Looping

Life is all about Repetition.

We do same thing everyday

What is loop?

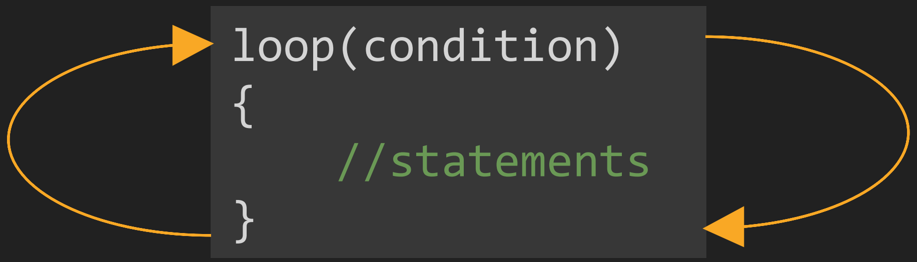
- Loop is used to execute the block of code several times according to the condition given in the loop. It means it executes the same code multiple times.

"Hello" 5

```
printf("Hello\n");
printf("Hello\n");
printf("Hello\n");
printf("Hello\n");
printf("Hello\n");
```

Output

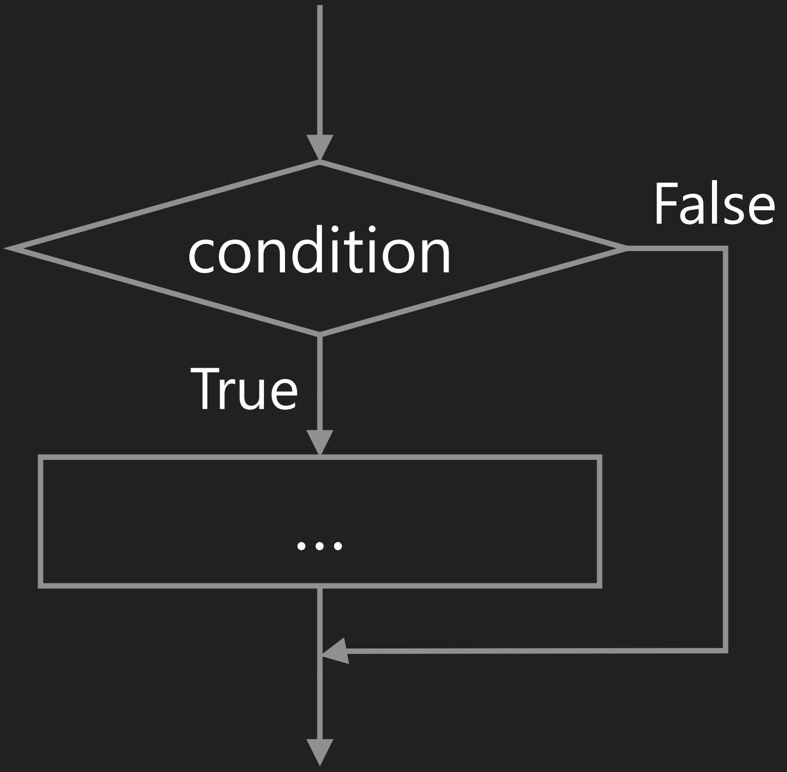
```
Hello
Hello
Hello
Hello
Hello
```



```
loop(condition)
{
    //statements
}
```

