CONTROL STATEMENTS

SUBMITTED BY:SHABNAM

Control statements

- - **Control Statements**, if, elseif, while, do, for loop - Free tutorial and references for ANSI C Programming. You will learn ISO GNU K and R C99 C Programming computer language in easy steps. C is the most popular system programming and widely used computer language in the computer world.

CONDITIONAL EXECUTION AND SELECTION

- Selection Statements
- The Conditional Operator
- The switch Statement

SELECTION STATEMENTS

One-way decisions using if statement

Two-way decisions using if-else statement

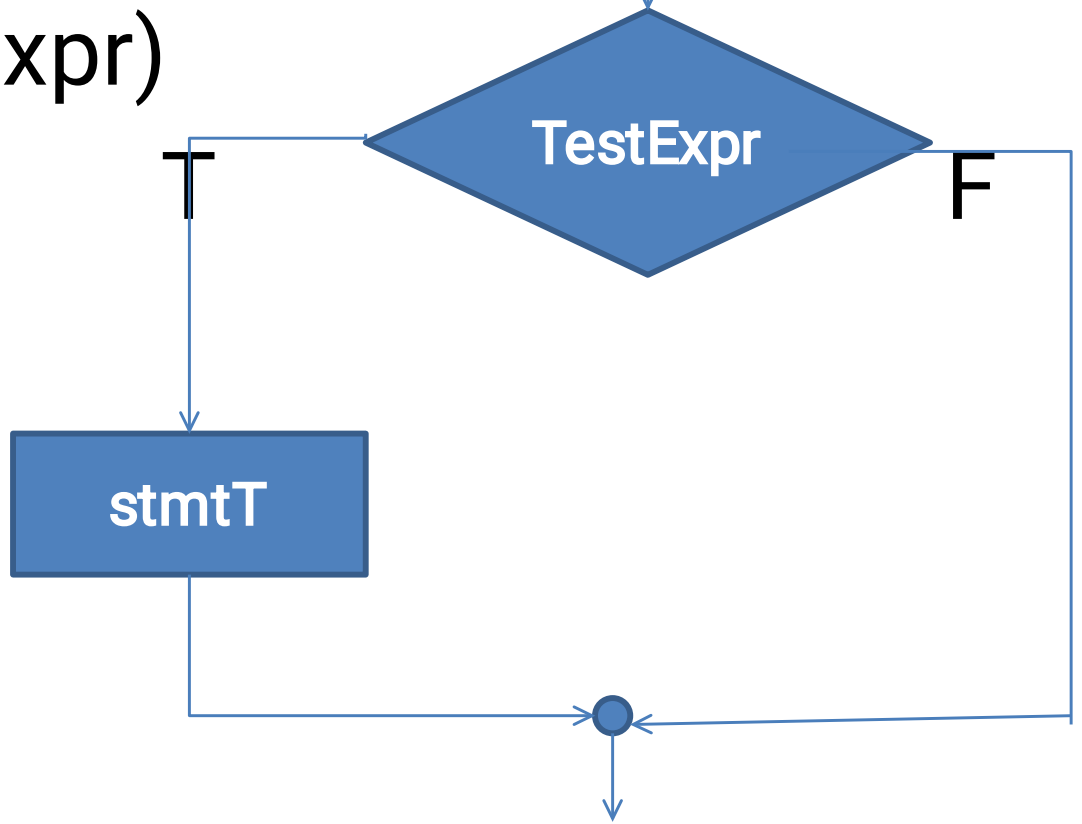
Multi-way decisions

Dangling else Problem

ONE-WAY DECISIONS USING IF STATEMENT

Flowchart for if construct

```
if(TestExpr)
stmtT;
```



WRITE A PROGRAM THAT PRINTS THE LARGEST AMONG THREE NUMBERS.

