CONTROL STRUCTURES

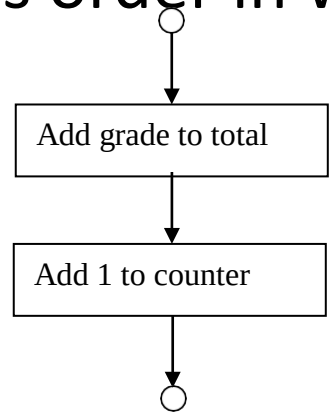
What are a control structures?

- These are programming statements that control the flow of execution in a program. They are organized into three kinds of control structures.
  - Sequence
  - Selection
  - Repetition/Iteration/Looping

Sequence control structure

- The sequence structure is essentially built into Unless directed otherwise, the computer automatically executes C statements one after the other in the order in which they are written.
- NB: Flowlines indicates order in which actions are performed.

total=total +grade;

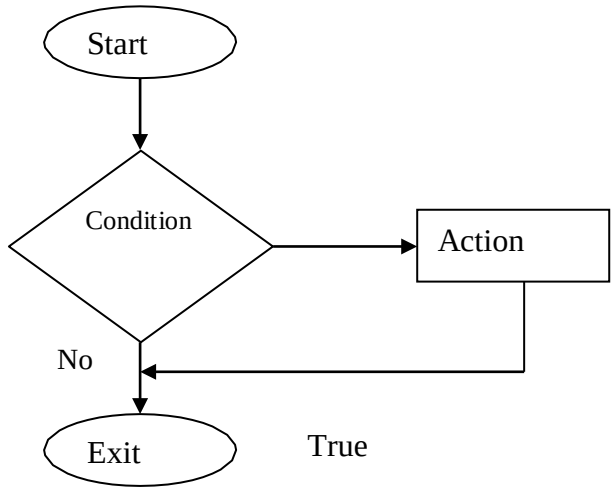


Selection Control Structure

- They are used to choose among alternatives causes of actions. They include:
  i. The if selection statement(Single selection statement)
  ii. The if……else selection statement(Double-selection statement)
  iii. Switch statement(Multiple-selection statement)

The If…Selection

- It performs (selects) an action if a condition is true or skips the action if the condition is false.



