CS102/IT102
Computer Programming I

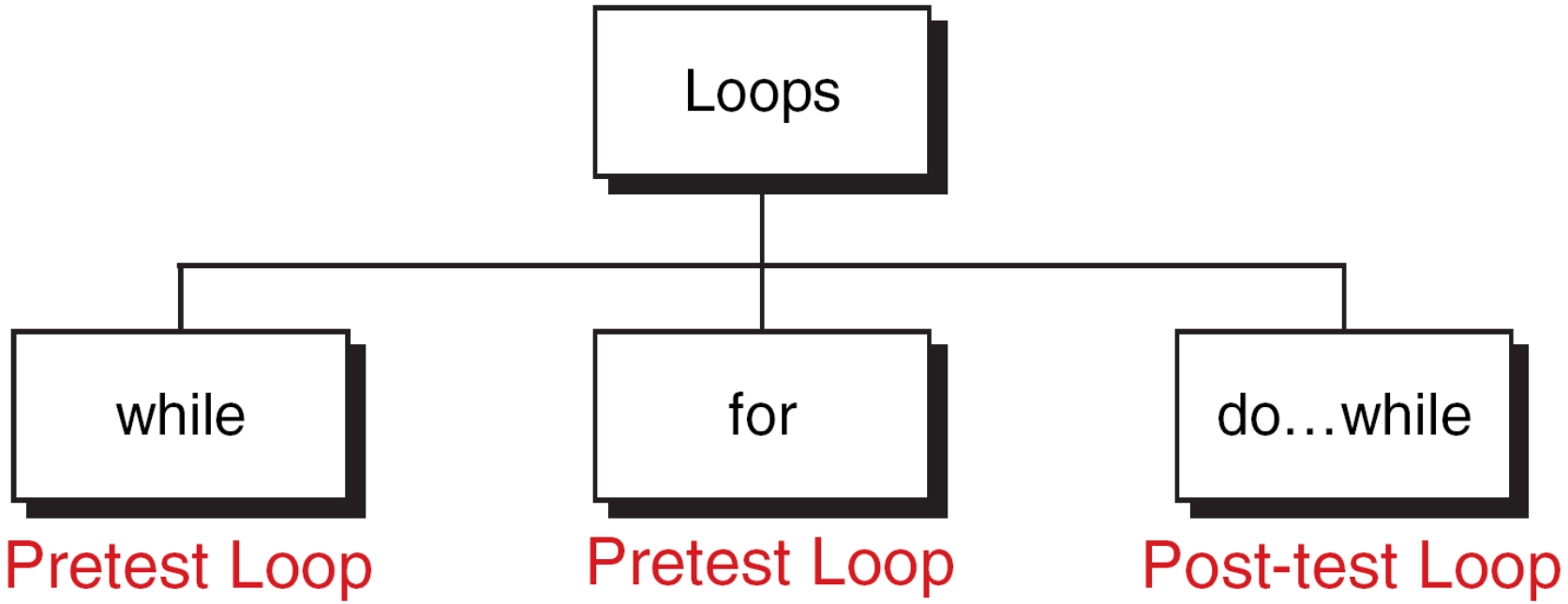
Lecture 11:
Repetition (Part 2)

Bicol University College of Science
CSIT Department
1st Semester, 2023-2024

Topics

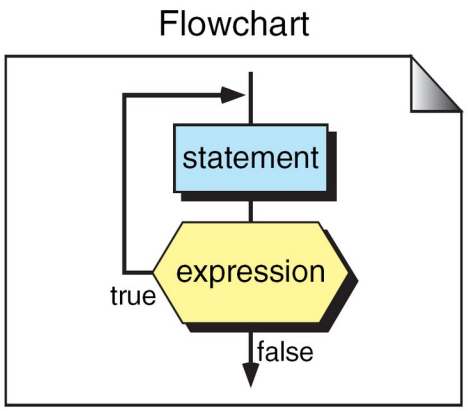
- Post-test loop in C
  - **`do...while`** statement
- **`continue`** statement
- Comma expression
- Sample problems