Algorithm	C Program

1. START	#include <stdio.h>
	int main()
	{
	int a, b, c, max;
	printf("\nEnter 3 numbers");
	scanf("%d %d %d", &a, &b, &c);
	max=a;
	if(b>max)
	max=b;
	if(c>max)
	max=c;
	printf("Largest No is %d", max);
	return 0;
	}

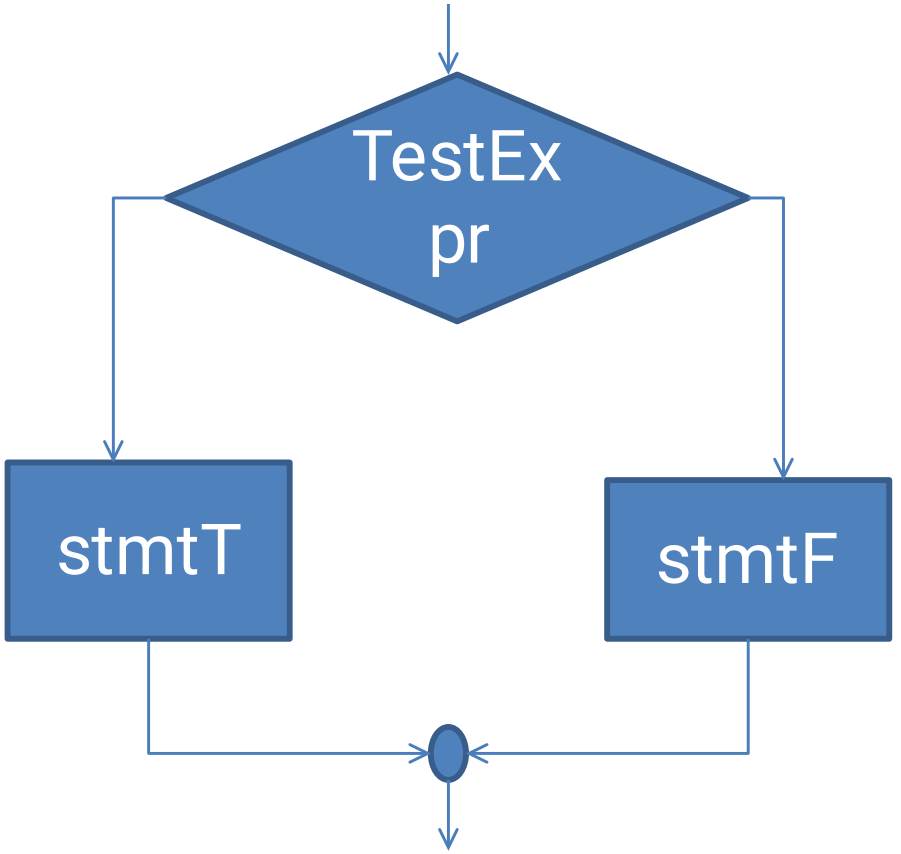
2. PRINT "ENTER THREE NUMBERS"	
3. INPUT A, B, C	
4. MAX=A	
5. IF B>MAX THEN MAX=B	
6. IF C>MAX THEN MAX=C	
7. PRINT "LARGEST NUMBER IS", MAX	
8. STOP	

TWO-WAY DECISIONS USING IF-ELSE STATEMENT

The form of a two-way decision is as follows:

```
if(TestExpr)
   stmtT;
else
   stmtF;
```

Flowchart of if-else construct



WRITE A PROGRAM THAT PRINTS THE LARGEST AMONG THREE NUMBERS.

Algorithm	C Program

1. START	#include <stdio.h>
	int main()
	{
	int a, b, c, max;
	printf("\nEnter 3 numbers");
	scanf("%d %d %d", &a, &b, &c);
	max=a;
	if(b>max)
	max=b;
	if(c>max)
	max=c;
	printf("Largest No is %d", max);
	return 0;
	}

2. PRINT "ENTER THREE NUMBERS"	
3. INPUT A, B, C	
4. MAX=A	
5. IF B>MAX THEN MAX=B	
6. IF C>MAX THEN MAX=C	
7. PRINT "LARGEST NUMBER IS", MAX	
8. STOP	

