Input and Output

Dr. Nachiket Tapas

I/O Functions

printf function is used to display results to the user (output)

scanf function is used to read data from the user (input)

Printf function

printf("The sum of %d and %d is: %d", no1, no2, sum);
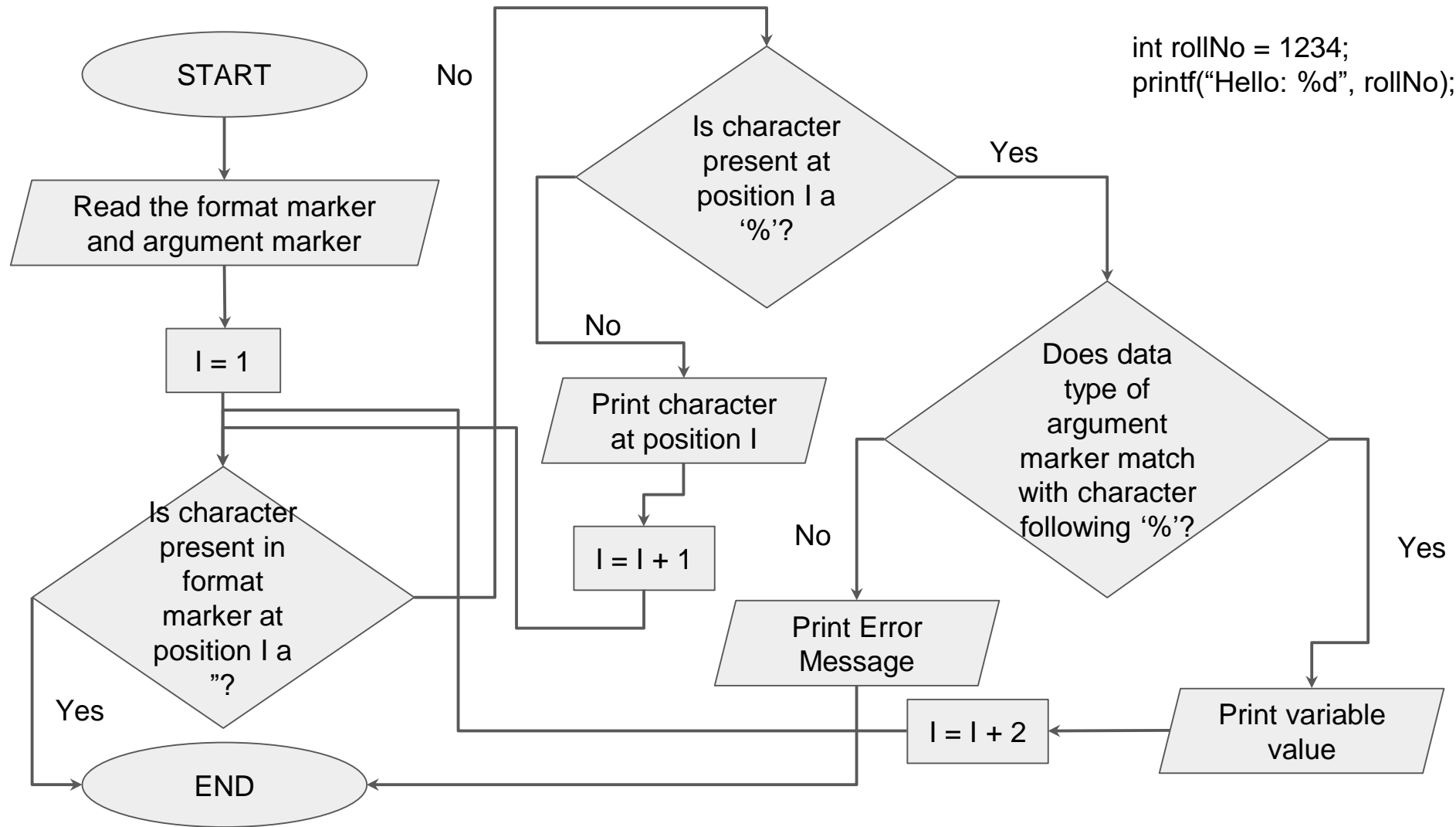
format marker   argument marker

- The format marker is printed one character at a time till a % is encountered.
- If a % is encountered, then the argument marker is used one value at a time.
- The string terminates at ".

Flowchart for printf function

Can you draw the flowchart for printf function?

What all things you need to consider?

```
int rollNo = 1234;
printf("Hello: %d", rollNo);
```

START

Read the format marker and argument marker

I = 1

Is character present in format marker at position I a "?

Is character present at position I a '%'?

No → Print character at position I → I = I + 1

Yes → Does data type of argument marker match with character following '%'?