Syntax

```
If (condition)
{
        Statement to be executed
}
```

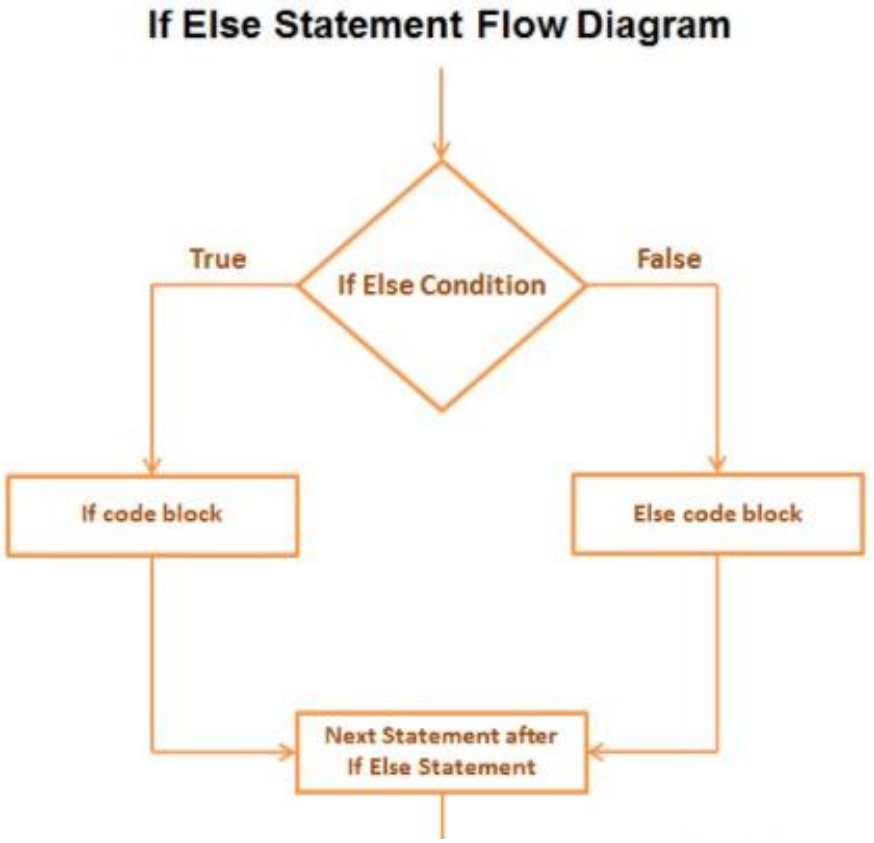
Example 1

- Write a program to classify a student as pass if they get above 40 marks.

```
#include<stdio.h> int main()
{
int grade;
printf("Enter grade\n");
scanf("%d", &grade);
if(grade>=40)
{
printf("Passed\n");
}
return 0;
}
```

The if…else selection statement

- It performs an indicated action only when the condition is true; otherwise it executes the else block.



Syntax

If (condition)

{

Statement to execute if condition is true

}

Else

{

Statement to execute if condition is not true

}

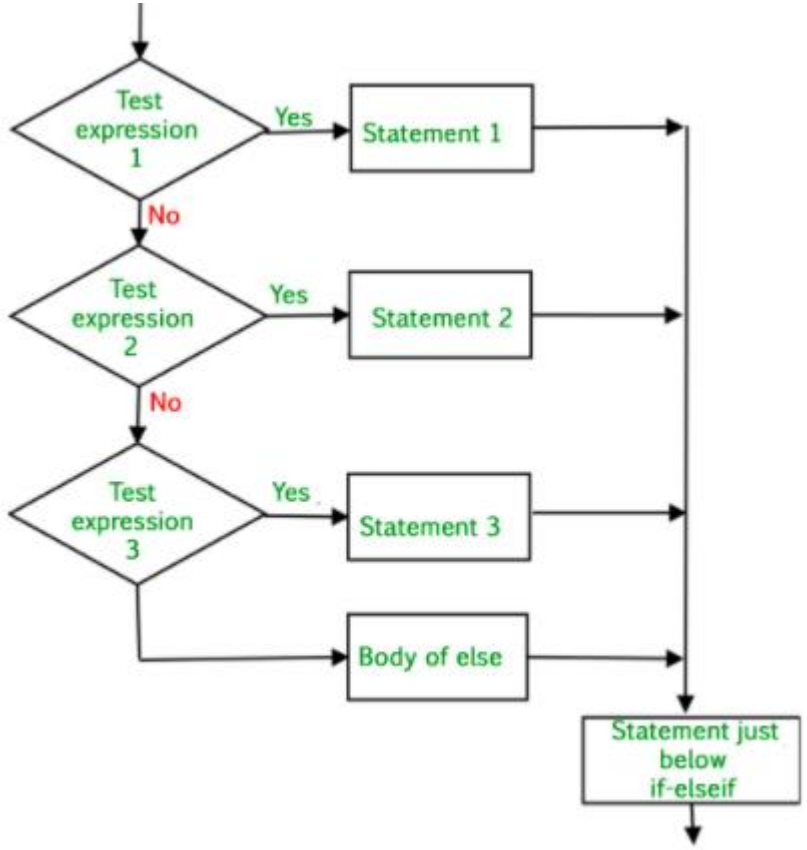
Example

- Write a program to classify a student as pass if they get above 40 marks and fail if they get anything else.

```
#include<stdio.h> int main()
{
int grade;
printf("Enter grade\n");
scanf("%d", &grade);
if(grade>=40)
{
printf("Passed\n");
}
Else {printf(“Failed\n"); }
return 0;
}
```

Nested if…else Statement

- It tests for multiple cases by placing if….else statements inside if….else statements.



