CHAPTER 2: INPUT AND OUTPUT

Topics:

- Using printf() and scanf() for I/O operations
- Formatted output and basic formatting options
- Operators

1.Using printf() and scanf() for I/O Operations:

In C, `printf()` and `scanf()` are standard library functions used for input and output operations.

1. `printf()` function: It is used to print formatted output to the console.

The syntax is:
printf("format string", variable1, variable2, ...);

The format string contains the text to be displayed along with optional format specifiers, which define how the variables should be formatted and displayed.

2. `scanf()` function: It is used to read input from the user.

The syntax is:

scanf("format string", &variable1, &variable2, ...);

The format string contains format specifiers similar to `printf()`, but here, you need to provide memory addresses (using `&`) of the variables where the input will be stored.

Example demonstrating `printf()` and `scanf()`:

```
#include <stdio.h>
int main() {
   int number;
   printf("Please enter an integer: ");
   scanf("%d", &number);
   printf("You entered: %d\n", number);
   return 0;
}
```

2. Formatted Output and Basic Formatting Options:

- `%d`: Used for integers.
- `%f`: Used for floating-point numbers.
- `%c`: Used for characters.
- `%s`: Used for strings.
- `%lf`: Used for double-precision floating-point numbers.
- `%x`: Used to display integers in hexadecimal format.

Example: Reading and displaying a floating-point number

```
#include <stdio.h>
int main() {
   float number;
   printf("Please enter a floating-point number: ");
   scanf("%f", &number);
   printf("You entered: %.2f\n", number);
   return 0;
}
```

Example: Reading and displaying a character

```
#include <stdio.h>
int main() {
   char ch;
   printf("Please enter a character: ");
   scanf(" %c", &ch);
   printf("You entered: %c\n", ch);
   return 0;
}
```

Example: Reading and displaying a string

```
#include <stdio.h>

int main() {
   char name[50];
   printf("Please enter your name: ");
   scanf("%s", name);
   printf("Hello, %s!\n", name);
   return 0;
}
```

Example: Reading and displaying two integers

```
#include <stdio.h>

int main() {
   int num1, num2;
   printf("Please enter two integers separated by a space: ");
   scanf("%d %d", &num1, &num2);
   printf("You entered: %d and %d\n", num1, num2);
   return 0;
}
```

Example: Reading and displaying multiple integers

```
#include <stdio.h>

int main() {
   int num1, num2, num3;
   printf("Please enter three integers separated by spaces: ");
   scanf("%d %d %d", &num1, &num2, &num3);
   printf("You entered: %d, %d, and %d\n", num1, num2, num3);
   return 0;
}
```

In C, operators are symbols that perform operations on variables and values.

1. Arithmetic Operators:
- `+`: Addition
- `-`: Subtraction
- `*`: Multiplication
- `/`: Division
- `%`: Modulo (remainder after division)

2. Relational Operators:
- `==`: Equal to
- `!=`: Not equal to
- `<`: Less than
- `>`: Greater than
- `<=`: Less than or equal to
- `>=`: Greater than or equal to

3. Logical Operators:
- `&&`: Logical AND (returns true if both operands are true)
- `||`: Logical OR (returns true if at least one operand is true)
- `!`: Logical NOT (returns true if the operand is false)

4. Assignment Operators:
- `=`: Assigns the value on the right to the variable on the left
- `+=`: Adds the right operand to the left operand and assigns the result to the left operand
- `-=`: Subtracts the right operand from the left operand and assigns the result to the left operand
- `*=`: Multiplies the left operand by the right operand and assigns the result to the left operand
- `/=`: Divides the left operand by the right operand and assigns the result to the left operand
- `%=`: Calculates the remainder of the division of the left operand by the right operand and assigns the result to the left operand

5. Increment and Decrement Operators

- `++`: Increment the value of a variable by 1
- `--`: Decrement the value of a variable by 1

6. Conditional (Ternary) Operator

- `condition ? expr1 : expr2`: If the condition is true, evaluate expr1; otherwise, evaluate expr2.

Example: Arithmetic Operators

```
#include <stdio.h>

int main() {
    int a = 10, b = 5;
    int sum = a + b;
    int difference = a - b;
    int product = a * b;
    int quotient = a / b;
    int remainder = a % b;

    printf("Sum: %d\n", sum);
    printf("Difference: %d\n", difference);
    printf("Product: %d\n", product);
    printf("Quotient: %d\n", quotient);
    printf("Remainder: %d\n", remainder);

    return 0;
}
```

Example: Relational Operators

```
#include <stdio.h>
int main() {
    int a = 5, b = 10;

    int equal = a == b;
    int notEqual = a != b;
    int lessThan = a < b;
    int greaterThan = a > b;
    int lessThanOrEqual = a <= b;
    int greaterThanOrEqual = a >= b;

    printf("Equal: %d\n", equal);
    printf("Not Equal: %d\n", notEqual);
    printf("Less Than: %d\n", lessThan);
    printf("Greater Than: %d\n", greaterThan);
    printf("Less Than or Equal: %d\n", lessThanOrEqual);
    printf("Greater Than or Equal: %d\n", greaterThanOrEqual);

    return 0;
}
```

Example: Logical Operators

```
#include <stdio.h>

int main() {
   int a = 5, b = 10;

   int logicalAnd = (a > 0) && (b > 0);
   int logicalOr = (a > 0) || (b > 0);
   int logicalNotA = !(a > 0);
   int logicalNotB = !(b > 0);

   printf("Logical AND: %d\n", logicalAnd);
   printf("Logical OR: %d\n", logicalOr);
   printf("Logical NOT for a: %d\n", logicalNotA);
   printf("Logical NOT for b: %d\n", logicalNotB);

   return 0;
}
```

Example: Assignment Operators

```
#include <stdio.h>
int main() {
   int a = 10, b = 5;

   a += b;
   b *= 2;
   a -= 3;
   b /= 2;

   printf("a: %d\n", a);
   printf("b: %d\n", b);

   return 0;
}
```

Example: Increment and Decrement Operators

```
#include <stdio.h>

int main() {
    int a = 5;

    printf("Original value of a: %d\n", a);

    a++;
    printf("After increment: %d\n", a);

    a--;
    printf("After decrement: %d\n", a);

    return 0;
}
```

Example: Conditional (Ternary) Operator

```
#include <stdio.h>

int main() {
    int a = 10, b = 5;
    int max = (a > b) ? a : b;

    printf("The maximum value is: %d\n", max);

    return 0;
}
```