CS F111: Computer Programming

(Second Semester 2020-21)

Lect 7: C Programming

Dr. Nikumani Choudhury
Asst. Prof., Dept. of Computer Sc. & Information Systems
nikumani@hyderabad.bits-pilani.ac.in

BITS Pilani

Hyderabad Campus

Logical Operators (bool)

- There are two logical operators in C (also called logical connectives).
  - && Logical AND
  - || Logical OR
  - ! Logical NOT
- What they do?
  - They act upon operands that are themselves logical expressions.
  - The individual logical expressions get combined into more complex conditions that are true or false.

Logical Operators

– **Logical AND**

• **Result is true if both the operands are true.**

– **Logical OR**

• **Result is true if at least one of the operands are true.**

X	Y	X && Y	X || Y
FALSE	FALSE	FALSE	FALSE
FALSE	TRUE	FALSE	TRUE
TRUE	FALSE	FALSE	TRUE
TRUE	TRUE	TRUE	TRUE

X	!X
TRUE	FALSE
FALSE	TRUE

Logical NOT – Inverts the operand

Logical negation (NOT) operator (!)

- **Example :**

int val1 = !5, val2 = !0, val3 = !-8;

Value of val1 = 0, val2 = 1, val3 = 0

sizeof() operator

- Gives the size of data-type in bytes.

Ex :

sizeof(int) = 4 (bytes)

sizeof(float) = 4 (bytes)

sizeof(char) = 1 (bytes)

sizeof(double) = 8 (bytes)

sizeof(8.6) = 8 bytes, sizeof('A') = 1 byte

HW: print output of sizeof using %ld or %zu may print using %d, but will get a warning

More about printf / scanf

printf() : returns number of characters successfully written on output stream i.e. screen

```
printf("%d\n", printf("Hello World !!!"));
```

Output : Hello World !!!15

scanf() : returns number of items successfully scanned from input stream i.e. keyboard

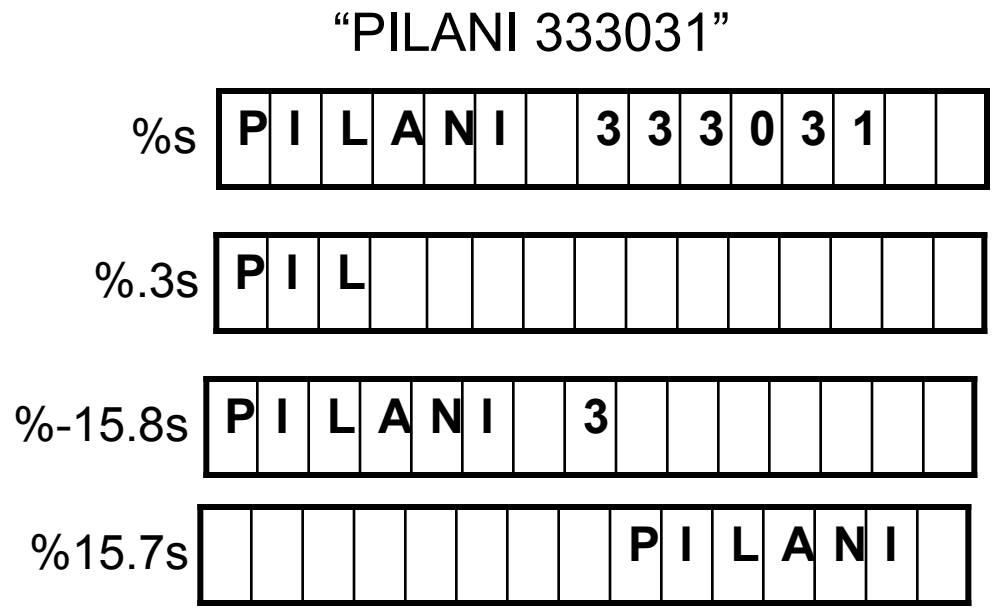
```
int a, b;
float c;
printf("%d\n", scanf("%d %d %f", &a, &b, &c));   // out : 3
```

