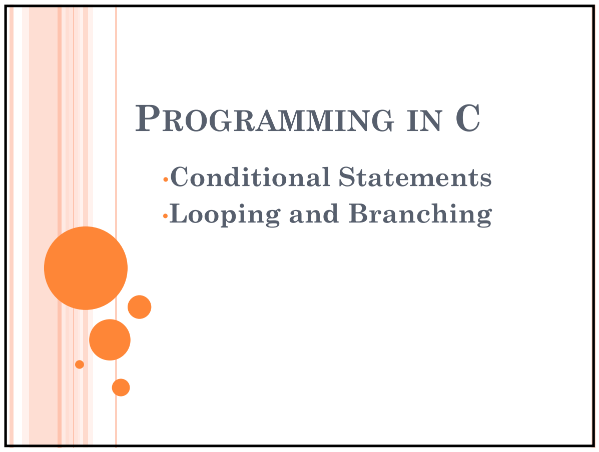
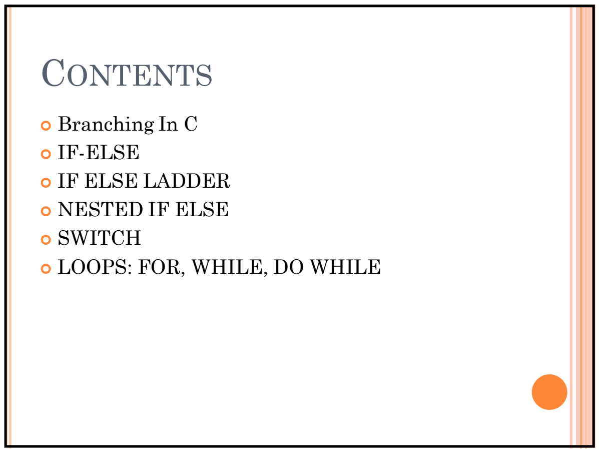


