, ніж операції попередньої групи, наприклад, у виразі a < b = = c < d спочатку здійснюються порівняння a < b та c < d, результати кожного з них мають значення 0 або 1, після чого операція = = дає результат 0 або 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.
Умовна тримісна операція (тернарна).
На відміну від унарних і бінарних операцій умовна тернарна операція використовується з трьома операндами. В зображенні умовної операції використовуються два символи ‘?’ і ‘:’ і три вирази:
вираз1 ? вираз2 : вираз3.
Першим обчислюється виразу1. Якщо воно істинне, тобто не дорівнює нулю, то обчислюється значення виразу2, яке стає результатом. Якщо при обчисленні виразу 1 отримується 0 (нуль), то в якості результату приймається значення виразу3.
Приклад:
x<0? – x : x;
Вираз повертає абсолютну величину змінної x.

Завдання:

1. Здійснити виконання програми порівняння двох чисел:

```c
#include <stdio.h>

void main(void) {
    float var1, var2;
    printf("Введіть перше число (var1): ");
    scanf("%f", &var1);
    printf("Введіть друге число (var2): ");
    scanf("%f", &var2);

    printf("var1 > var2 дає %d\n", var1 > var2);
    printf("var1 < var2 дає %d\n", var1 < var2);
    printf("var1 == var2 дає %d\n", var1 == var2);
    printf("var1 >= var2 дає %d\n", var1 >= var2);
    printf("var1 <= var2 дає %d\n", var1 <= var2);
    printf("var1 != var2 дає %d\n", var1 != var2);
    printf("!var1 дає %d\n", !var1);
    printf("!var2 дає %d\n", !var2);
    printf("var1 || var2 дає %d\n", var1 || var2);
    printf("var1 && var2 дає %d\n", var1 && var2);
}
```

```
Введіть перше число (var1): 5
Введіть друге число (var2): 1
var1 > var2 дає 1
var1 < var2 дає 0
var1 == var2 дає 0
var1 >= var2 дає 1
var1 <= var2 дає 0
var1 != var2 дає 1
!var1 дає 0
!var2 дає 0
var1 || var2 дає 1
var1 && var2 дає 1
```

2. Здійснити модифікацію та виконання програми згідно взірця, показаного нижче..

```c
#include <stdio.h>
#define TRUE "ІСТИНА"
#define FALSE "ХИБНІСТЬ"

void main(void) {
    float var1, var2;
    printf("Введіть перше число (var1): ");
    scanf("%f", &var1);
    printf("Введіть друге число (var2): ");
    scanf("%f", &var2);
    printf("var1 > var2 це %s\n", var1 > var2 ? TRUE : FALSE);
    printf("var1 < var2 це %s\n", var1 < var2 ? TRUE : FALSE);
    printf("var1 == var2 це %s\n", var1 == var2 ? TRUE : FALSE);
    printf("var1 >= var2 це %s\n", var1 >= var2 ? TRUE : FALSE);
    printf("var1 <= var2 це %s\n", var1 <= var2 ? TRUE : FALSE);
    printf("var1 != var2 це %s\n", var1 != var2 ? TRUE : FALSE);
    printf("var1 || var2 це %s\n", var1 || var2 ? TRUE : FALSE);
    printf("var1 && var2 це %s\n", var1 && var2 ? TRUE : FALSE);
    printf("!var1 це %s\n", !var1 ? TRUE : FALSE);
    printf("!var2 це %s\n", !var2 ? TRUE : FALSE);
}
```

input

```
Введіть перше число (var1): 10
Введіть друге число (var2): 10
var1 > var2 це ХИБНІСТЬ
var1 < var2 це ХИБНІСТЬ
var1 == var2 це ІСТИНА
var1 >= var2 це ІСТИНА
var1 <= var2 це ІСТИНА
var1 != var2 це ХИБНІСТЬ
var1 || var2 це ІСТИНА
var1 && var2 це ІСТИНА
!var1 це ХИБНІСТЬ
!var2 це ХИБНІСТЬ
```

