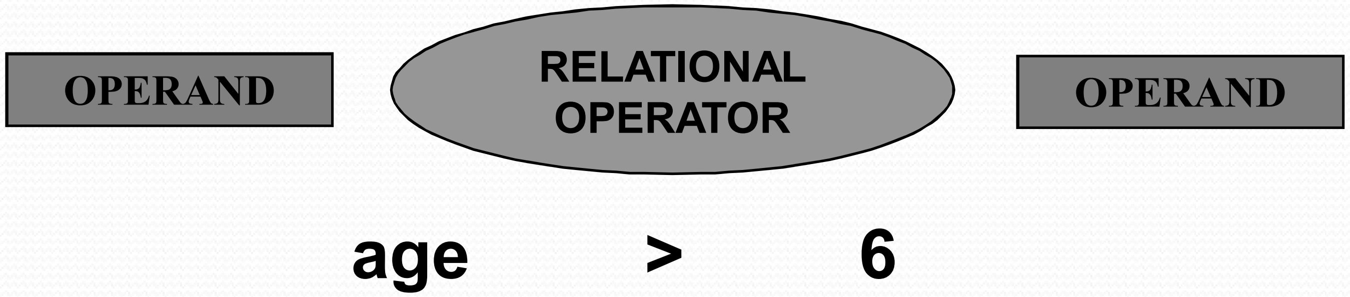
4. SELECTION

Relational Expressions

A relational expression consists of a relational operator connecting using two variables and/or constant operands.



Relational Operators

Relational Operator	Meaning	Example
<	less than	age <30
>	greater than	ht > 4.5
<=	less than or equal to	age <=3
>=	greater than or equal to	ht>=4
==	equal to	g ==2
!=	not equal to	s != 3

Relational expressions are evaluated to yield only an integer value of '1' or '0'. A condition true evaluates to '1' false evaluates '0'

Precedence Level :

Operator	Symbols	Precedence
Comparison	<,<=,>,>=	Level 10
Equality	== !=	Level 9

Associativity : Left to right

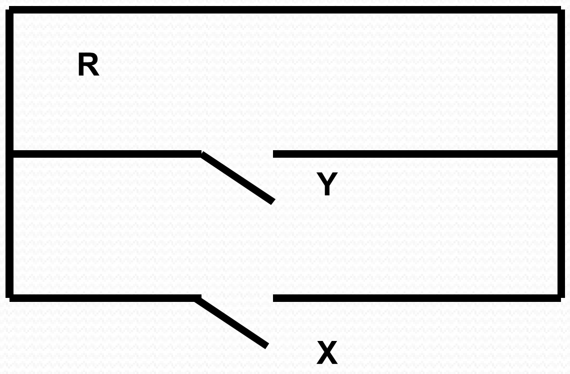
Example:

```
#include<stdio.h>
int main(void)
{
printf("The value of 2<3 is %d\n",2<3);
printf("The value of 3<2 is %d\n"3<2);
return 0;
}
```

LOGICAL OPERATORS:

Logical Operator	Meaning	**Example**
&&	and	(age <31) && (age>41)
| |	or	(age > 21) | | (sex ==1)
!	not	!(age <=3)

Logical Operators Truth values:
i) AND Function (&&)
[Precedence Level : 5]

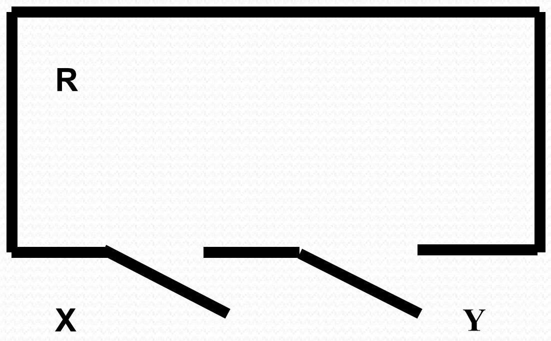


Truth Table for AND Function		
X	Y	R = X&&Y
0	0	0
0	1	0
1	0	0
1	1	1

Representation :R = X && Y

ii) OR Function (| |)

[Precedence Level : 4]



Truth Table for OR Function		
X	Y	R = X|| y
0	0	0
0	1	1
1	0	1
1	1	1

Representation : R = X || Y

iii) NOT Function: (!)

[Precedence Level : 15]

Truth Value of NOT Function	
X	R = ! X
0	1
1	0

Representation : R = ! X

```
int i = 7;
f loat f = 5.5;
```

Expression	Interpretation	Value
f > 5	true	1
!(f > 5)	false	0
i <= 3	false	0
!(i <= 3)	true	1
i > f +1	true	1
!(i > f+1)	false	0

Example:

```
#include<stdio.h>
int main(void)
{
printf"The value of ((20>30) &&(10>30)) is %d\n", ((20>30)
&&(10>30)));
}
```

Result:

```
The value of ((20>30) &&(10>30))  is 0
```

Selection

Selection is the second construct of a structured programming language.

We perform an action depending upon the prevailing conditions.

Selection allows you to make some decisions and choose between two or more alternatives. We make some decision and select a particular choice.