Example

Write a program to grade students based on the following criteria.

Below 40- fail

40-49- D

50-59- C

60-69- B

70-100 A

Above 100- Invalid

Syntax

```
If(condition 1)
{
Statement to execute
}
Else if(Condition 2)
{
Statement to execute
}
Else if(condition 3)
{
Statement to execute
}
```

```
#include<stdio.h>
int main()
{
int marks;
printf("Enter grade\n");
scanf("%d", &marks);
if ((marks>=0) &&(marks<=39))
{
printf("Grade is Fail\n");
}
else if((marks>=40) &&(marks<=49))
{
printf("Grade is D\n");
```

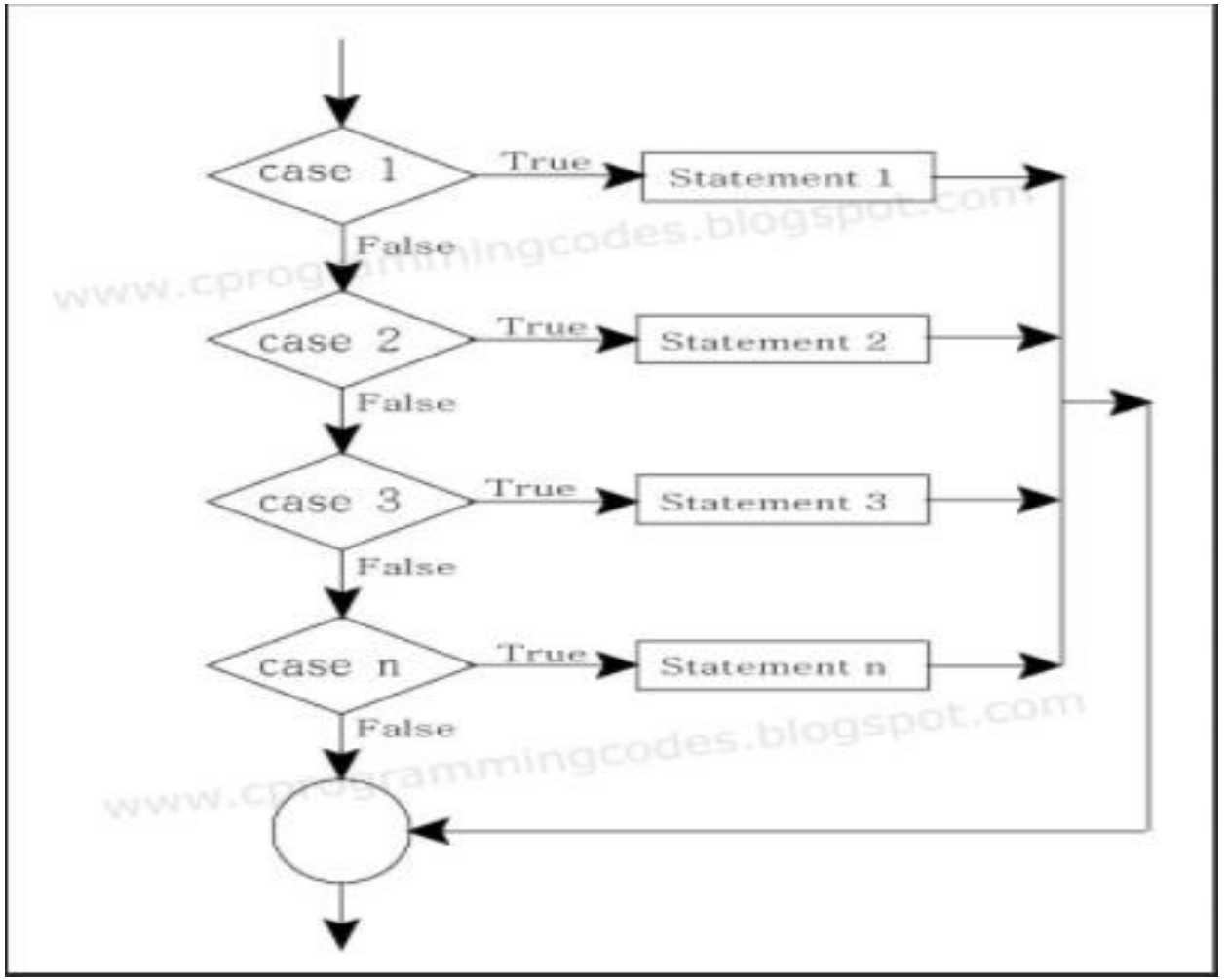
```
}
else if((marks>=50) &&(marks<=59))
{
printf("Grade is C\n");
}
else if((marks>=60) &&(marks<69))
{
printf("Grade is B\n");
else if((marks>=70) &&(marks<100))
{
printf("Grade is A\n");
}
else
```

```
{
printf("Invalid entry");
}
return 0;
}
```