3. Створити програму для виконання прикладу:

```c
#include <stdio.h>

void main(void) {
    int x, y, z;
    x = 2;
    y = 1;
    z = 0;
    x = x && y || z;
    printf("%d\n", x);
    printf("%d\n", x || !y && z);
}
```

```
1
1
```

```c
#include <stdio.h>
#include <conio.h>

void 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
```

```c
#include <stdio.h>
#include <conio.h>

void main() {
    int a = 1, b = 0, c;
    c = b * 2 || (a >= 0) && (++b * a) == 0;
    printf("c=%d\n", c);
    getch();
}
```

```
c=0


...Program finished with exit code 255
Press ENTER to exit console.
```

```c
#include <stdio.h>
#include <conio.h>

void main() {
    int x = 1, y = 2, z;
    z = (x / 2 * 7 <= 0) && (y < 0) || (y % x == 0);
    printf("z=%d\n", --z);
    getch();
}
```

```
z=0
```

```c
#include <stdio.h>
#include <conio.h>

void main() {
    int x = 1, z, b = 0, y = 2;
    z = (x++ * y >= 0) || b++ || (x / y * 3 == 0);
    printf("z=%d\n", z);
    getch();
}
```

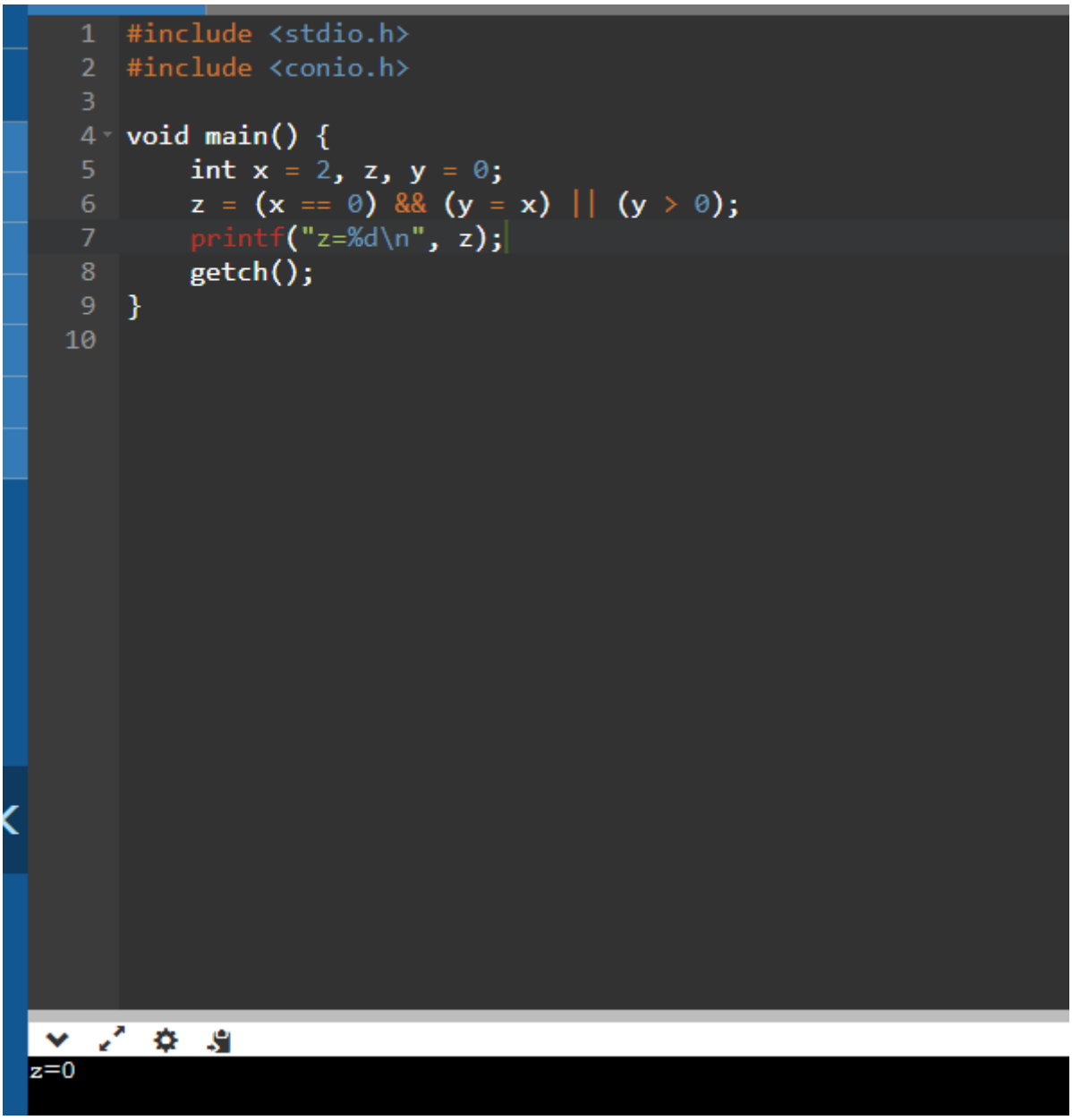
```
z=1
```

```c
#include <stdio.h>
#include <conio.h>

void main() {
    int x = 1, y = 0, z = 2;
    int a = 0;
    z = ((a = x++) * y == 0 || a < 0 && z);
    printf("z=%d\n", z);
    getch();
}
```

```
z=1


...Program finished with exit code 255
Press ENTER to exit console.
```