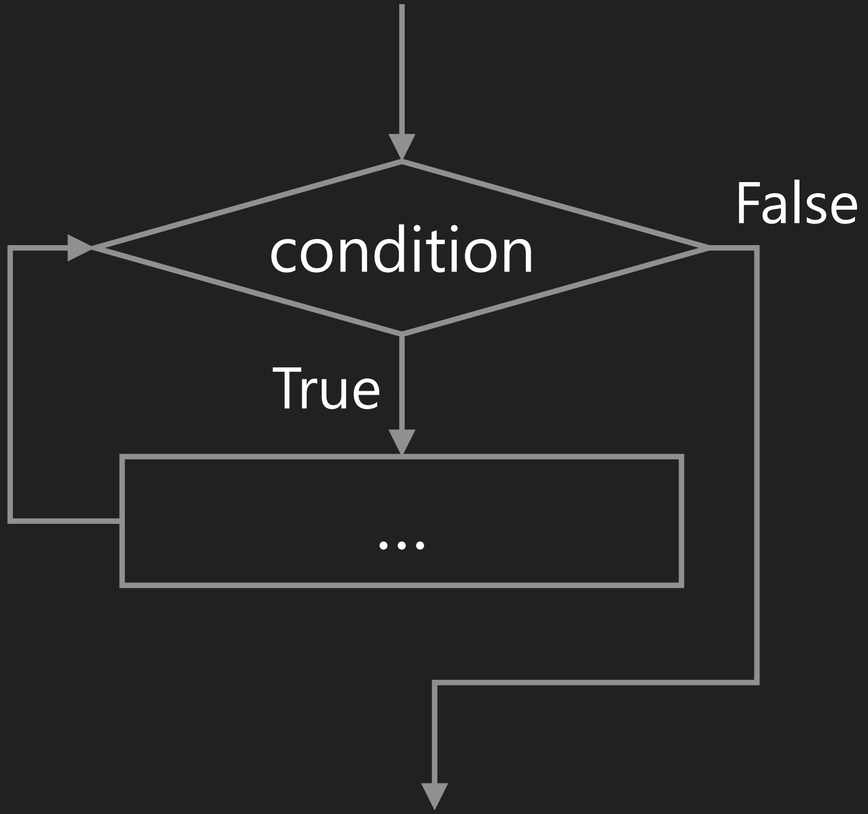
if v/s while

Flowchart of if



v/s

Flowchart of while



Looping or Iterative Statements in C

Looping Statements are

Entry Controlled Loop:	`while`, `for`
Exit Controlled Loop:	`do…while`
Virtual Loop:	`goto`

While loop

While Loop

- while is an entry controlled loop
- Statements inside the body of while are repeatedly executed till the condition is true
- while is keyword

Syntax

```
while(condition)
{
    // Body of the while
    // true part
}
```

WAP to print 1 to n(while loop)

Program

```c
#include <stdio.h>
void main()
{
    int i,n;
    i=1;
    printf("Enter n:");
    scanf("%d",&n);
    while(i<=n)
    {
        printf("%d\n",i);
        i=i+1;
    }
}
```

Output

```
Enter n:10
1
2
3
4
5
6
7
8
9
10
```

WAP to print multiplication table(while loop)

Program

```c
#include<stdio.h>
void main()
{
    int i=1,n;
    printf("Enter n for multiplication table:");
    scanf("%d",&n);
    while(i<=10)
    {
        printf("%d * %d = %d\n",n,i,n*i);
        i=i+1;
    }
}
```

Output

```
Enter n for multiplication table:5
5 * 1  = 5
5 * 2  = 10
5 * 3  = 15
5 * 4  = 20
5 * 5  = 25
5 * 6  = 30
5 * 7  = 35
5 * 8  = 40
5 * 9  = 45
5 * 10 = 50
```

WAP to Sum of 5 numbers entered by user(while loop)

Program

```c
#include<stdio.h>
void main()
{
    int sum=0, i=1,n;
    while(i<=5)
    {
        printf("Enter a number=");
        scanf("%d",&n);
        sum=sum+n;
        i=i+1;
    }
    printf("Sum is=%d",sum);
}
```

Output

```
Enter a number=10
Enter a number=20
Enter a number=30
Enter a number=40
Enter a number=50
Sum is=150
```

Syntax and Logic

Swimming Rules

1. Breath control
2. Kicking legs
3. Back stroke with arms
4. Front stroke with arms
5. Crawling in water

To Swim

Syntax

```
while(condition)
{
    // Body of the while
    // true part
}
```

Logic

```
int i = 1;
while (i <= 5)
{
    printf("%d\n", i);
    i=i+1;
}
```

How to build logic? Step-1

Step 1: Understand the problem statement

- e.g. Write a program to find factors of a number.
- Run following questions through mind

