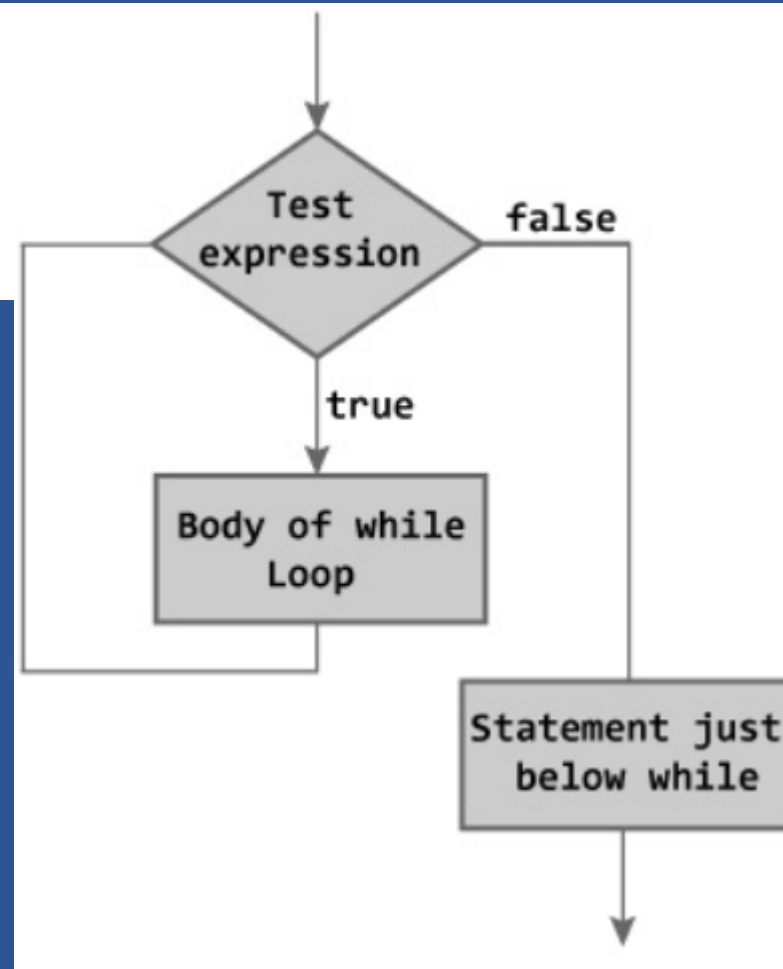


Loop

While loop

```
while (condition)
{
    loop body;
}
```



Print number from 0 to 9

```
// Initialize the counter
int counter = 0;
// Execute the loop body while the loop condition holds
while (counter <= 9)
{
    // Print the counter value
    Console.WriteLine("Number : " + counter);
    // Increment the counter
    counter++;
}
```

Summing the Numbers from 1 to N

```
// Summing the Numbers from 1 to N
Console.Write("n = ");
int n = int.Parse(Console.ReadLine());
int num = 1;
int sum = 1;
Console.Write("The sum 1");
while (num < n)
{
    num++;
    sum += num;
    Console.Write(" + " + num);
}
Console.WriteLine(" = " + sum);
// N = 3
//The sum 1 + 2 + 3 = 6
```

Check if a Number is prime

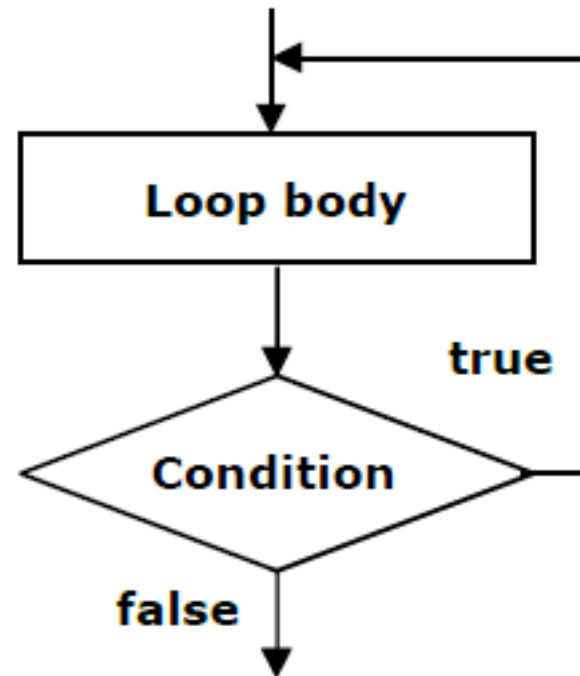
```
// Check if a Number is prime
Console.Write("Enter a positive number: ");
int num2 = int.Parse(Console.ReadLine());
int divider = 2;
int maxDivider = (int)Math.Sqrt(num2);
bool prime = true;
while (prime && (divider <= maxDivider))
{
    if (num2 % divider == 0)
    {
        prime = false;
    }
    divider++;
}
Console.WriteLine("Prime? " + prime);
//Enter a positive number: 37
//Prime? True
//Enter a positive number: 34
//Prime? False
```