printf() Examples

1. printf (“%d%c%f”, 23, ‘z’, 4.1);
   23z4.100000
2. printf (“%d %c %f”, 23, ‘z’, 4.1);
   23 z 4.100000
3. printf (“%d\t%c\t%5.1f\n”, 23, ‘z’, 14.2)
   23 z 14.2
4. printf (“The number is %6d”, 23)
   The number is 23
5. printf (“The tax is %8.2f this year”, 233.15);
   The tax is 233.15 this year
6. printf (“The tax is %08.2f this year”, 233.15);
   The tax is 00233.15 this year

printf() Examples continued…

7. printf (“ \”%8c %d\””, ‘h’, 23);
   “ h 23”
8. printf (“This line \r… A new line\n”);
   printf (“A null character\0kills the rest of the line\n”);
   printf (“This is \\ the escape character itself\n”);
   … A new line
   A null character
   This is \ the escape character itself
9. Common printf () errors
   1. printf (“%d %d %d\n”, 45, 65);
      45 65 0 or 45 65 134513657
   2. printf (“%d %d\n”, 34, 76, 43);
      34 76
   3. float x = 123.45;
      printf (“The data is: %d\n”, x);
      The data is: -1087656734 or 0



More printf() Home-reading

- printf("%7.0f %+7.1f %10.1e\n", x, x, y); /*let x=12.0 and y=-3.3*/

  12 +12.0 -3.3e+00

- printf("%7f %7.1f %12e %12.3e", x, x, x, x); /* let x = 123.456 */

  123.456000 123.5 1.234560e+02 1.235e+02

- printf("%.3f %.1f ", x, x); /* let x=123.45 */

  123.450 123.5

- printf("%f %e %f", x, y, x*y); /let x=5000.0 and y=0.0025 */

  5000.000000 2.500000e-03 12.500000

Input formatting: scanf()

- scanf("control string", arg_1, arg_2, …arg_n);
- Conversion specification
  - % | Flag | Maximum width | Size | Code
- scanf("%d%d%d%c", &a, &b, &c, &d);

  214 145 23z
- scanf("%d%d%d %c", &a, &b, &c, &d);/* handles a space betw 23andz */
- scanf("%2d/%2d %2d/%2d", &a, &b, &c, &d);

  13/56 23/76
- scanf("%d-%d-%d", &a, &b, &c);

  31-03-09
- scanf("%d %*d %d", &a, &b);

  342 567 786

char address[50];
scanf("%[a-z]", address);
scanf("%[^\n]", address);

Commonly used codes:
%c, %d, %e, %f, %g, %h, %i, %o, %s, %u, %x

Input formatting: scanf() continued… Home Reading

- scanf("%3d %3d", &a, &b);

  1 2

  123 45

  1234567

  1234 67

- scanf("%d a %f", &i, &x);

  12 a 3.0

  12 3.0

- scanf("%c%c%c", &x, &y, &z);

  a b c

- scanf("%c%1s%1s", &x, &y, &z);

  a b c

More scanf () Home Reading

- Common input errors:

1.
```
float a = 2.1;
scanf (“%5.2f”, &a);
printf (“%4.2f”, a);
```
54.15 (Input)

2.10 (Output)

2.
```
int a, b;
scanf (“%d%d%d”, &a, &b);
printf (“%d %d\n”, a, b);
```

5 16 79
Segmentation fault

3.
```
int a =1;
int b=2;
int c=3;
scanf (“%d%d”, &a, &b, &c);
printf (“%d %d\n”, a, b, c);
```
5 10 14
5 10

Attempts to access a memory location in a way that is not allowed e.g. attempting to write to a read-only location:

```
const char *s = "hello";
*s = 'H';
```

Results may vary depending upon the compiler and hardware.