- What is the factor of a number?
  - Factor is a number that divides another number evenly with no remainder.
  - For example, 1,2,3,4,6,12 are factors of 12.
- How many variables needed? What should be their data types?(Inputs/Outputs)
  - To get number from user we need variable **`n`**.
  - Now we need to divide **`n`** with 1,2,3,...,n. For this we will declare a loop variable **`i`** initialized as 1.
  - Both variables should be of `integer` data type.
- What control structure you require?
  - First we need `a loop` to divide **`n`** by 1,2,3,...,n, loop will start from 1 and ends at **`n`**.
  - Inside loop we need `if structure` to check **`n%i==0`** (Number n is evenly divisible by **`i`** or not).

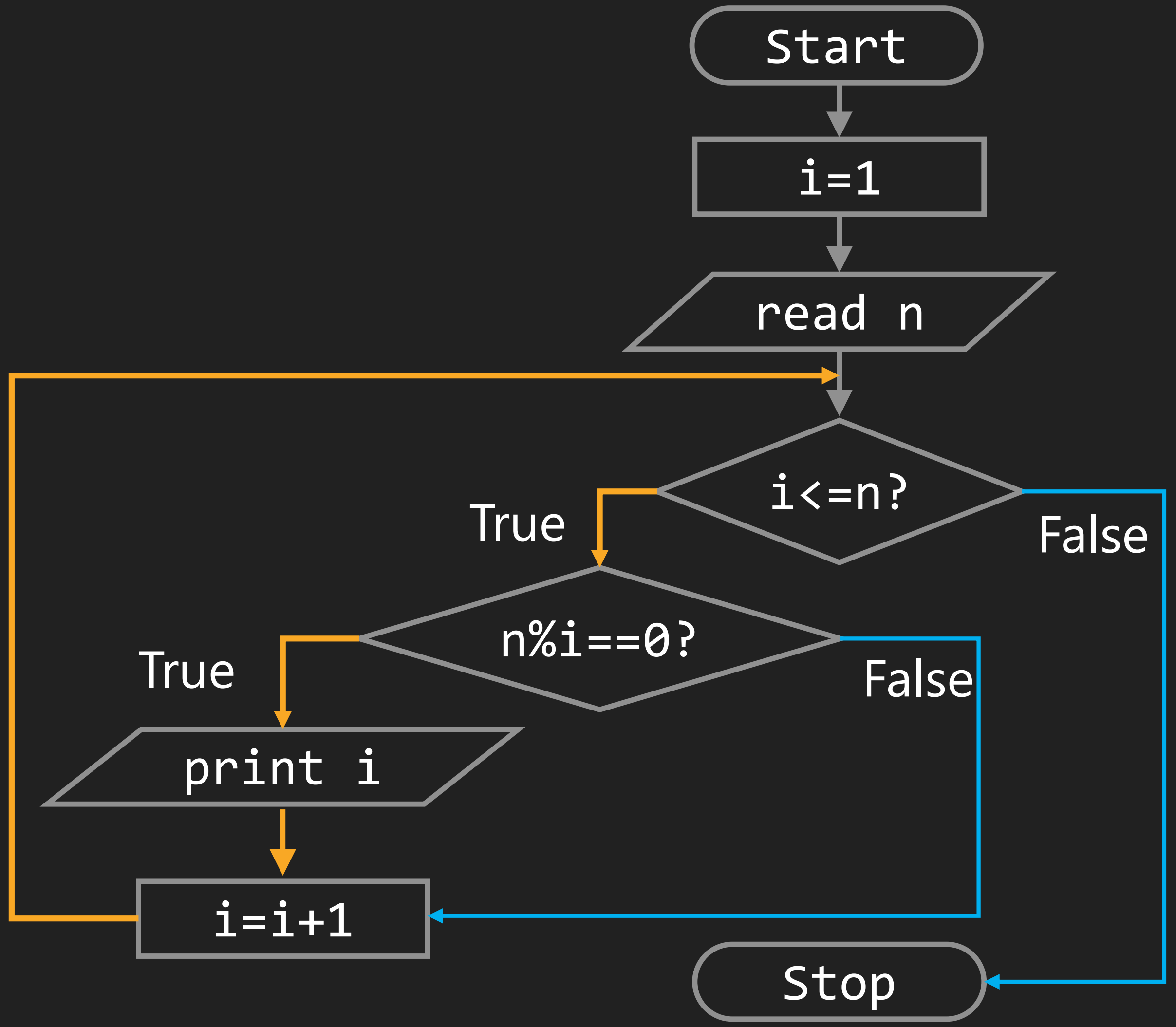
How to build logic? Step-2

Step 2: Think for 1 or 2 examples

- Consider n=6, now take i=1
  - 6%1==0, TRUE; So, 1 is factor of 6
  - 6%2==0, TRUE; So, 2 is factor of 6
  - 6%3==0, TRUE; So, 3 is factor of 6
  - 6%4==2, FALSE; S0, 4 is not factor of 6
  - 6%5==1, FALSE; S0, 5 is not factor of 6
  - 6%6==0, TRUE; S0, 6 is factor of 6
- From this we can infer that loop variable `i` starts with 1 and incremented by one for next iteration then ends at value `n`.
- Consider n=10, factors are 1,2,5,10
- Consider n=11, factor is 1,11
- From this we can infer that 1 and number itself are always factors of any number `n`.

How to build logic? Step-3

Step 3: Draw flowchart/steps on paper or in mind



Steps

```
Step 1: Start
Step 2: Declare variables n,i
Step 3: Initialize variable
        i ← 1
Step 4: Read value of n
Step 5: Repeat the steps until i = n
    5.1: if n%i == 0
            Display i