Break command

```
// break command. Example using factorial
// using break command in while loop
// The factorial = all integers less than or equal to n or equal to it.
// It is written down as n!
// formular  $N! = 1 * 2 * 3 \dots (n - 1) * n$ , for  $n > 1$ ;
//  $2! = 1 * 2$ ;
//  $1! = 1$ ;
//  $0! = 1$ 
Console.WriteLine("Enter a number : ");
int n = int.Parse(Console.ReadLine());
// "decimal" is the biggest C# type that can hold integer values
decimal factorial = 1;
// Perform an "infinite loop"
while (true)
{
    if (n <= 1)
    {
        break;
    }
    factorial *= n;
    n--;
}
Console.WriteLine("n! = " + factorial);
```

Do-While Loops

```
do  
{  
    executable code;  
} while (condition);
```



do-while factorial

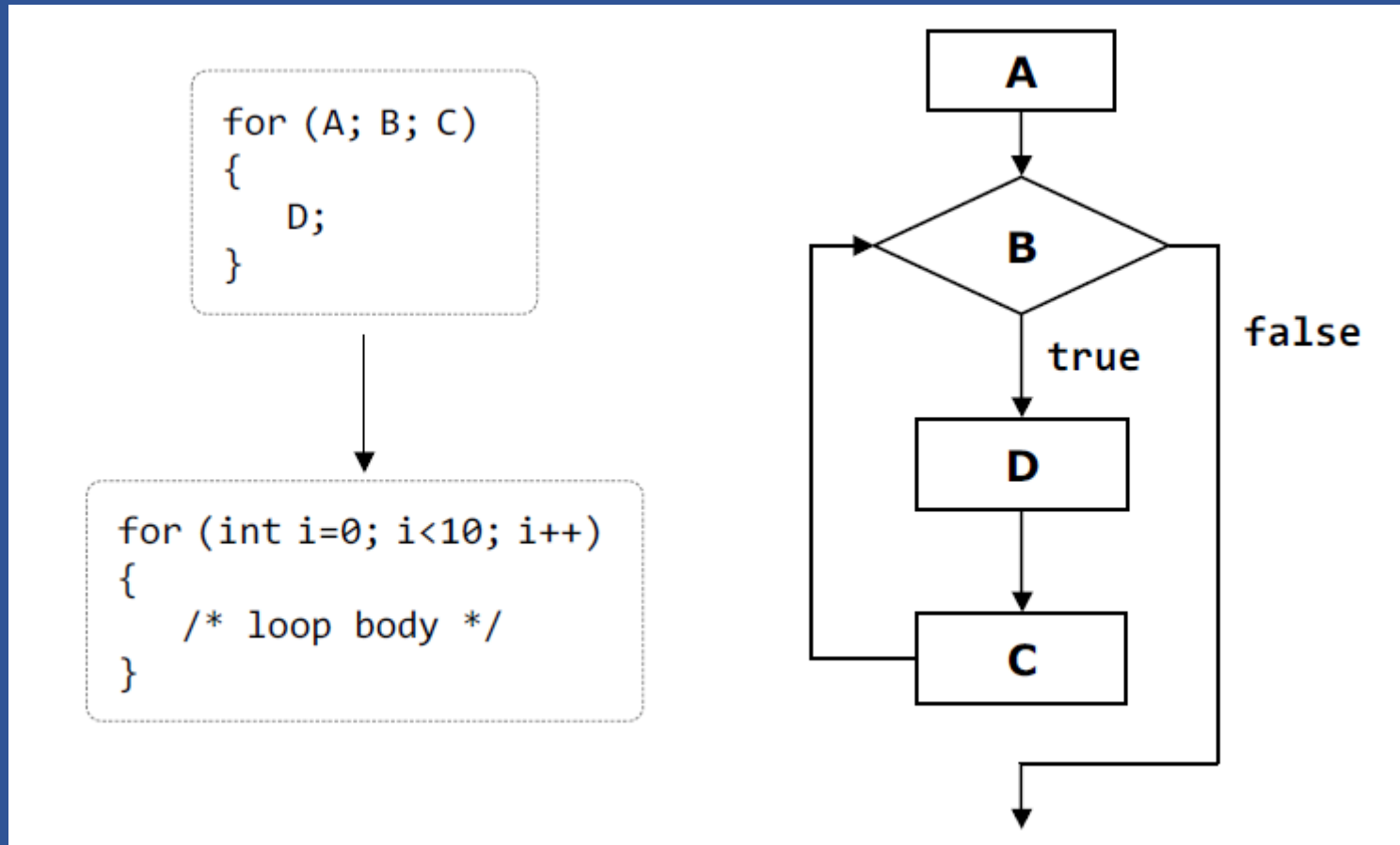
```
// do-while factorial using BigInteger
// Add using System.Numerics
Console.Write("n = ");
int n = int.Parse(Console.ReadLine());
BigInteger factorial = 1;
do
{
    factorial *= n;
    n--;
} while (n > 0);
Console.WriteLine("n! = " + factorial);
```

```
n = 100
n! = 9332621544394415268169923885626670049071596826
438162146859296389521759999322991560894146397615651
828625369792082722375825118521091686400000000000000
0000000000
Press any key to continue . . .
```