MULTI-WAY DECISIONS

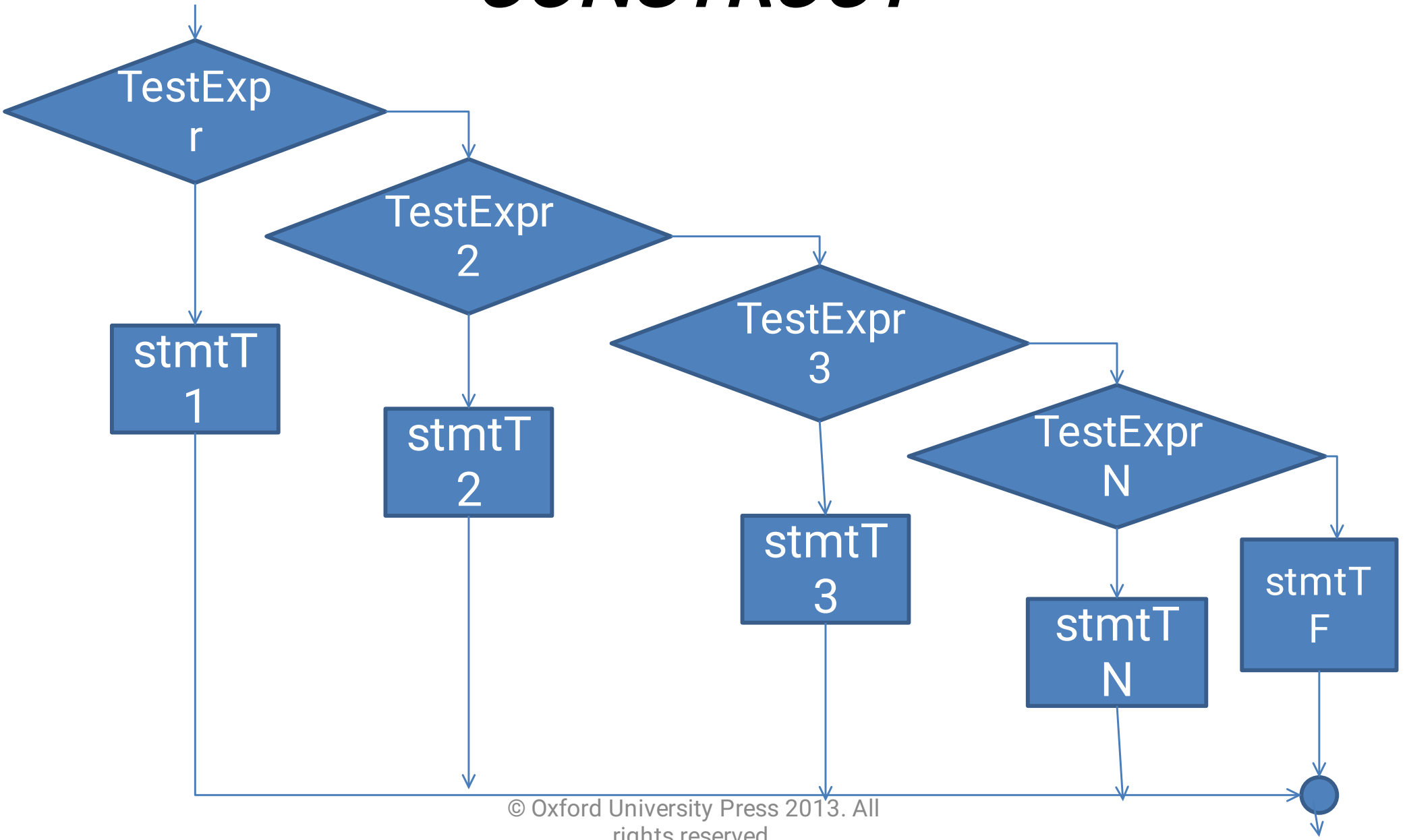
```
if(TestExpr1)
  stmtT1;
    else if(TestExpr2)
      stmtT2;
        else if(TestExpr3)
          stmtT3;
            .. .
              else if(TestExprN)
                stmtTN;
                  else
                    stmtF;
```

if-else-if ladder

```
switch(expr)
{
  case constant1: stmtList1;
     break;
  case constant2: stmtList2;
     break;
  case constant3: stmtList3;
     break;
      ……….
      ……….
  default: stmtListn;
}
```