    5.2: i=i+1
Step 7: Stop
```

How to build logic? Step-4

Step 4: Writing Pseudo-code

- Pseudo-code is an informal way to express the design of a computer program or an algorithm.
- It does not require any strict programming language syntax.

Pseudo-code

```
Initialize i=1 integer
Declare n as integer
Input n
while i<n
    if n%i
        print i
    end if
    increment i=i+1
end while
```

WAP to find factors of a number(while loop)

Program

```c
#include <stdio.h>
void main()
{
    int i=1,n;
    printf("Enter n to find factors=");
    scanf("%d",&n);
    while(i<=n)
    {
        if(n%i==0)
            printf("%d,",i);
        i=i+1;
    }
}
```

Output

```
Enter n to find factors=12
1,2,3,4,6,12,
```

WAP to print reverse a number(while loop)

Program

```c
#include <stdio.h>
void main()
{
    int n;
    printf("Enter a number=");
    scanf("%d",&n);
    while(n!=0)
    {
        printf("%d",n%10);
        n=n/10;
    }
}
```

Output

```
Enter a number=1234
4321
```

WAP to check given number is perfect or not(while loop)

```c
void main(){
    int i=1,n,sum=0;
    printf("Enter a number:");
    scanf("%d",&n);
    while(i<n)
    {
        if(n%i==0)
        {
            printf("%d+",i);
            sum=sum+i;
        }
        i=i+1;
    }
    printf("=%d",sum);
    if(sum==n)
        printf("\n%d is a perfect number",n);
    else
        printf("\n%d is not a perfect number",n);
}
```

Output

```
Enter a number:6
1+2+3=6
6 is a perfect number
```

Output

```
Enter a number:8
1+2+4+=7
8 is not a perfect number
```

Output

```
Enter a number:496
1+2+4+8+16+31+62+124+248+=496
496 is a perfect number
```

