📘 C Programming Basics

🧪 Lab Session: Introduction to C Language

Course: Operating Systems
Date: 16.05.2025

📜 Introduction to C Language

C is a powerful, general-purpose programming language developed by Dennis Ritchie in the early 1970s. It is widely used for system programming and embedded systems because of its speed, portability, and close-to-hardware capabilities.

⭐ Key Features:

- Simple and efficient syntax
- Structured programming approach
- Close to hardware and memory-level manipulation
- High-performance execution
- Cross-platform portability
- Large support of standard libraries

✅ Topics Covered in This practical session

- Printing output using printf()
- Variable declaration and initialization
- Data types: int, float, double, char
- Format specifiers: %d, %f, %lf, %c, %s
- Getting memory size using sizeof()
- Accepting user input with scanf()
- Performing arithmetic operations: +, -, *, /, fmod()
- Assigning values between variables
- Displaying characters and their ASCII values

📌 Practical Code Examples with Output and Explanation

---

1. Hello World Program

```
#include<stdio.h> //insert library
int main(){
   printf("Hello World!\n");
   return 0;
}
```

Fedora Output:

```
[2021ict108@fedora ~]$ vi first.c
[2021ict108@fedora ~]$ gcc first.c -o first
[2021ict108@fedora ~]$ ./first
Hello World!
```

Explanation:
This is the most basic C program. It includes the standard I/O library and prints Hello World! to the console.
After every edit on the .c file we should run the command : gcc filename.c -o filename.
To Run the program: ./filename

---

2. Using Integer Variables

```
#include<stdio.h>
int main(){
   int age = 25;
   printf("%d\n", age);
   age = 31;
   printf("\nNew age: %d\n", age);
   return 0;}
```

Output:

25

New age: 31

Explanation:
We declared an integer variable age, assigned it values, and printed them. %d is used for integers.
here are some other syntaxes used for
%d or %i - integers

%f or %F - float

%lf - double

%e - char

%s - string

---

3. Variable Assignment Between Two Integers

```
#include<stdio.h>

int main(){
   int firstNumber = 25;
   printf("First number: %d\n", firstNumber);


   int secondNumber = firstNumber;
   printf("Second number: %d\n", secondNumber);
   return 0;
}
```

Output:

First number: 25

Second number: 25

Explanation:
This demonstrates assigning the value of one variable to another.

---

4. Multiple Declarations in One Line

```
#include<stdio.h>

int main(){

   int firstNumber, secondNumber = 25;

   printf("First number: %d\n", firstNumber);

   printf("Second number: %d\n", secondNumber);

   return 0;

}
```

Output (may vary):

First number: 1 (Garbage value)

Second number: 25

Explanation:
firstNumber is declared but not initialized, so it may print a garbage value. Always initialize variables!

---

5. Using sizeof() and Printing Double

```
#include<stdio.h>

int main(){

   int age = 10;

   printf("%d\n", age);

   printf("Size: %zu\n", sizeof(age));


   double number = 12.45;

   printf("%.2lf\n", number);
```

```
    return 0;
}
```

Output:

```
10
Size: 4
12.45
```

Explanation:
sizeof() gives memory size in bytes. %.2lf prints a double with 2 decimal places.

---

6. Working with float

```
#include<stdio.h>
int main(){
    float number = 10.9f;
    printf("%f\n", number);
    printf("%.1f\n", number);
    return 0;
}
```

Output:

```
10.900000
10.9
```

Explanation:
%f displays a float. Use %.1f for one decimal place.

---

7. Printing Characters and ASCII

```
#include<stdio.h>
int main(){
    char character = 'z';
```

```
    printf("%c\n", character);
    printf("%d\n", character);
    return 0;
}
```

Output:

z

122

Explanation:
%c prints the character, %d prints its ASCII value.

---

8. User Input Handling

```
#include<stdio.h>
int main(){
    int age;
    double number;
    char alpha;

    printf("Enter your age: ");
    scanf("%d", &age);
    printf("Age = %d\n", age);

    printf("Enter double input: ");
    scanf("%lf", &number);
    printf("Number is = %lf\n", number);

    printf("Enter char input: ");
```

```
    scanf(" %c", &alpha); // Note the space before %c to avoid buffer issues
    printf("Char is = %c\n", alpha);

    return 0;
}
```

Output:

Enter your age: 10

Age = 10

Enter double input: 32.45

Number is = 32.450000

Enter char input: h

Char is = h

Explanation:
We use scanf() to accept different data types from the user. Note the space before %c to clear input buffer.

---

9. Arithmetic Operations in C

```
#include<stdio.h>
#include<math.h>

int main() {
    float num1, num2;

    printf("Enter num1: ");
    scanf("%f", &num1);
```

```
    printf("Enter num2: ");
    scanf("%f", &num2);

    float sum = num1 + num2;
    float sub = num1 - num2;
    float mul = num1 * num2;
    float div = num1 / num2;
    float mod = fmod(num1, num2);

    printf("Add: %.2f\n", sum);
    printf("Sub: %.2f\n", sub);
    printf("Mul: %.2f\n", mul);
    printf("Div: %.2f\n", div);
    printf("Mod: %.2f\n", mod);

    return 0;
}
```

Output Example:

```
Enter num1: 15
Enter num2: 4
Add: 19.00
Sub: 11.00
Mul: 60.00
Div: 3.75
Mod: 3.00
```

Explanation:
This code performs all basic arithmetic operations including modulo (fmod() from math.h for float values).

✅ Summary

Today we are focused on :

- How to write, compile, and run C programs
- How variables and types work
- How to handle input/output
- How to use arithmetic operators
- The behavior of format specifiers and sizeof()