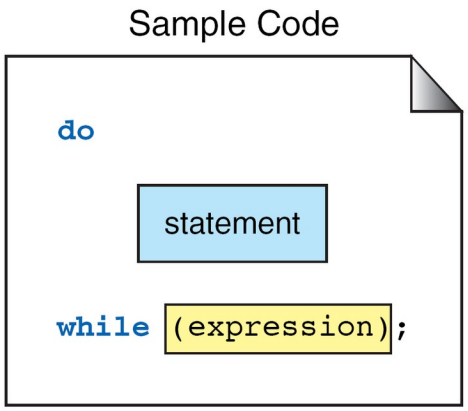
Loops in C

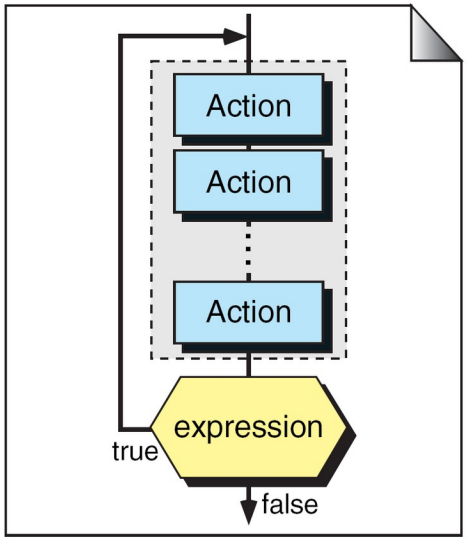

Loops
while
for
do…while
Pretest Loop
Pretest Loop
Post-test Loop

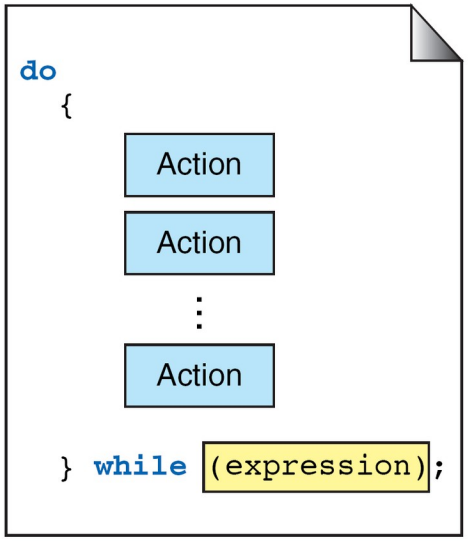
The `do...while` Loop

- The **do...while** statement is a post-test loop.
- It also uses an expression to control the loop, but it tests this expression after the execution of the body
- Note that the do… while is concluded with a semicolon.