Switch multiple selection statement

- An algorithm contain a series of decisions in which a variable or expression is tested separately for each of the constant integral values it may assume and different actions are taken. This is called multiple selection.
- Each case can have one or more actions.
- The switch statement consists of a series of case labels, and an optional default case.
-

Select…Case flowchart



```
Switch(expression)
{
Case (expression1):
{
One or more C statements;
}
Case (expression2):
{
One or more C statements;
Default:
{
One or more C statements;
}
}
```

- If the block is only one statement long, you do not need the braces, but they are recommended.
- The default line is optional and it doesn’t have to be the last line of the switch body.

Example

```
#include <stdio.h>
   int main() {
      int num;
      printf("Enter Number\n");
scanf("%d", &num);

      switch (num) {
         case 7:
            printf("Value is 7");
            break;
```

```
case 8:
            printf("Value is 8");
            break;
        case 9:
            printf("Value is 9");
            break;
        default:
            printf("Out of range");
            break;
    }
    return 0;
}
```

Iteration/Looping Structures

- A loop is a group of instructions the computer executes repeatedly while some loop- continuation condition remains true.

Types of loops

- Counter-controlled repetition-Also called definite repetition because we know in advance exactly how many times the loop will be executed.e.g for loop
- Sentinel-uncontrolled repetition-Also called indefinite repetition because its not known in advance how many times the loop will be executed e.g do…while and while