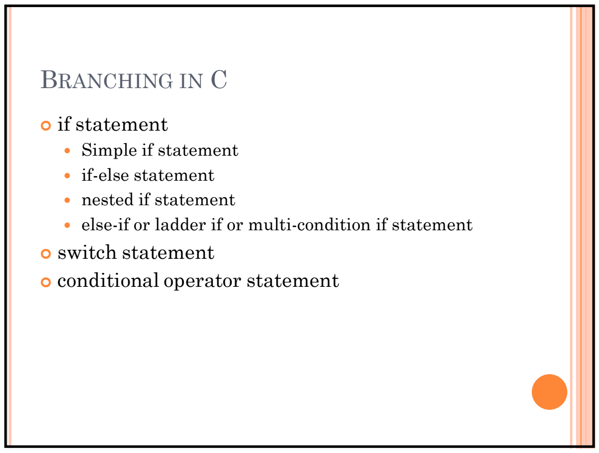
Programming in C

- Conditional Statements
- Looping and Branching



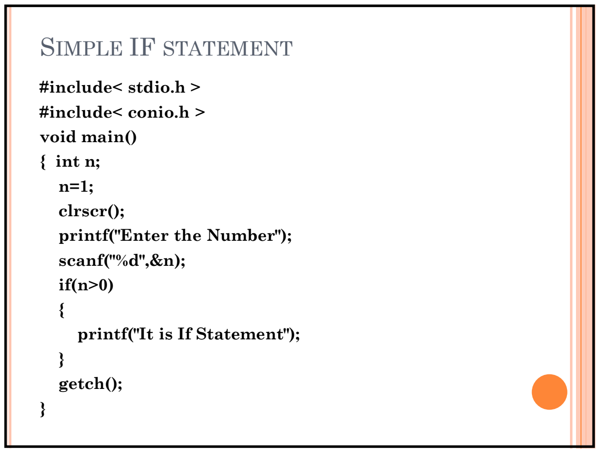
Contents

- Branching In C
- IF-ELSE
- IF ELSE LADDER
- NESTED IF ELSE
- SWITCH
- LOOPS: FOR, WHILE, DO WHILE



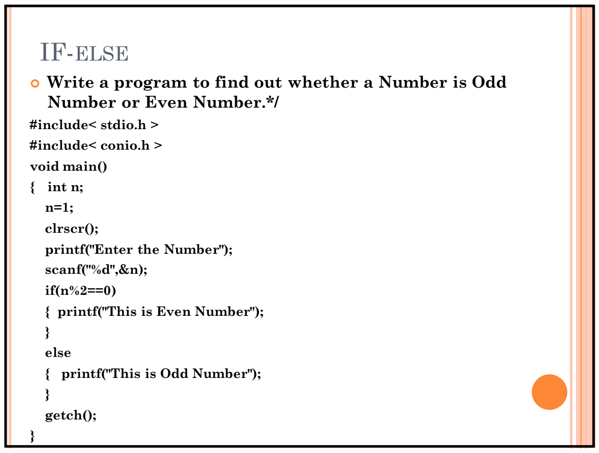
Branching in C

- if statement
  - Simple if statement
  - if-else statement
  - nested if statement
  - else-if or ladder if or multi-condition if statement
- switch statement
- conditional operator statement



Simple IF statement

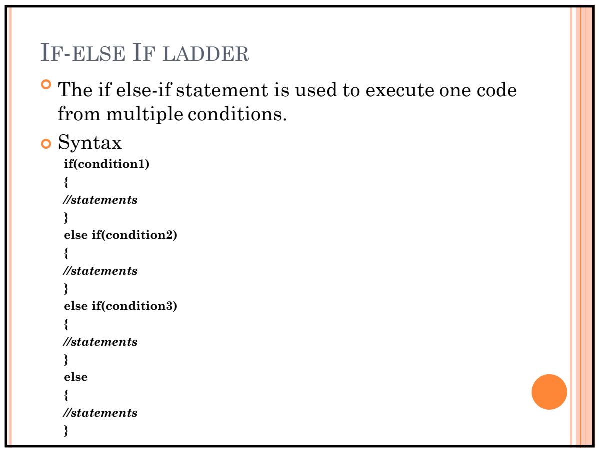
```
#include< stdio.h >
#include< conio.h >
void main()
{  int n;
   n=1;
   clrscr();
   printf("Enter the Number");
   scanf("%d",&n);
   if(n>0)
   {
      printf("It is If Statement");
   }
   getch();
}
```



IF-ELSE

- **Write a program to find out whether a Number is Odd Number or Even Number.*/**

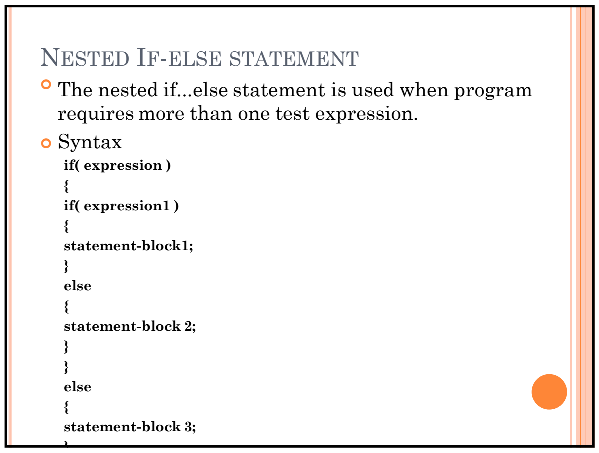
```
#include< stdio.h >
#include< conio.h >
void main()
{   int n;
    n=1;
    clrscr();
    printf("Enter the Number");
    scanf("%d",&n);
    if(n%2==0)
    {  printf("This is Even Number");
    }
    else
    {   printf("This is Odd Number");
    }
    getch();
}
```



IF-ELSE IF LADDER

- The if else-if statement is used to execute one code from multiple conditions.
- Syntax

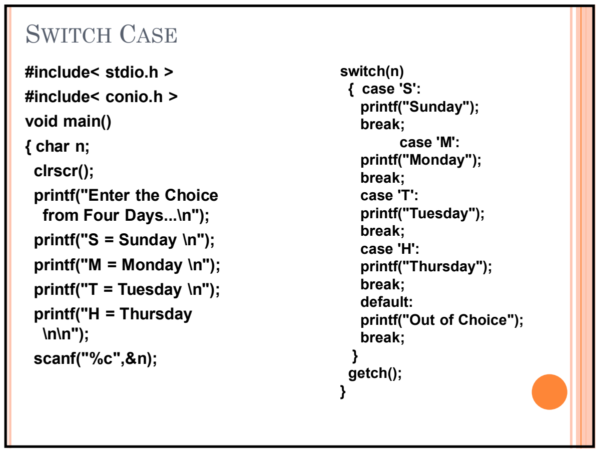
```
if(condition1)
{
//statements
}
else if(condition2)
{
//statements
}
else if(condition3)
{
//statements
}
else
{
//statements
}
```