WAP to check given number is prime or not(while loop)

```
void main()
{
    int n, i=2,flag=0;
    printf("Enter a number:");
    scanf("%d",&n);
    while(i<=n/2)
    {
        if(n%i==0)
        {
            flag=1;
            break;
        }
        i++;
    }
    if (flag==0)
        printf("%d is a prime number",n);
    else
        printf("%d is not a prime number",n);
}
```

Output

```
Enter a number:7
7 is a prime number
```

Output

```
Enter a number:9
9 is not a prime number
```

for loop

for Loop

- for is an entry controlled loop
- Statements inside the body of for are repeatedly executed till the condition is true
- for is keyword

Syntax

```
for (initialization; condition; updateStatement)
{
    // statements
}
```

- The initialization statement is executed only once.
- Then, the condition is evaluated. If the condition is false, the for loop is terminated.
- If the condition is true, statements inside the body of for loop are executed, and the update statement is updated.
- Again the condition is evaluated.

WAP to print numbers 1 to n (for loop)

Program

```c
#include<stdio.h>
void main()
{
    int i,n;
    printf("Enter a number:");
    scanf("%d",&n);
    for(i=1;i<=n;i++)
    {
        printf("%d\n",i);
    }
}
```

Output

```
Enter a number:5
1
2
3
4
5
```

WAP to find factors of a number (for loop)

Program

```c
#include <stdio.h>
void main()
{
    int i,n;
    printf("Enter n to find factors=");
    scanf("%d",&n);
    for(i=1;i<=n;i++)
    {
        if(n%i==0)
            printf("%d,",i);
    }
}
```

Output

```
Enter n to find factors=12
1,2,3,4,6,12,
```

WAP to check given number is perfect or not(for loop)

```c
void main(){
    int i,n,sum=0;
    printf("Enter a number:");
    scanf("%d",&n);
    for(i=1;i<n;i++)
    {
        if(n%i==0)
        {
            printf("%d+",i);
            sum=sum+i;
        }
    }
    printf("=%d",sum);
    if(sum==n)
        printf("\n%d is a perfect number",n);
    else
        printf("\n%d is not a perfect number",n);
}
```

Output

```
Enter a number:6
1+2+3=6
6 is a perfect number
```

Output

```
Enter a number:8
1+2+4+=7
8 is not a perfect number
```

Output

```
Enter a number:496
1+2+4+8+16+31+62+124+248+=496
496 is a perfect number
```

do while loop

do while Loop

- do while is an exit controlled loop.
- Statements inside the body of do while are repeatedly executed till the condition is true.
- Do and while are keywords.

Syntax

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

- Loop body will be executed first, and then condition is checked.
- If the condition is true, the body of the loop is executed again and the condition is evaluated.
- This process goes on until the condition becomes false.
- If the condition is false, the loop ends.

WAP to print Odd numbers between 1 to n(do while loop)

Program

```c
void main()
{
    int i=1,n;
    printf("Enter a number:");
    scanf("%d",&n);
    do
    {
        if(i%2!=0)
        {
            printf("%d,",i);
        }
        i=i+1;
    }
    while(i<=n);
}
```

Output

```
Enter a number:5
1,3,5
```

WAP to find factors of a number(do while loop)

Program

```c
void main()
{
    int i=1,n;
    printf("Enter a number:");
    scanf("%d",&n);
    do
    {
        if(n%i==0)
        {
            printf("%d,",i);
        }
        i=i+1;
    }
    while(i<=n);
}
```

Output

```
Enter a number:6
1,2,3,6,
```

WAP to print reverse a number(do while loop)

Program

```c
void main()
{
    int n;
    printf("Enter a number:");
    scanf("%d",&n);
    do
    {
        printf("%d",n%10);
        n=n/10;
    }
    while(n!=0);
}
```

Output

```
Enter a number=1234
4321
```

goto statement

goto Statement

- goto is an virtual loop
- The goto statement allows us to transfer control of the program to the specified label.
- goto is keyword

Syntax

```
goto label;
.
.
.
label:
```

Syntax

```
label:
.
.
.
goto label;
```

- The label is an identifier. When the goto statement is encountered, the control of the program jumps to label: and starts executing the code.