General format of switch statements

FLOWCHART OF AN IF-ELSE-IF CONSTRUCT



NESTED IF

- When any if statement is written under another if statement, this cluster is called a nested if.
- The syntax for the nested is given here:

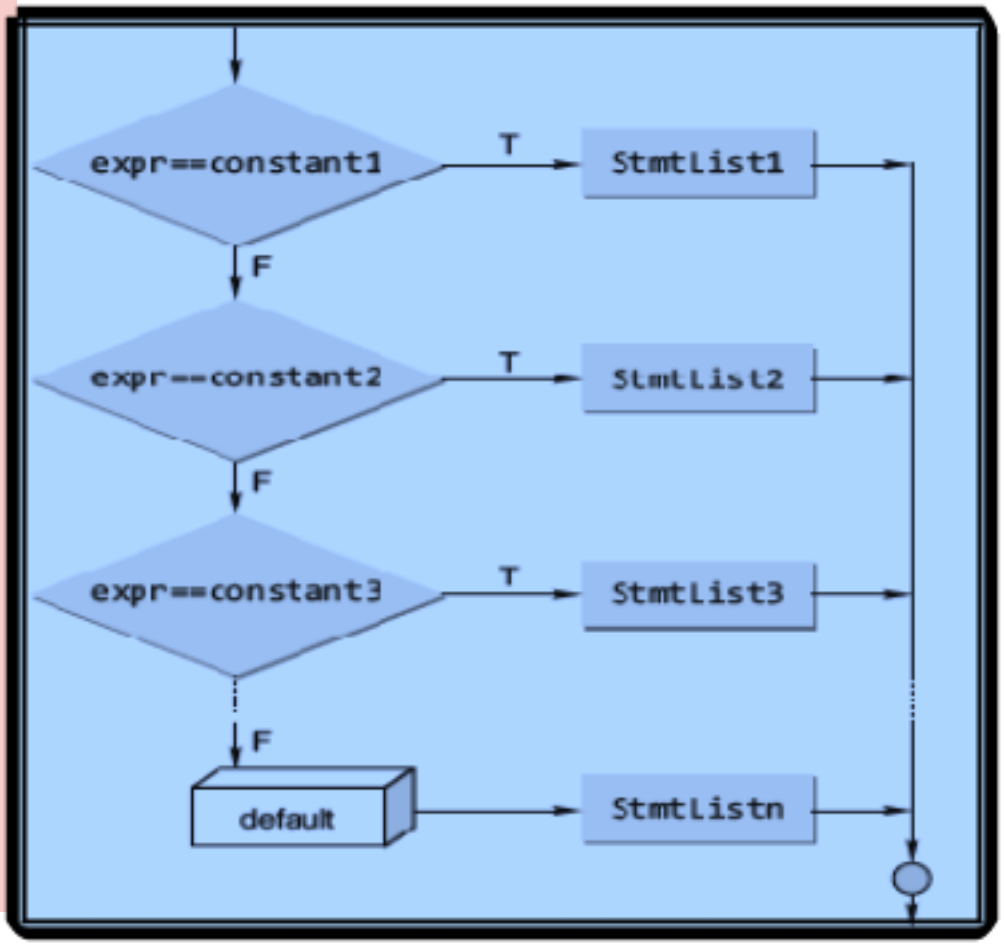
Construct 1	Construct 2

if(TestExprA)
 	
if(TestExprB)
stmtBT;
else
stmtBF;
else
stmtAF;
	if(TestExprA)
	
	if(TestExprB)
	stmtBT;
	else
	stmtBF;
	else
	
	if(TestExprC)
	stmtCT;
	else
	stmtCF;

THE SWITCH STATEMENT

The general format of a switch statement is

```
switch(expr)
{
case constant1: stmtList1;
break;
case constant2: stmtList2;
break;
case constant3: stmtList3;
break;
……
……
default: stmtListn;
}
```