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

main.c

```c
#include <stdio.h>
#include <conio.h>

void main() {
    int x = 0, y = 3, z;
    z = (++x > y || y-- && y > 0);
    printf("z=%d\n", z);
    getch();
}
```

```
z=1


...Program finished with exit code 255
Press ENTER to exit console.
```

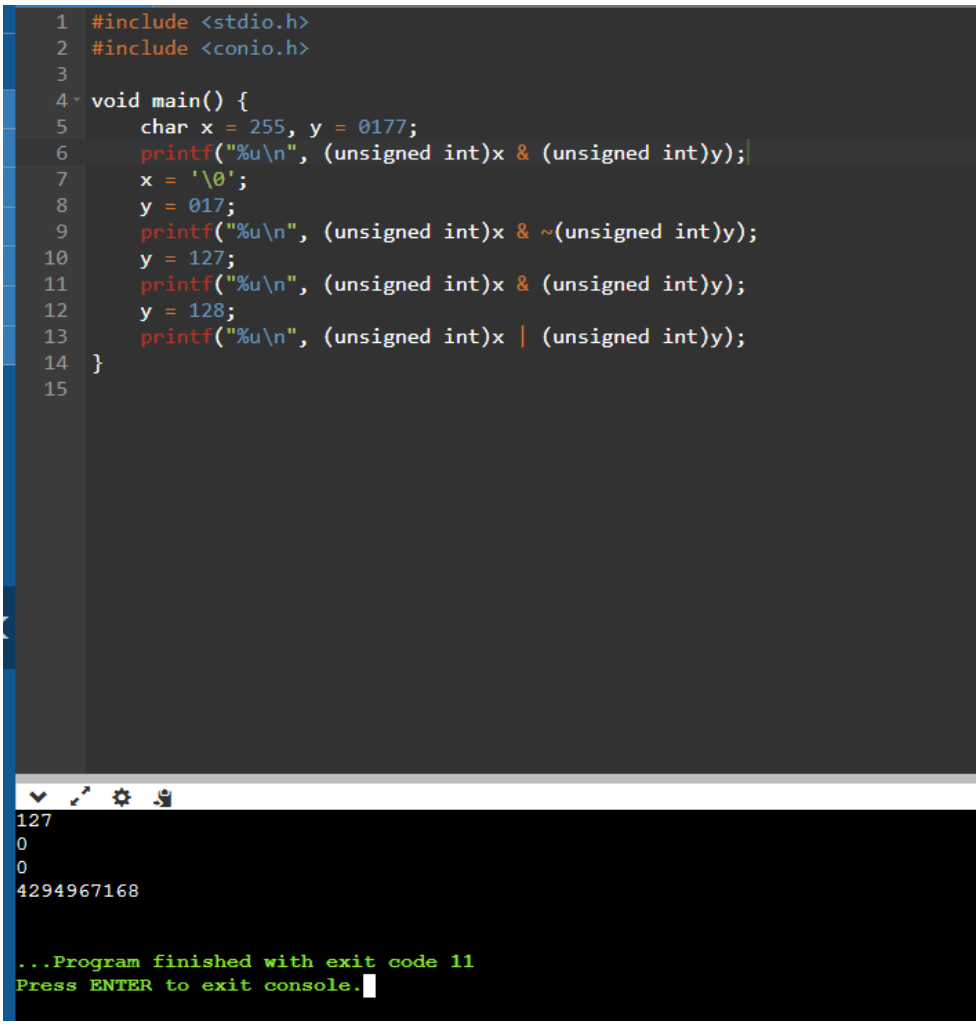
```c
#include <stdio.h>
#include <conio.h>

void main() {
    unsigned int x = 2, y = 1, z = 3, res;
    char chx = 0xAF;

    printf("%u\n", x & y | z);
    x = y = z = 2;
    printf("%u\n", x | y & z);
    x = 3; y = 0; z = 1;
    printf("x^y|~z=%u\n", x ^ y | ~z);
    printf("3|0^~1=%u\n", 3 | 0 ^ ~1);
    x = 1; y = 2; z = 0;
    printf("1&2|0=%u\n", x & y | z);
    printf("~1^2&0=%u\n", ~x ^ y & z);
    printf("2|0&1=%u\n", y | z & x);
    printf("2++&~0|~1=%u\n", ++y & ~z | ~x);
    printf("~3|1&++0=%u\n", ~y | x & ++z);
    x = 0xAF;
    printf("%X\n", x >> 4);
    chx <<= 7;
    printf("0x=%X\n", chx);
    getch();
}
```

```
3
2
x^y|~z=4294967295
3|0^~1=4294967295
1&2|0=0
~1^2&0=4294967294
2|0&1=2
2++&~0|~1=4294967295
~3|1&++0=4294967293
A
0x=FFFFFF80


...Program finished with exit code 255
Press ENTER to exit console.
```



Контрольні запитання:
Ось відповіді на ваші контрольні запитання:

1. Призначення операторів порівняння та тип результату:
   - Оператори порівняння використовуються для порівняння значень. Тип результату - логічне значення (true або false), що вказує на те, чи вірне порівняння.

2. Особливість оператора "логічне дорівнює":
   - Оператор "логічне дорівнює" (===) порівнює значення обох операндів і їх типи даних. Це означає, що не тільки значення повинні бути однакові, але й типи даних повинні бути однакові.

3. Відрізняються операнди в логічних операціях від операндів в операціях порівняння:
   - Операнди в логічних операціях це логічні значення (true або false), тоді як операнди в операціях порівняння - це значення, що порівнюються.

4. Пріоритети операцій:
   - Порядок виконання операцій може залежати від пріоритету. Наприклад, зазвичай арифметичні операції мають вищий пріоритет, ніж логічні.

5. Таблиця істинності логічного І:

A	B	A І B
0	0	0
0	1	0
1	0	0
1	1	1

6. Таблиця істинності логічного АБО:

A	B	A АБО B
0	0	0
0	1	1
1	0	1
1	1	1

7. Особливості виконання обітових операцій зсуву:
- При зсуві вправо операнди зсуваються на вказану кількість позицій вправо, а при зсуві вліво - на вказану кількість позицій вліво. При цьому звільнені позиції заповнюються нулями.

8. Порядок виконання бітових операцій І, АБО:
- Порядок виконання бітових операцій І (AND) та АБО (OR) зазвичай відбувається зліва направо.

9. Таблиця істинності бітової операції XOR:

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0