Міністерство освіти і науки України
Львівський національний університет імені Івана Франка
Факультет електроніки та комп’ютерних технологій

Звіт

Про виконання лабораторної роботи №4

“Мова програмування Сі, компіляції, функцій вводу/виводу”

Виконав:
Студент групи ФЕП-11с
Качмар Денис

Викладач:
Кужій Юрій Іванович

Львів-2024

Мета: Вивчити поняття: основні типи змінних, вказівники, операції та їх пріоритет.

Хід роботи

1) Написав програму, котра ілюструє використання всіх арифметичних і логічних операторів мови Сі, для логічних побітових операцій використовуючи представлення чисел в двійковому форматі.

```c
#include <stdio.h>

// Функція для виведення числа у двійковому форматі
void printBinary(unsigned int n) {
    for (int i = 31; i >= 0; i--) {
        printf("%d", (n >> i) & 1);
    }
    printf("\n");
}

int main() {
    int a = 10, b = 3;

    // Арифметичні оператори
    printf("Арифметичні оператори:\n");
    printf("a + b = %d\n", a + b); // Додавання
    printf("a - b = %d\n", a - b); // Віднімання
    printf("a * b = %d\n", a * b); // Множення
    printf("a / b = %d\n", a / b); // Ділення
    printf("a %% b = %d\n", a % b); // Остача від ділення

    // Інкремент та декремент
    printf("a++ = %d\n", a++); // Постфіксний інкремент
    printf("++a = %d\n", ++a); // Префіксний інкремент
    printf("b-- = %d\n", b--); // Постфіксний декремент
    printf("--b = %d\n", --b); // Префіксний декремент
```

```c
// Логічні оператори
int x = 1, y = 0;
printf("\nЛогічні оператори:\n");
printf("x && y = %d\n", x && y); // Логічне І
printf("x || y = %d\n", x || y); // Логічне АБО
printf("!x = %d\n", !x);         // Логічне НЕ

// Побітові оператори
unsigned int m = 5, n = 3;
printf("\nПобітові оператори:\n");
printf("m у двійковому форматі: ");
printBinary(m);
printf("n у двійковому форматі: ");
printBinary(n);
printf("m & n = ");
printBinary(m & n); // Побітове І
printf("m | n = ");
printBinary(m | n); // Побітове АБО
printf("m ^ n = ");
printBinary(m ^ n); // Побітове виключне АБО (XOR)
printf("~m = ");
printBinary(~m);    // Побітове НЕ
printf("m << 1 = ");
printBinary(m << 1); // Зсув вліво
printf("m >> 1 = ");
printBinary(m >> 1); // Зсув вправо

urn 0;
```

```
Арифметичні оператори:
a + b = 13
a - b = 7
a * b = 30
a / b = 3
a % b = 1
a++ = 10
++a = 12
b-- = 3
--b = 1

Логічні оператори:
x && y = 0
x || y = 1
!x = 0

Побітові оператори:
m у двійковому форматі: 00000000000000000000000000000101
n у двійковому форматі: 00000000000000000000000000000011
m & n = 00000000000000000000000000000001
m | n = 00000000000000000000000000000111
m ^ n = 00000000000000000000000000000110
~m = 11111111111111111111111111111010
m << 1 = 00000000000000000000000000001010
m >> 1 = 00000000000000000000000000000010
```

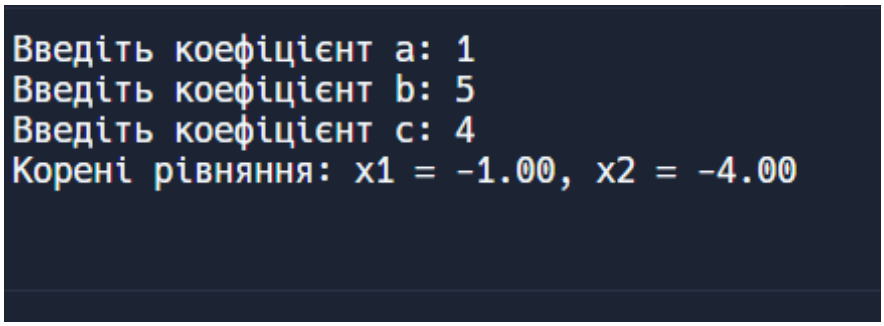
2) Написав програму з введенням числа у змінну і виведенням її адреси та значення через вказівник.

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

int main() {
    setlocale(LC_ALL, "uk_UA.UTF-8");

    int number; // Змінна для зберігання числа
    int* ptr;   // Вказівник на ціле число

    // Введення числа
    printf("Введіть число: ");
    scanf("%d", &number);

    // Присвоєння адреси змінної number вказівнику ptr
    ptr = &number;

    // Виведення значення та адреси через вказівник
    printf("Значення: %d\n", *ptr); // Значення, на яке вказує ptr
    printf("Адреса: %p\n", (void*)ptr); // Адреса, яку має ptr

    return 0;
}
```

```
Введіть число: 2
Значення: 2
Адреса: 0x7ffc6f963f94
```

3) Написав програму розв’язку квадратного рівняння, коефіцієнти якого вводяться з клавіатури.

```c
#include <math.h> // Для використання функцій sqrt
#include <stdio.h>

int main() {
  // Оголошення змінних для коефіцієнтів
  double a, b, c;
  double discriminant, root1, root2;

  // Введення коефіцієнтів з клавіатури
  printf("Введіть коефіцієнт a: ");
  scanf("%lf", &a);
  printf("Введіть коефіцієнт b: ");
  scanf("%lf", &b);
  printf("Введіть коефіцієнт c: ");
  scanf("%lf", &c);

  // Перевірка, чи a не дорівнює 0
  if (a == 0) {
    printf("Коефіцієнт a не може дорівнювати нулю.\n");
    return 1; // Завершення програми з помилкою
  }

  // Обчислення дискримінанта
  discriminant = b * b - 4 * a * c;

  // Визначення коренів залежно від дискримінанта
  if (discriminant > 0) {
    // Два різних корені
    root1 = (-b + sqrt(discriminant)) / (2 * a);
    root2 = (-b - sqrt(discriminant)) / (2 * a);
    printf("Корені рівняння: x1 = %.2f, x2 = %.2f\n", root1, root2);
  } else if (discriminant == 0) {
    // Один корінь
    root1 = -b / (2 * a);
    printf("Рівняння має один корінь: x = %.2f\n", root1);
  } else {
    // Немає дійсних коренів
    printf("Рівняння не має дійсних коренів.\n");
  }

  return 0; // Завершення програми
}
```



Висновок: В процесі роботи над цими завданнями я поглибив свої знання про мову програмування Сі, особливо в аспектах, пов’язаних з арифметичними та логічними операціями. Я навчився використовувати всі типи операторів, включаючи побітові, що дозволяє більш гнучко працювати з двійковими числами. Написавши програму для роботи зі змінними та вказівниками, я здобув практичний досвід у маніпуляціях з пам’яттю, що є важливим аспектом програмування на Сі.