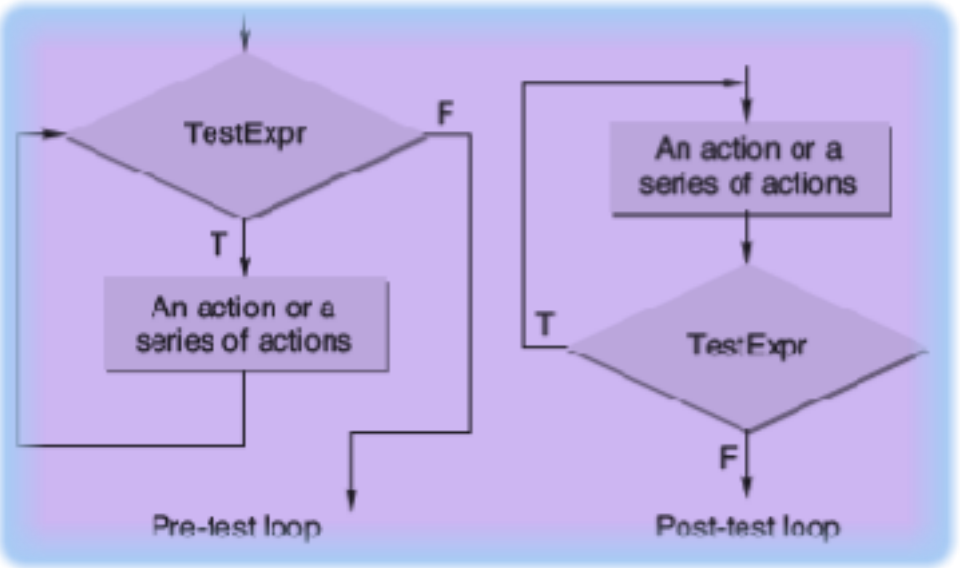
The C switch construct

SWITCH VS NESTED IF

- The switch differs from the else-if in that switch can test only for equality, whereas the if conditional expression can be of a test expression involving any type of relational operators and/or logical operators.
- A switch statement is usually more efficient than nested ifs.
- The switch statement can always be replaced with a series of else-if statements.

ITERATION AND REPETITIVE EXECUTION

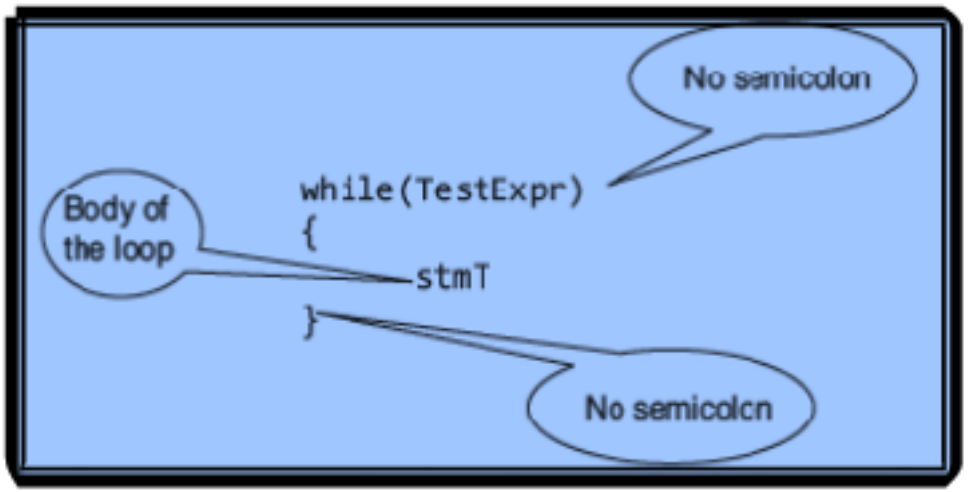
- A loop allows one to execute a statement or block of statements repeatedly. There are mainly two types of iterations or loops – *unbounded iteration or unbounded loop* and *bounded iteration or bounded loop*.
- A loop can either be a *pre-test loop or be a post-test loop* as illustrated in the diagram.