Repetition structures include

- while
- Do…while
- for loop

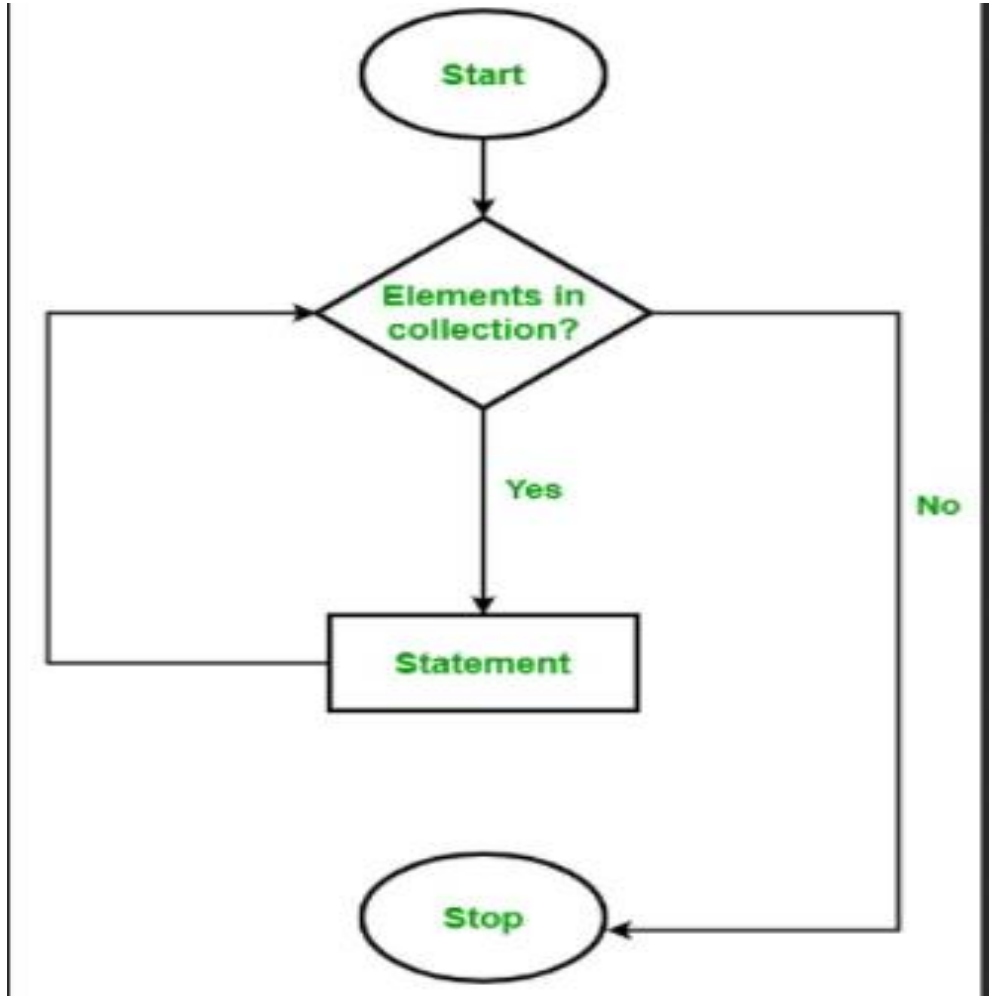
for loop

Format of the for loop:

for (start expression; test expression; count expression)
{
Block of one or more c statements;
}

- Start expression is an assignment statement (such as ctr=1;)
-
- Test expression evaluates to true or false,and then determines if the body of the loop repeats again.(loop condition)
- Count expression usually increments and decrements

For flow chart



Example

- The following program uses for loop to print 1-10

```
#include <stdio.h>
Int main()
{
   int i;

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

While Loop

Format

While (test expression)

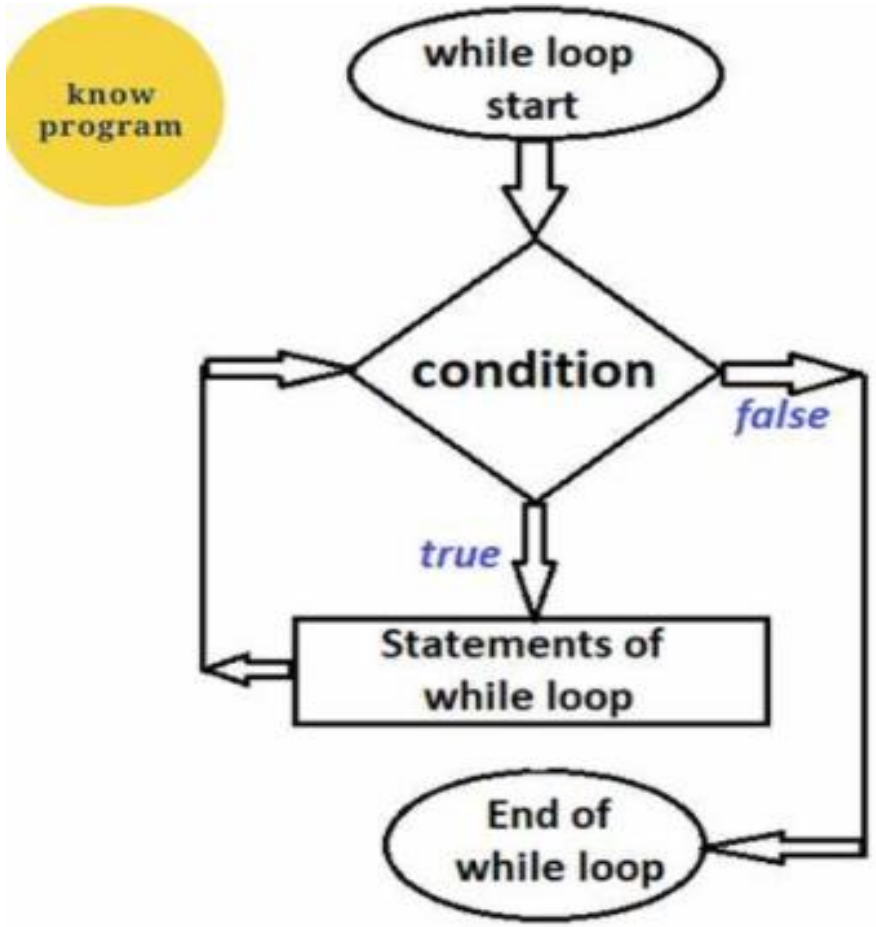
{

Block of one or more statements;

}

- The test expression usually contains relational, and possibly logical operators. The while loop tests the expression at the top of the loop.

While Loop flow chart



Example: The following programs prints numbers 1-10