WAP to print Odd numbers between 1 to n(goto)

Program

```c
void main()
{
    int i=1,n;
    printf("Enter a number:");
    scanf("%d",&n);
    odd:
    if(i%2!=0)
    {
        printf("%d,",i);
    }
    i=i+1;
    if(i<=n)
    {
        goto odd;
    }
}
```

Output

```
Enter a number:5
1,3,5
```

WAP to find factors of a number(goto)

Program

```c
void main()
{
    int i=1,n;
    printf("Enter a number:");
    scanf("%d",&n);
    odd:
    if(n%i==0)
    {
        printf("%d,",i);
    }
    i=i+1;
    if(i<=n)
    {
        goto odd;
    }
}
```

Output

```
Enter a number:6
1,2,3,6,
```

Types of loops

Entry Control Loop

```
int i=1;
while(i<=10)
{
    printf("%d",i++);
}
```

Entry Control Loop

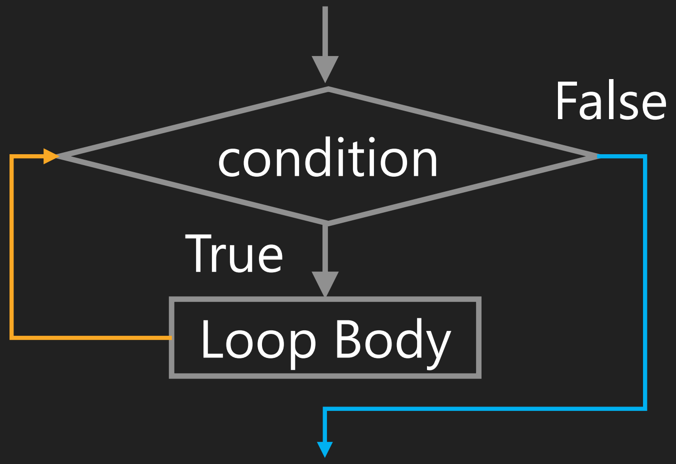
```
int i;
for(i=1;i<=10;i++)
{
    printf("%d",i);
}
```

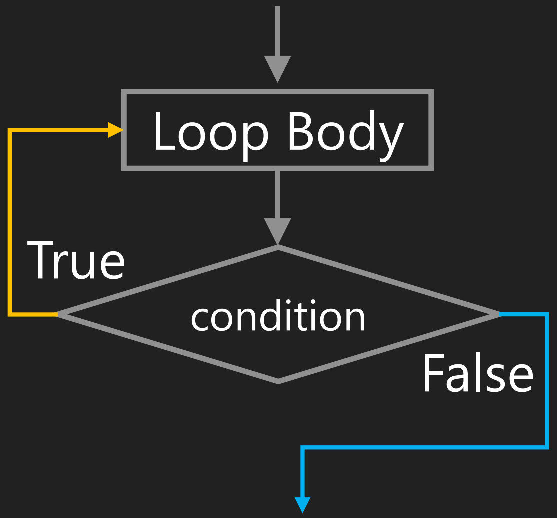
Exit Control Loop

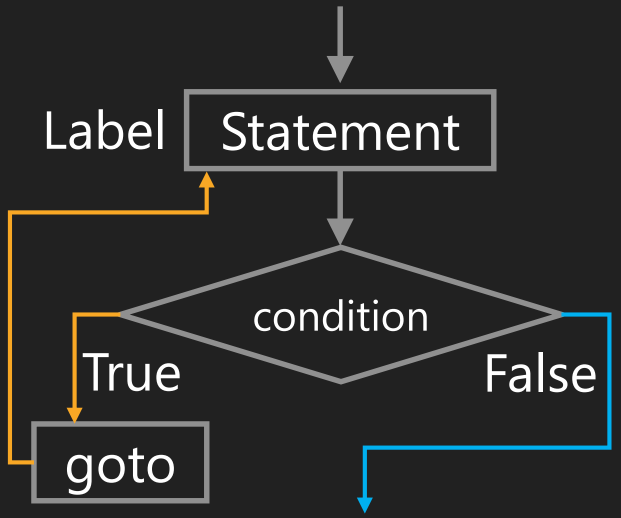
```
int i=1;
do
{
    printf("%d",i++);
}
while(i<=10);
```