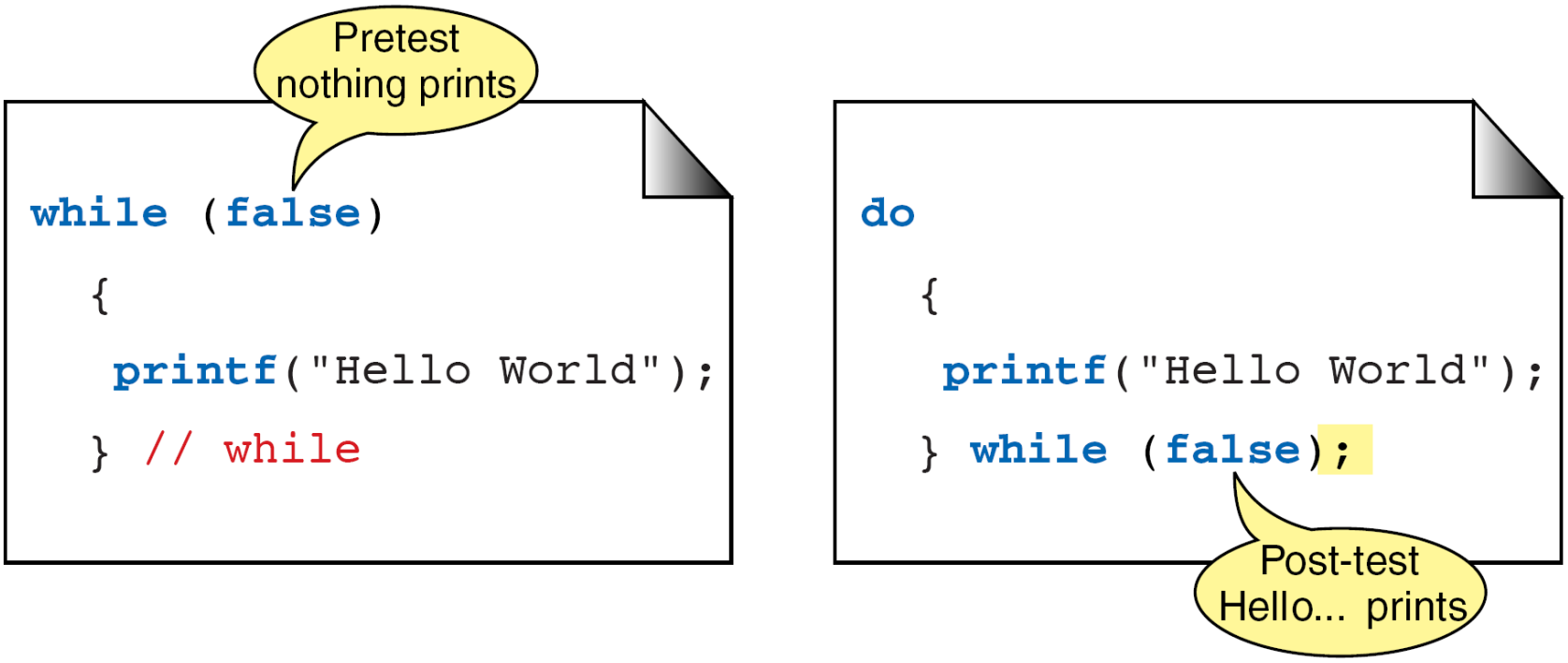


Example:

Two Simple Loops

```
#include <stdio.h>
int main (void)
{

int loopCount;

loopCount = 5;
printf("while loop : ");
while (loopCount > 0)
   printf ("%3d", loopCount--);
printf("\n\n");

loopCount = 5;
printf("do...while loop: ");
do
   printf ("%3d", loopCount--);
while (loopCount > 0);
printf("\n");

return 0;
} /* main */
```

Pre- and Post-test Loops



- In the while loop, the message is not printed, because the limit condition is tested first.
- In the do…while loop, even though the limit test is false, the message is printed because the message is printed before the limit test.

do…while Loop

- Programmers commonly use the do…while loop in data validation to make the program robust:

```
do
   {
      printf ("Enter a number between 10 and 20: ");
      scanf  ("%d", &a);
   }   while (a < 10 || a > 20);
```

The Comma Expression

- is a complex expression made up oftwo expressions separated by commas.
- it is most often used in the for statements.
- The expressions are evaluated left to right
- has the lowest priority of all expressions.
- The value and type of the expressions are the value and type of the right expression; the other expression is included for its side effect.

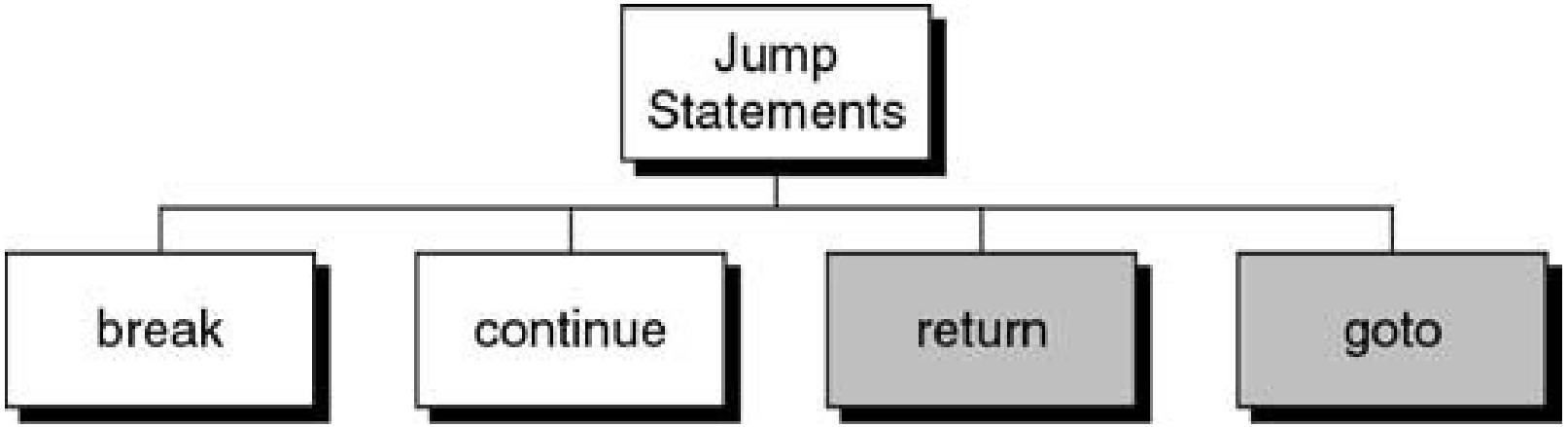

```
for(sum=0,i=1;i<=N;i++){
  scanf("%d", &a);
  sum += a;
}
```

```
i=1,j=2,k=i+j
```