do-while example Product in the Range [N...M]

```
// do-while example
// Product in the Range [N...M]
Console.Write("n = ");
int n = int.Parse(Console.ReadLine());
Console.Write("m = ");
int m = int.Parse(Console.ReadLine());
int num = n;
long product = 1;
do
{
    product *= num;
    num++;
} while (num <= m);
Console.WriteLine("product[n...m] = " + product);
//n = 2
//m = 6
//product[n...m] = 720
```

For Loops



Calculating N^M

```
// Calculating N^M
Console.Write("n = ");
int n = int.Parse(Console.ReadLine());
Console.Write("m = ");
int m = int.Parse(Console.ReadLine());
decimal result = 1;
for (int i = 0; i < m; i++)
{
    result *= n;
}
Console.WriteLine("n^m = " + result);
//n = 2
//m = 10
//n ^ m = 1024
```

Operator "continue"

```
// calculate the sum of all odd integers
// in the range [1...n], which are not divisible by 7
int n = int.Parse(Console.ReadLine());
int sum = 0;
for (int i = 1; i <= n; i += 2)
{
    if (i % 7 == 0)
    {
        continue;
    }
    sum += i;
}
Console.WriteLine("sum = " + sum);
```

foreach

```
// int array items iteration
int[] numbers = { 2, 3, 5, 7, 11, 13, 17, 19 };
foreach (int i in numbers)
{
    Console.Write(" " + i);
}
Console.WriteLine();
// string array items iteration
string[] towns = { "London", "Paris", "Milan", "New York" };
foreach (string town in towns)
{
    Console.Write(" " + town);
}
//2 3 5 7 11 13 17 19
//London Paris Milan New York
```

Nested Loop

```
// Nested loop example
// Printing a Triangle
int n = int.Parse(Console.ReadLine());
for (int row = 1; row <= n; row++)
{
    for (int col = 1; col <= row; col++)
    {
        Console.Write(col + " ");
    }
    Console.WriteLine();
}

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

Exercise

1. Print multiplication table