NESTED IF-ELSE STATEMENT

- The nested if...else statement is used when program requires more than one test expression.
- Syntax

```
if( expression )
{
if( expression1 )
{
statement-block1;
}
else
{
statement-block 2;
}
}
else
{
statement-block 3;
```



SWITCH CASE

```
#include< stdio.h >
#include< conio.h >
void main()
{ char n;
  clrscr();
  printf("Enter the Choice
   from Four Days...\n");
  printf("S = Sunday \n");
  printf("M = Monday \n");
  printf("T = Tuesday \n");
  printf("H = Thursday
   \n\n");
  scanf("%c",&n);
switch(n)
  {  case 'S':
     printf("Sunday");
     break;
           case 'M':
     printf("Monday");
     break;
     case 'T':
     printf("Tuesday");
     break;
     case 'H':
     printf("Thursday");
     break;
     default:
     printf("Out of Choice");
     break;
   }
  getch();
}
```

Ternary Statement (conditional operator)

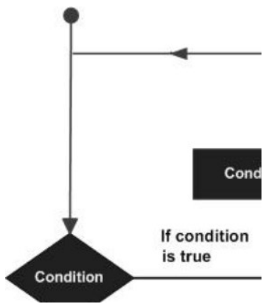
- Ternary statement or Ternary operator is like if-else statement in its functioning. It can be represented with **? :** . It is also called as **conditional operator**
- For example:

z = a > b ? a : b;

- **a<b ? printf("a is less") : printf("a is greater");**

Loops in C

- A loop statement allows us to execute a statement or group of statements multiple times.



Types Of Loops

Sr.No.	Loop Type & Description
1	**while loop**: Repeats a statement or group of statements while a given condition is true. It tests the condition before executing the loop body.
2	**for loop**: Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.
3	**do...while loop**: It is more like a while statement, except that it tests the condition at the end of the loop body.
4	**nested loops**: You can use one or more loops inside any other while, for, or do..while loop.

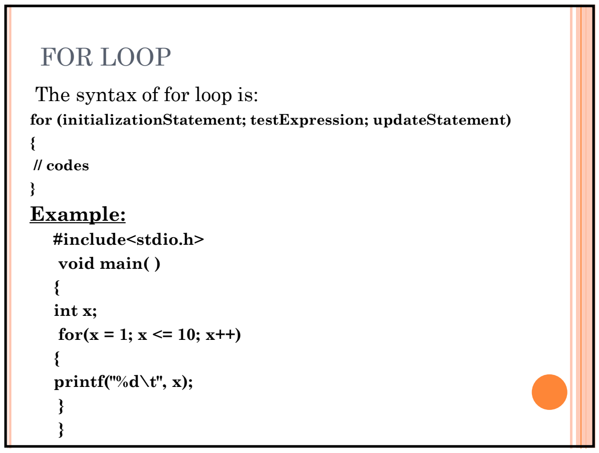
While Loop

While loop can be addressed as an **entry control** loop. It is completed in 3 steps.

- Variable initialization.(e.g int x = 0;)
- condition(e.g while(x <= 10))
- Variable increment or decrement (x++ or x-- or x = x + 2)

Example:

```
#include<stdio.h>
void main( )
{ int x;
x = 1;
while(x <= 10)
{
  printf("%d\t", x);
  x++;
}
}
```



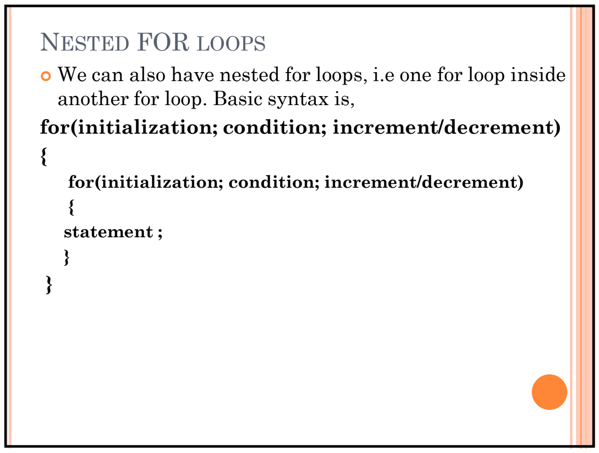
FOR LOOP

The syntax of for loop is:

```
for (initializationStatement; testExpression; updateStatement)
{
// codes
}
```