is evaluated as

```
((i=1),(j=2)),(k=(i+j))
```

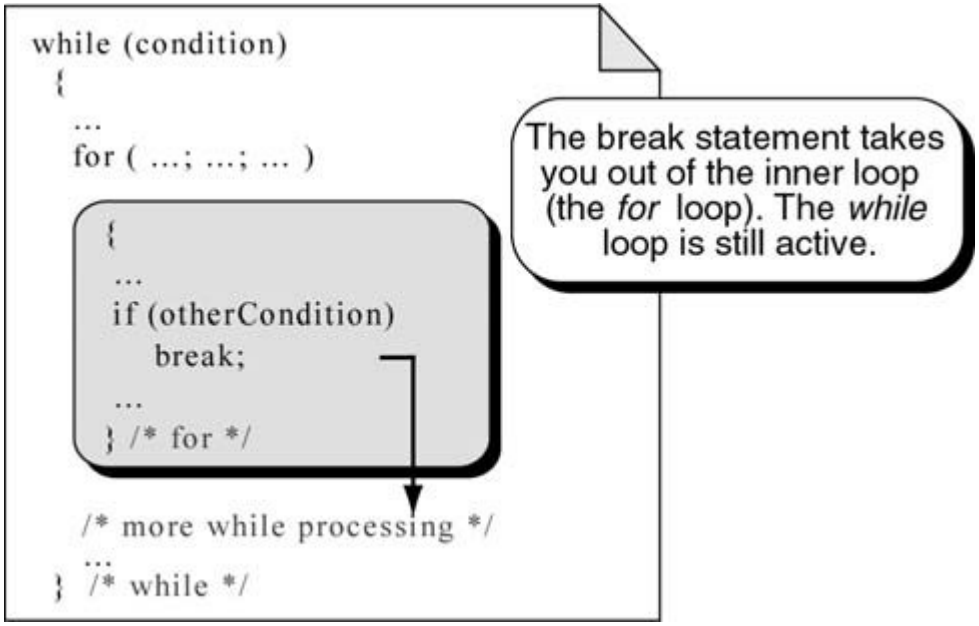
Other Statements Related to Looping



- **break** and **continue** jump statements are related to loops
- **goto** is not valid for structured programs.
- **return** is associated with function

