Module 6 – Control Flow: Branching and Looping – Part 2

BITS Pilani
Pilani Campus

Dr. Jagat Sesh Challa

Department of Computer Science & Information Systems

Module Overview

- Control Flow: Looping
- while Loop
- do… while Loop
- for Loop

Control Flow: Looping

Looping – basics

- Many activities repetitive in nature
- Examples:
  - *Decimal-to-binary* → repeated division
  - *Computing an average* → repeated addition
- We need a mechanism to perform such repetitive tasks
- The repetition framework should also include the halting mechanism

Loops in C support repetitive tasks

Loops in C

while loop

do… while loop

for loop

innovate achieve lead

BITS Pilani

Pilani Campus

while loop

while loop


```
while (exp is true)
    statement;
```

OR

```
while (exp is true)
{
    statement1;
    statement2;
    …
}
```

Control expressions: Examples

innovate achieve lead

- `while (answer == 'Y')`
- `while (count)`
- `while (5)`
- `while (base>0 && power>=0)`
- `while (--num > 0)`
- `while (printf("Enter the value of N: "))`

Example


```
int var=1;
while (var <=10)
{
    printf("%d \n", var++);
}
printf("While loop over");
```

This program prints first 10 positive integers

A null body


```
while(num > 0);
```

is different from

```
while(num > 0)
```

Exercises

Write a program to calculate the sum of 10 numbers entered by the user

Nested while loops

```
while(exp1){
  …
  while(exp2){
     …
     }
  }
```

```
int i = 10, j, count;
while(i) {
   j = 5;
   while(j) {
      printf("Hello!\n");
      j--;
      }
      i--;
   }
```

How many times is "Hello" printed?

Examples

innovate achieve lead

```
int i = 10, j, count=0;
while(i) {
  j = 5;
  while(j) {
     printf("%d ",
  count++);
     j--;
     }
    i--;
  }
```

```
int i = 10, j, count=0;
while(i) {
  j = 5;
  while(j) {
     count++;
     j--;
     }
    i--;
  }
printf("%d ", count);
```

Exercises

Print this pattern on your screen using loops. No. of rows are user input.

```
int i, j, rows;
printf("Enter number of rows: ");
scanf("%d",&rows);
i =1;
while(i<=rows){
    j =1;
    while(j<=i){
          printf("* ");
         j++;  }
    printf("\n");
   i++;
}
```

```
*
* *
* * *
* * * *
* * * * *
```

Exercises

Print this pattern on your screen using loops.

No. of rows are user input.

```
int i, space, rows, k=0;
printf("Enter number of rows: ");
scanf("%d",&rows);
i=1;
while(i<=rows) {
    space=1;
    while(space<=rows-i){
        printf("  ");
        space++;
    }
while(k != 2*i-1){
    printf("* ");
    ++k;
}
printf("\n");
i++;
k=0;
}
```

```
                *
              * * *
            * * * * *
          * * * * * * *
        * * * * * * * * *
```

Practice tasks

innovate achieve lead

```
* * * * * * * * *
 * * * * * * *
  * * * * *
   * * *
    *
```

```
1
2 3
4 5 6
7 8 9 10
```

innovate achieve lead

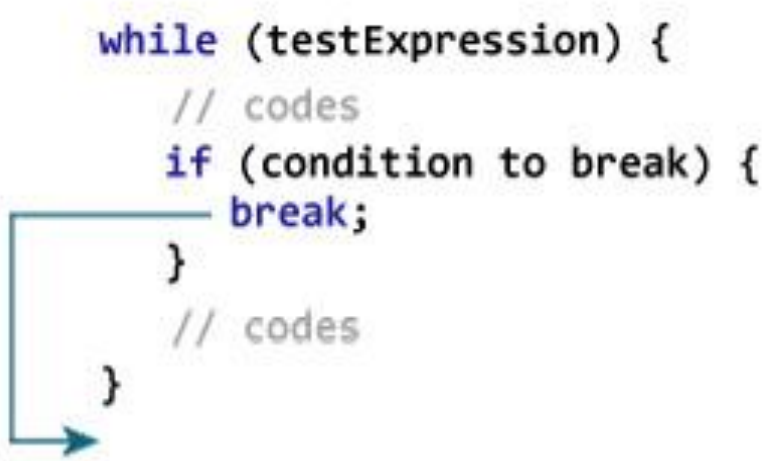
BITS Pilani

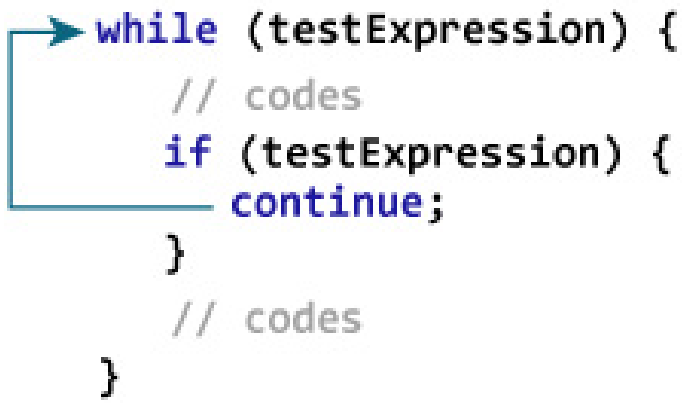
Pilani Campus

Break and Continue

Break and continue

- break – immediate suspension of the current loop
- continue – restart a new iteration





Break and continue (Work out on board)