No → Print Error Message

Yes → Print variable value → I = I + 2

Yes → END

Scanf function

scanf("%d", &no2);

format marker   argument marker

- & is called address-of operator.
- Since scanf is an input function, the value from the user should be stored in memory location.
- The address-of operator is used to give address of the memory location.

int no1;   float length;
        char gender;

scanf("%d", &no1);   scanf("%f", &length);
        scanf("%c", &gender);

Scanf function (contd.)

What will be the output of the following code?

```
#include<stdio.h>
void main() {
        int no1, no2;
        printf("\n Enter two numbers:\n");
        scanf("%d, %d", &no1, &no2);
        printf("Two numbers are:%d and %d", no1, no2);
}
```

Depends on what input you have given. Format is important.

Programs to code

- Write a program to read the side of a square and displays the area along with the side.
- Write a program to read the length and width of a rectangle and displays the area along with the side.
- Write a program to read the price of an item in decimal form (like 15.95) and print the output in paise (1595 paise).
- Write a program that requests two float type numbers from the user and then divides the first number by the second and displays the result along with the numbers.
- Write a program to read the radius of a circle and displays the area along with the radius.

Operators and Expressions

Based on number of operands

- Unary operator (single operand)
- Binary operator (two operands)
- Ternary operator (three operands)

Types of operators

- Arithmetic operators
- Relational operators
- Logical operators
- Increment and decrement operators
- Assignment operators
- Conditional operators
- Bitwise operators
- Special operators

Arithmetic Operators

+	Addition or unary plus
-	Subtraction or unary minus
*	Multiplication
/	Division
%	Modulo division

Unary Operators

Operators that take only one argument

- -5
- +3
- -no1

The / Operator: for integers

When both operand of / are of type integer

- ● Result is integer part of the division
- ● Result is of type integer (floor value of the actual result)

9/2 gives output 4

1/2 gives output 0

The / Operator: for float

When either or both operand of / are of type float

- Result is same as real division
- Result is of type float

9/2 gives output 4.5

1/2 gives output 0.5

The % Operator

The remainder operator or % operator returns integer remainder of the division.

Both operands must be integer

4%2 gives output 0

31%3 gives output 1

Division / and Remainder %

Second operand cannot be 0
-    else run time error

What will be the output of the following?

8/-3

8%-3

Relational Operators

<	is less than
<=	is less than or equal to
>	is greater than
>=	is greater than or equal to
==	is equal to
!=	is not equal to

Relational Operators (contd.)

The result of the expression is always TRUE (1 or non-zero) or FALSE (0).

```
#include<stdio.h>

void main() {

        printf("%d", 8 < 3);

        printf("%d", 8 <= 3);

        printf("%d", 8 > 3);

        printf("%d", 8 >= 3);

        printf("%d", 8 == 3);

        printf("%d", 8 != 3);

}
```

Logical Operators

&&	Logical AND
||	Logical OR
!	Logical NOT

The result of the expression is always TRUE or FALSE.

Truth Table

A	B	A && B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A || B
0	0	0
0	1	1
1	0	1
1	1	1

A	! A
0	1
1	0

Operator Chain

A || B || C || D || …. || Z

Condition check till true is found.

A && B && C && D && …. && Z

Condition check till the end.

Increment and Decrement Operators

Pre-increment	++A
Post-increment	A++
Pre-decrement	--A
Post-decrement	A--

Program

```
#include<stdio.h>

void main() {

        int a = 2;

        printf("%d", a++);

        printf("%d", ++a);

        printf("%d", a--);

        printf("%d", --a);

}
```

What about the following program?

```
#include<stdio.h>

void main() {

        int a = 2;

        a++;

        printf("%d", a);

        ++a;

        printf("%d", a);

}
```

Assignment Operators

A = A + 1	A += 1
A = A - 1	A -= 1
A = A * 5	A *= 5
A = A / 5	A /= 5
A = A % 5	A %= 5

The advantage of assignment operators:
1. Reduced code
2. Evaluated only once

Exception

A = 5;

A = A++ + 5;

O/P: 10

Conditional Operators

● Ternary operator

exp1 ? exp2 : exp3

If exp1 evaluates to true, exp2 is executed. Else exp3 is executed.

Bitwise Operators

&	Bitwise AND
|	Bitwise OR
^	Bitwise XOR
<<	Shift Left
>>	Shift Right

Example

```c
#include <stdio.h>
int main() {
        int a = 5, b = 9;
        printf("%d\n", a&b);
        printf("%d\n", a|b);
        printf("%d\n", a^b);
        printf("%d\n", a<<1);
        printf("%d\n", a>>1);
        return 0;
}
```

Thank You!!