Example:

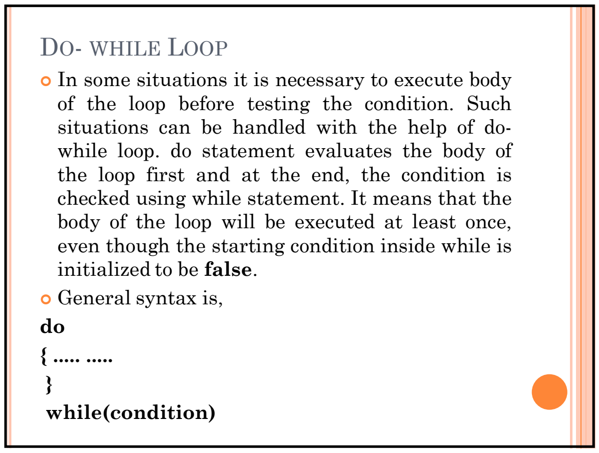
```
   #include<stdio.h>
    void main( )
   {
   int x;
    for(x = 1; x <= 10; x++)
   {
   printf("%d\t", x);
    }
    }
```



NESTED FOR LOOPS

- We can also have nested for loops, i.e one for loop inside another for loop. Basic syntax is,

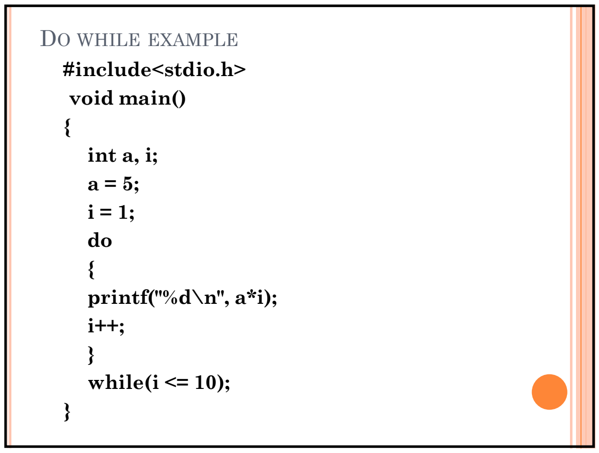
```
for(initialization; condition; increment/decrement)
{
    for(initialization; condition; increment/decrement)
    {
   statement ;
    }
 }
```



DO- WHILE LOOP

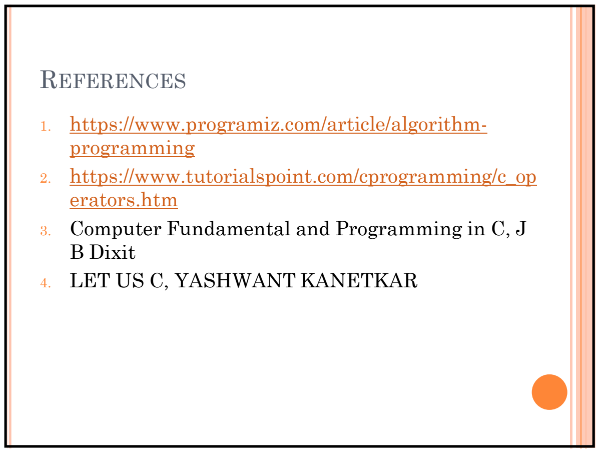
- In some situations it is necessary to execute body of the loop before testing the condition. Such situations can be handled with the help of do-while loop. do statement evaluates the body of the loop first and at the end, the condition is checked using while statement. It means that the body of the loop will be executed at least once, even though the starting condition inside while is initialized to be **false**.
- General syntax is,

```
do
{ ..... .....
 }
 while(condition)
```



DO WHILE EXAMPLE

```
 #include<stdio.h>
  void main()
 {
    int a, i;
    a = 5;
    i = 1;
    do
    {
    printf("%d\n", a*i);
    i++;
    }
    while(i <= 10);
 }
```



REFERENCES

1. https://www.programiz.com/article/algorithm-programming
2. https://www.tutorialspoint.com/cprogramming/c_operators.htm
3. Computer Fundamental and Programming in C, J B Dixit
4. LET US C, YASHWANT KANETKAR