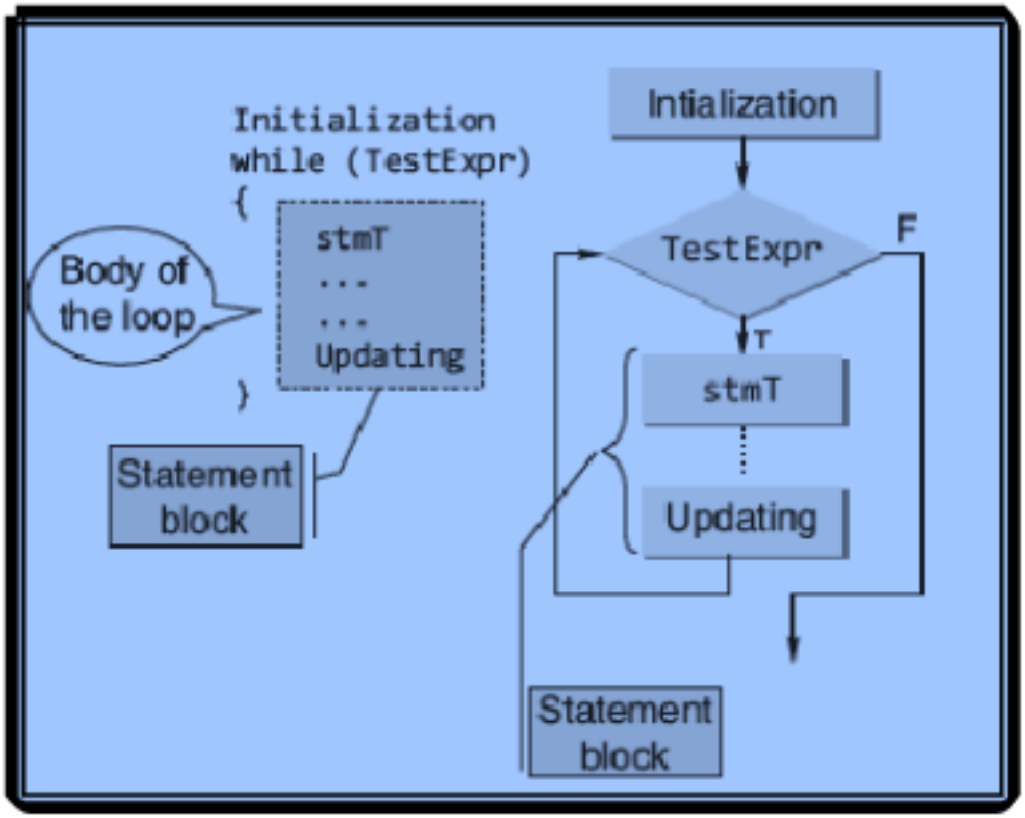


"WHILE" CONSTRUCT

Expanded Syntax of "while" and its Flowchart Representation

while statement is a pretest loop. The basic syntax of the while statement is shown below:





AN EXAMPLE

```
#include <stdio.h>
int main()
{
   int c;
   c=5; // Initialization
   while(c>0)
      { // Test Expression
         printf(“ \n %d”,c);
         c=c-1; // Updating
      }
    return 0;
}
```

This loop contains all the parts of a while loop. When executed in a program, this loop will output

5
4
3
2
1

TESTING FOR FLOATING-POINT 'EQUALITY'

```
float x;
x = 0.0;
while(x != 1.1)
{
   x = x + 0.1;
   printf("1.1 minus %f equals %.20g\n", x, 1.1 -x);
}
```

The above loop never terminates on many computers, because 0.1 cannot be accurately represented using binary numbers.

Never test floating point numbers for exact equality, especially in loops.

The correct way to make the test is to see if the two numbers are 'approximately equal'.