Virtual Loop

```
int i=1;
   labelprint:
    printf("%d",i++);
   if(i<=10)
     goto labelprint;
```







Pattern

Always detect pattern in pattern

Pattern

There are important points to note in pattern

1. Determine, how many rows?
2. Determine, how many numbers/characters/columns in a row?
3. Determine, Increment/Decrement among the number of rows.
4. Determine, starting in each row

```
1
11
111
1111
11111
```

```
1
12
123
1234
12345
```

```
1
23
456
78910
```

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

WAP to print given pattern (nested loop)

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

No. of rows: 5

No. of characters
Row-1: *
Row-2: **
Row-3: ***
Row-4: ****
Row-5: *****

Inner loop: Increment
Outer loop: Increment

Starting: *

Program

```
void main()
{
    int i,j;
    for(i=1;i<=5;i++)
    {
        for(j=1; j<=i; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

WAP to print given pattern (nested loop)

```
1
12
123
1234
12345
```

No. of rows: 5

No. of values
Row-1: 1
Row-2: 12
Row-3: 123
Row-4: 1234
Row-5: 12345

Inner loop: Increment
Outer loop: Increment

Starting: 1

Program

```c
void main()
{
    int i,j;
    for(i=1;i<=5;i++)
    {
        for(j=1; j<=i; j++)
        {
            printf("%d",j);
        }
        printf("\n");
    }
}
```

WAP to print given pattern (nested loop)

```
5
54
543
5432
54321
```

No. of rows: 5

No. of values
Row-1: 5
Row-2: 54
Row-3: 543
Row-4: 5432
Row-5: 54321

Inner loop: Decrement
Outer loop: Decrement/Increment

Starting: 5

Program

```
void main()
{
     int i,j;
     for(i=5;i>0;i--)
     {
          for(j=5; j>=i ; j--)
          {
               printf("%d",j);
          }
          printf("\n");
     }
}
```

WAP to print given pattern (nested loop)

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

No. of rows: 5

No. of values
Row-1: ----*
Row-2: ---**
Row-3: --***
Row-4: -****
Row-5: *****

Inner loop: Decrement
Outer loop: Decrement/Increment

Starting: -(space)
Ending: *

Program

```c
void main()
{
    int i,j,k;
    for(i=1;i<=5;i++)
    {
        for(k=5;k>i;k--)
        {
            printf(" ");
        }
        for(j=1;j<=i;j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

First we need to print 4 spaces before printing *

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

After printing spaces this inner loop prints *

Practice programs

1) Write a program to find sum of first N odd numbers. Ex. 1+3+5+7+……….+N
2) Write a program to find 1+1/2+1/3+1/4+….+1/n.
3) Write a program to print all Armstrong numbers in a given range. For example 153 = 1^3 + 5^3 + 3^3. So, 153 is Armstrong number.
4) Write a program to print given number in reverse order
5) Write a program to check whether a given string is palindrome or not.
6) Write a program to print Multiplication Table up to n.

```
1    2    3    4    5    6    7    .
2    4    6    8    10   12   14   .
3    6    9    12   15   18   21   .
4    8    12   16   20   24   28   .
5    10   15   20   25   30   35   .
.    .    .    .    .    .    .    .
```

7) Construct C programs to print the following patterns using loop statement.

```
1
22
333
4444
55555
```

```
*
# #
* * *
# # # #
* * * * *
```

```
1
0 1
1 0 1
0 1 0 1
```

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

```
    1
   A B
  2 3 4
 C D E F
```

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

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

Thank you