Write a program to calculate the sum of at most 10 numbers entered by the user

- When the user enters a negative value, prompt the user to enter a positive integer
- When the user enters zero, terminate the loop
- Display the sum of the valid numbers entered by the user

innovate achieve lead

BITS Pilani

Pilani Campus

The do… while loop

The do … while loop


```
do {
    statement(s);
} while( condition );
```

- Conditional expression appears at the end
- Statements in the loop execute once before the condition tested
- Useful in scenarios where you want the code to execute at least once
- While loop: first satisfy the condition then proceed
- Do while loop: proceed first, then check the condition

Example


```
int var=1;
do {
    printf("%d ", var++);
} while (var <=10);
printf("Do… While loop over");
```

This program prints first 10 positive integers

while vs do… while

while loop	do…while loop
This is entry-controlled loop. It checks condition before entering into loop	This is exit-controlled loop. Checks condition when coming out from loop
The while loop may run zero or more times	Do-While may run more than one times but at least once.
The variable of test condition must be initialized prior to entering into the loop	The variable for loop condition may also be initialized in the loop also.

`while(condition){`
`  //statement`
`}`
	`do{`
	`//statement`
	`}while(condition);`

innovate achieve lead

BITS Pilani

Pilani Campus

The for loop

For loop


```
for (initialization; condition; increment or decrement)
{
        //Statements to be executed repeatedly
}
```

- Initialization executed first, **and only once**
- Condition evaluated. If true, loop body executed
- Increment/decrement executed to update control variable
- Condition evaluated again. The process repeats.
- Condition false → loop terminates

Examples

```
int main()
{
    int i;
    for (i=1; i<=3; i++)
    {
        printf("%d\n", i);
    }
    return 0;
}
```

```
int main()
{
    int i;
    for (i=1; i<=3; i++)
        printf("%d\n", i);
    return 0;
}
```

Various forms of for loop

innovate achieve lead

- `for (num=10; num<20; num=num+2)`
-
```
int num=10;
for ( ;num<20; num++)
```
- `for (num=10; num<20; ) { … …  num++; }`
- `for( ; ; )`
- `for (i=1,j=10; i<10 && j>0; i++, j--)`

Exercises

What does this code print?

```
int count;
for (count=1; count<=3; count++);
            printf("%d\n", count);
```

Exercises

innovate achieve lead

Repeat this exercise using for loop:

Write a program to calculate the sum of max 10 numbers entered by the user

- When the user enters a negative value, prompt the user to enter a positive integer
- When the user enters zero, terminate the loop
- Display the sum of the valid numbers entered by the user

Exercises

Print this pattern on your screen using For loops. No. of rows are user input.

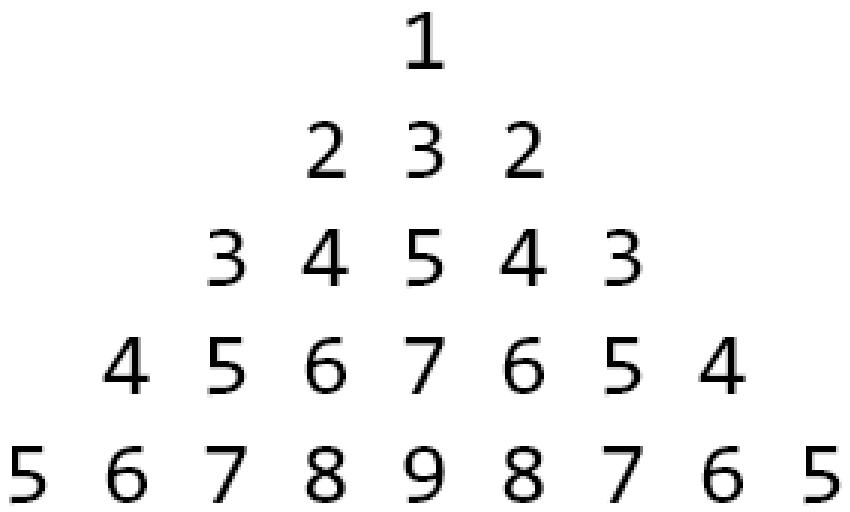
```
int i, j, rows;
   printf("Enter number of rows: ");
   scanf("%d",&rows);
   for(i=1; i<=rows; ++i)
   {
      for(j=1; j<=i; ++j)
            printf("* ");
      printf("\n");
   }
```

```
*
* *
* * *
* * * *
* * * * *
```

Exercises

innovate achieve lead

Homework



“for” vs “while” loop

innovate achieve lead

BASIS FOR COMPARISON	FOR	WHILE
Declaration	`for(initialization; condition; iteration){ //body of 'for' loop }`	`while ( condition) { statements; //body of loop }`
Format	Initialization, condition checking, iteration statement are written at the top of the loop.	Only initialization and condition checking is done at the top of the loop.
Use	The 'for' loop used only when we already knew the number of iterations.	The 'while' loop used only when the number of iteration are not exactly known.
Condition	If the condition is not put up in 'for' loop, then loop iterates infinite times.	If the condition is not put up in 'while' loop, it provides compilation error.
Initialization	In 'for' loop the initialization once done is never repeated.	In while loop if initialization is done during condition checking, then initialization is done each time the loop iterate.
Iteration statement	In 'for' loop iteration statement is written at top, hence, executes only after all statements in loop are executed.	In 'while' loop, the iteration statement can be written anywhere in the loop.

innovate achieve lead

BITS Pilani

Pilani Campus

Thank you

Q & A