```
Please Enter a number : 2
Multiplication Table for : 2
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
2 x 11 = 22
2 x 12 = 24
Press any key to continue . . .
```











2. Print ASCII Table. Pause every 12 lines

Dec	Hex	Char
33	21	!
34	22	"
35	23	#
36	24	\$
37	25	%
38	26	&
39	27	'
40	28	(
41	29)
42	2A	*
43	2B	+
44	2C	,
-	-	-

3. Play a song

Duration

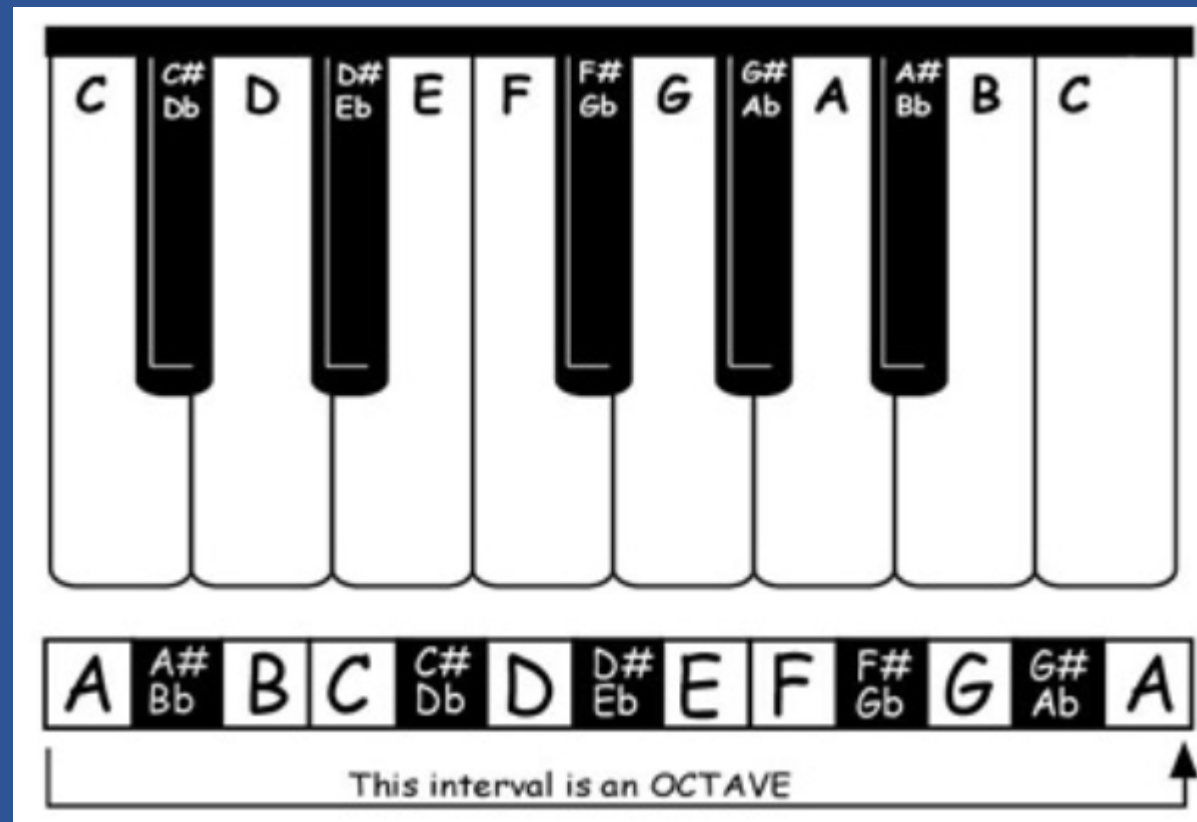
WHOLE = 1600,
 HALF = WHOLE/2,
 QUARTER = HALF/2,
 EIGHTH = QUARTER/2,
 SIXTEENTH = EIGHTH/2,

Whole Note 	4 Counts	Whole Rest 	4 Counts
Half Note 	2 Counts	Half Rest 	2 Counts
Quarter Note 	1 Count	Quarter Rest 	1 Count
Eighth Note 	$\frac{1}{2}$ Count	Eighth Rest 	$\frac{1}{2}$ Count
$\frac{1}{16}^{\text{th}}$ Note 	$\frac{1}{4}$ Count	$\frac{1}{16}^{\text{th}}$ Rest 	$\frac{1}{4}$ Count



musical chromatic scale

REST = 0,
 GbelowC = 196,
 A = 220,
 Asharp = 233,
 B = 247,
 C = 262,
 Csharp = 277,
 D = 294,
 Dsharp = 311,
 E = 330,
 F = 349,
 Fsharp = 370,
 G = 392,
 Gsharp = 415,



Example code to play chromatic scale

```
1  using System;
2  using System.Threading;
3
4  namespace _003_Play_Note
5  {
6      class Program
7      {
8          static void Main(string[] args)
9          {
10             //GbelowC,A,A#,B,C,C#,D,D#,E,F,F#,G,G#
11             int duration = 1600;
12             int[] note = {196,220,233,247,262,277,294,311,330,349,370,392,415};
13             foreach(int n in note)
14             {
15                 Console.Beep(n, duration/2);
16             }
17         }
18     }
19 }
```