```
#include <stdio.h>
int main()
{
int number;
number =1;
printf("Numbers from 1 to 10: \n");
while(number<=10)
{
printf("%d ",number);
number++;
}
return 0;
}
```

do……while

- its similar to the while loop except the relational test occurs at the bottom(rather than top)of the loop.this ensures that the body of the loop executes at least once.
- **Format:**

do

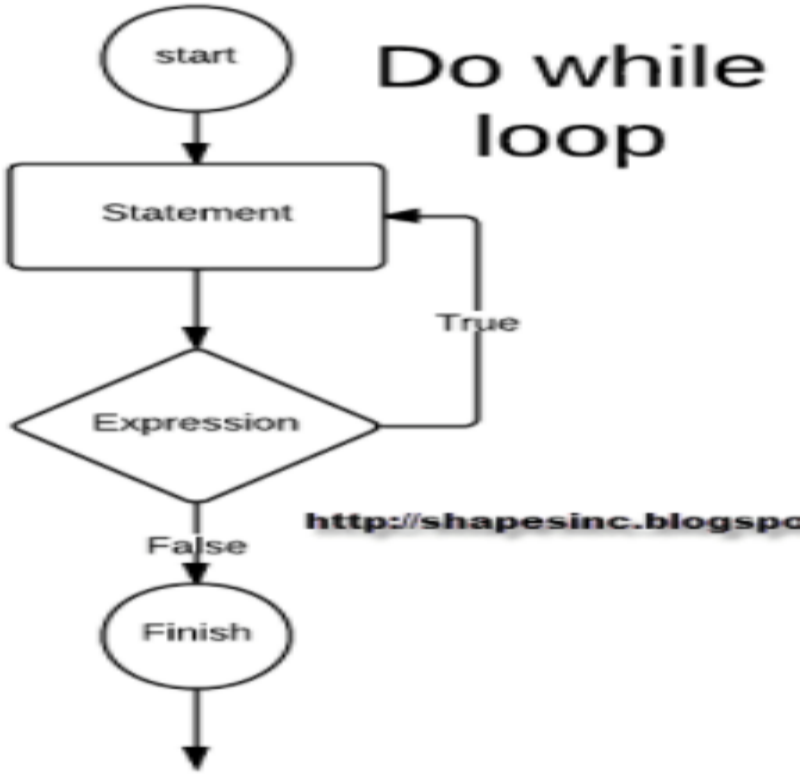
{

Block of one or more c statements;

}

while(test expression)

Do…While flowchart



Example

```
#include<stdio.h>
int main(){
        int num = 1;
        //Start do while loop
         do{
                printf("%d\n", num);

                                ++num;
         }while (num <= 10);

        return 0;
}
```

```
            printf("%d\n", num);

                ++num;
        }while (num <= target);

        return 0;
}
```

The END