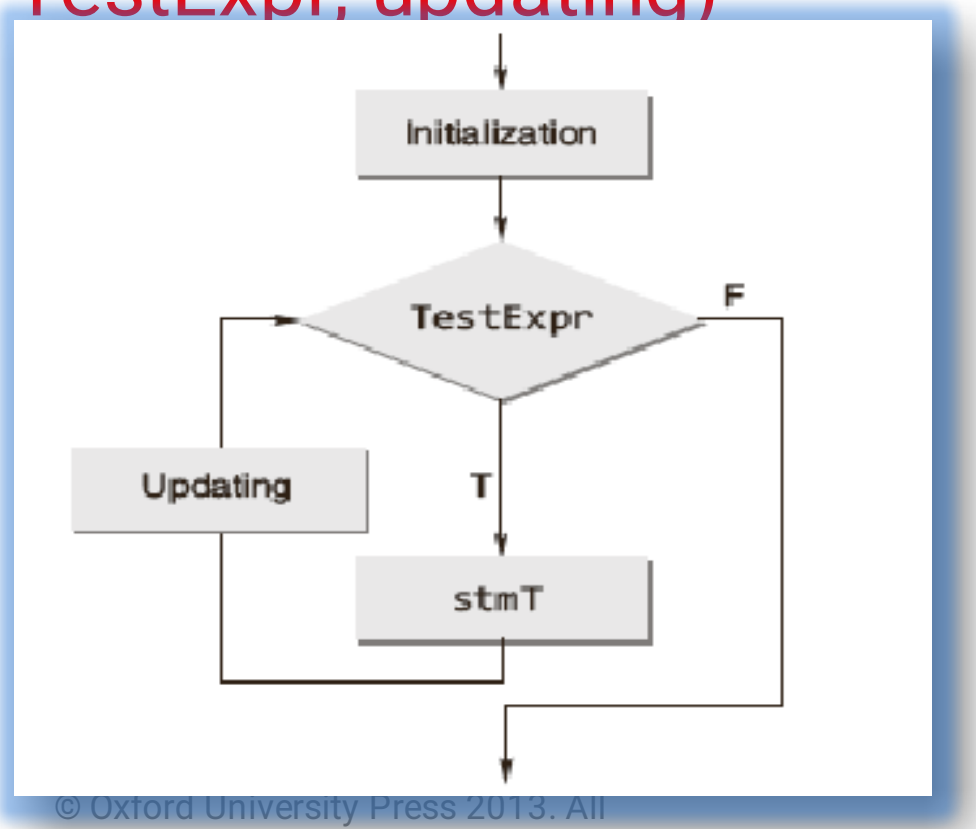
“FOR” CONSTRUCT

- The general form of the for statement is as follows:

for(initialization; TestExpr; updating)
  stmT;