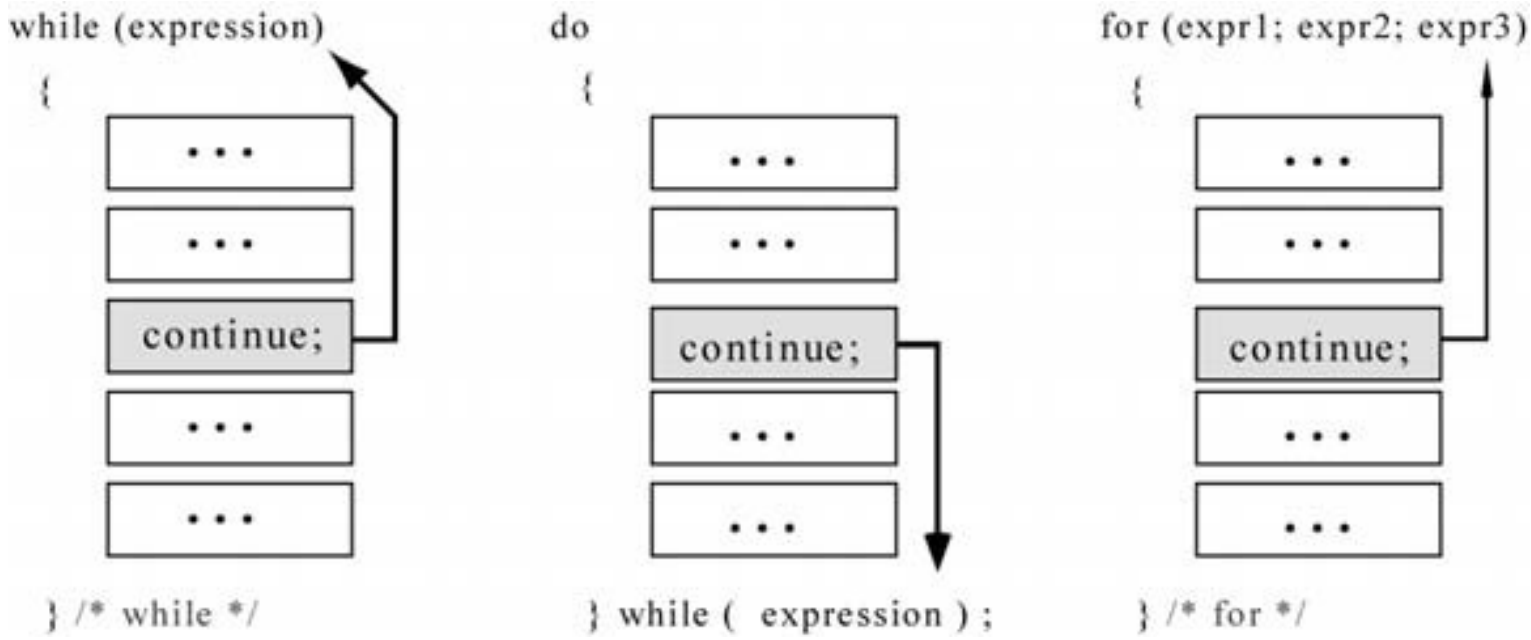
The **`break`** Statement

- **`break`** can be used in any of the loop statements – **`while`**, **`for`**, **`do…while`** – and the selection **`switch`** statement.
- Good structured programming limits the use of the break statement to the **`switch`** statement.



The **`continue`** Statement

- It does not terminate the loop but simply transfers to the testing expression in **`while`** and **`do…while`** statements and transfers to the updating expression in a **`for`** statement.



The use of continue is also considered unstructured programming.

PROBLEM 1

Design and implement it in C a loop that iterates through the positive odd numbers less than 30, and outputs the square of each number.

Output

1 9 25 49 81 121 169 225 289 361 441 529 625 729 841

PROBLEM 2

Design and implement in C a loop that iterates from 1 to 30, but only outputs numbers that are divisible by 3 or 5.

Output

3 5 9 10 12 15 18 21 24 25 27 30

PROBLEM 3

Design and implement in C the loop to create a multiplication table for all combinations of two numbers from 1 to 9.

1	2	3	4	5	6	7	8	9
2	4	6	8	10	12	14	16	18
3	6	9	12	15	18	21	24	27
4	8	12	16	20	24	28	32	36
5	10	15	20	25	30	35	40	45
6	12	18	24	30	36	42	48	54
7	14	21	28	35	42	49	56	63
8	16	24	32	40	48	56	64	72
9	18	27	36	45	54	63	72	81

PROBLEM 4

Design and write in C a loop that produces the following patterns using 2 ***for*** loops.

```
1
2 1
3 2 1
4 3 2 1
5 4 3 2 1
6 5 4 3 2 1
7 6 5 4 3 2 1
```

PROBLEM 5

I.) Design and write in C the loops to produce the following output. Use nested for loops.

```
1  2  3  4  5
1  2  3  4
1  2  3
1  2
1
```

PR=1;OBLEM 6

Create a program that will output the factorial of a positive integer.

Enter a positive integer: 5

The factorial of 5 is 120.

PROBLEM 7

Create a program that will output the factorial of numbers from 1 to any positive integer.

Enter a positive integer: 5

Integer	Factorial
5	120
4	24
3	6
2	2
1	1