for construct

flow chart



EXAMPLE

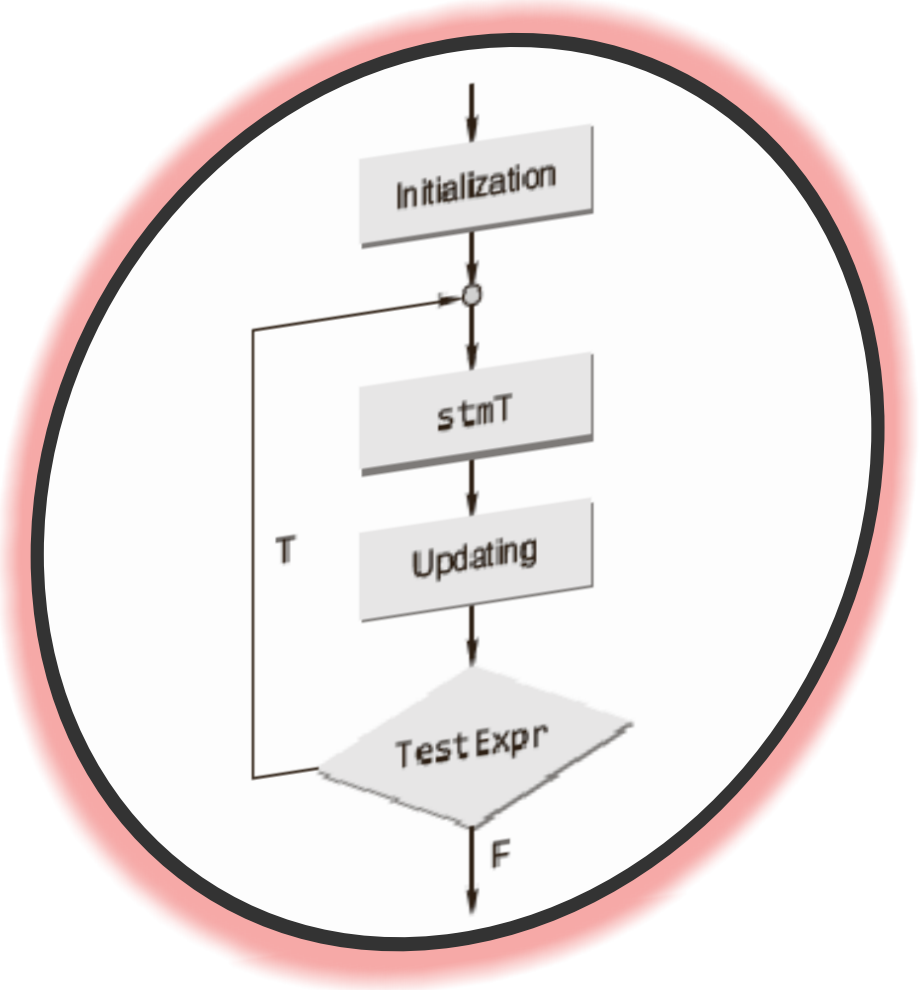
```
#include <stdio.h>
int main()
{
   int n, s=0, r;
   printf(“\n Enter the Number”);
   scanf(“%d”, &n);
   for(;n>0;n/=10)
       {
         r=n%10;
         s=s+r;
       }
    printf(“\n Sum of digits %d”, s);
    return 0;
}
```

8.3 "DO-WHILE" CONSTRUCT

The C do-while loop

The form of this loop construct is as follows:

```
do
{
  stmT; /* body of
   statements would be
   placed here*/
}while(TestExpr);
```



With a do-while statement, the body of the loop is executed first and the test expression is checked after the loop body is executed. Thus, the do-while statement always executes the loop body at least once.

```
#include <stdio.h>
int main()
{
   int x = 1;
   int count = 0;
   do {
         scanf("%d", &x);
         if(x >= 0) count += 1;
       } while(x >= 0);
   return 0;
}
```

Some methods of controlling repetition in a program are:

- Using Sentinel Values
- Using Prime Read
- Using Counter

GOTO STATEMENT

The control is unconditionally transferred to the statement associated with the label specified in the goto statement. The form of a goto statement is

goto label_name;

The following program is used to find the factorial of a number.

```
#include <stdio.h>
int main()
{
   int n, c;
   long int f=1;
   printf("\n Enter the number:");
   scanf("%d",&n);
   if(n<0)
      goto end;
   for(c=1; c<=n; c++)
       f*=c;
   printf("\n FACTORIAL IS %ld", f);
   end:
    return 0;
 }
```

- “return” statements
- “break” statements
- “continue” statements

break	continue
1. It helps to make an early exit from the block where it appears.	1. It helps in avoiding the remaining statements in a current iteration of the loop and continuing with the next Iteration
2. It can be used in all control statements including switch construct.	2. It can be used only in loop constructs.

- A nested loop refers to a loop that is contained within another loop.
- If the following output has to be obtained on the screen

1

2 2

3 3 3

4 4 4 4

then the corresponding program will be

```
#include <stdio.h>
int main()
{
  int row, col;
  for(row=1;row<=4;++row)
  {
    for(col=1;col<=row;++col)
      printf("%d \t", row);
    printf("\n");
  }
  return 0;
}
```

- *Writing expressions like a<b<c or a==b==c etc.*
- *Use of = instead of ==*
- *Forgetting to use braces for compound statement*
- *Dangling else*
- *Use of semicolon in loop*
- *Floating point equality*