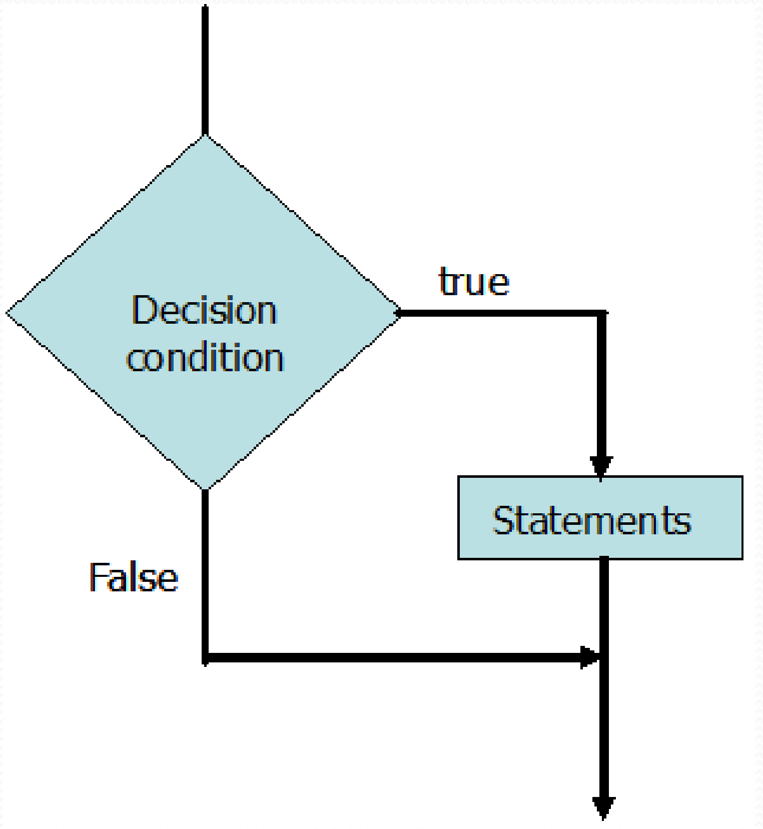
Your program reflects the real world problem and it should have the capability of making a decision and choose between two or more alternatives

The 'if' statement

The if statement is sometimes called a conditional statement. The operation of a if statement is governed by a conditional test. If the conditional test is true, one or more actions are executed.

The syntax of a simplest if statement is shown below:

```
if(expression)
        statement;
```



NOTE: In C, an expression is true; it is evaluated to a nonzero value. If the expression is zero, it is false.

NOTE:

1. The expression must be enclosed in parentheses.
2. The action statement may be either simple statement or a compound statement.
3. The expression can have a side effect.

The expression is evaluated first. If the expression is true (a non zero value), the action statement is executed and proceeds to the statement immediately following the action statement.

If the expression is false, the action statement will not be executed.

The action statement will be skipped and the statement following the action statement will be executed.

If the action statement is a compound statement, it must be enclosed in braces.

Example:

```
int mcode = 1;
if (mcode == 1)
        printf("The sun is very hot\n");
printf("Please switch on the AC\n");
```

Result:

```
  The sun is very hot
  Please switch on the AC
```

```
float interest;
float balance = 1000.00:
if (balance>900)    {
  interest = 0.2 * balance;
   printf("interest = %f\n", interest);    }
```

Result:

```
  interest = 5.0000
```

```
/* Program to demonstrate a if statement */

#include<stdio.h>

int main(void)
{
int number;

printf(“Enter a number “);
scanf(“%d”,&number);

if (number%2 == 0)
   printf(“The number %d is an even number\n”, number);

return 0;
}
```

Mrs.ABC Marketing Company announces a bonus of 30% to their employees whose basic salary is less than $3000 and more than 2 years of service. Write a program in C to read employees basic salary, number of years of service and compute their bonus.

```
#include<stdio.h>
 int main(void)
{
float pay,bonus;
int year;
bonus = 0.0;
printf("Enter Basic Pay ");
scanf("%f",&pay);
printf("Enter Number of years of service ");
scanf("%d",&year);
if(pay < 3000 && year > 2)
  bonus = pay*0.3;
printf("Bonus = %8.2f\n", bonus);
return 0;
}
```

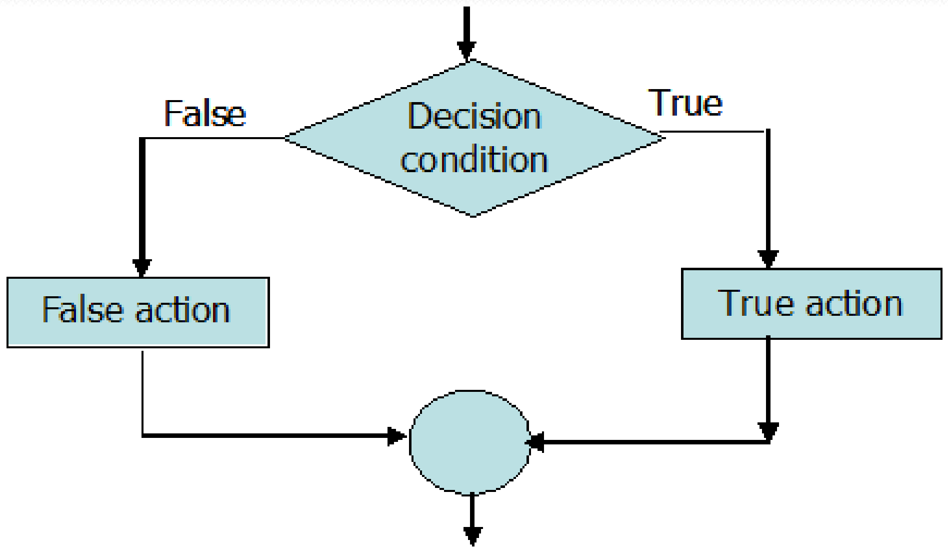
If-Else Statement

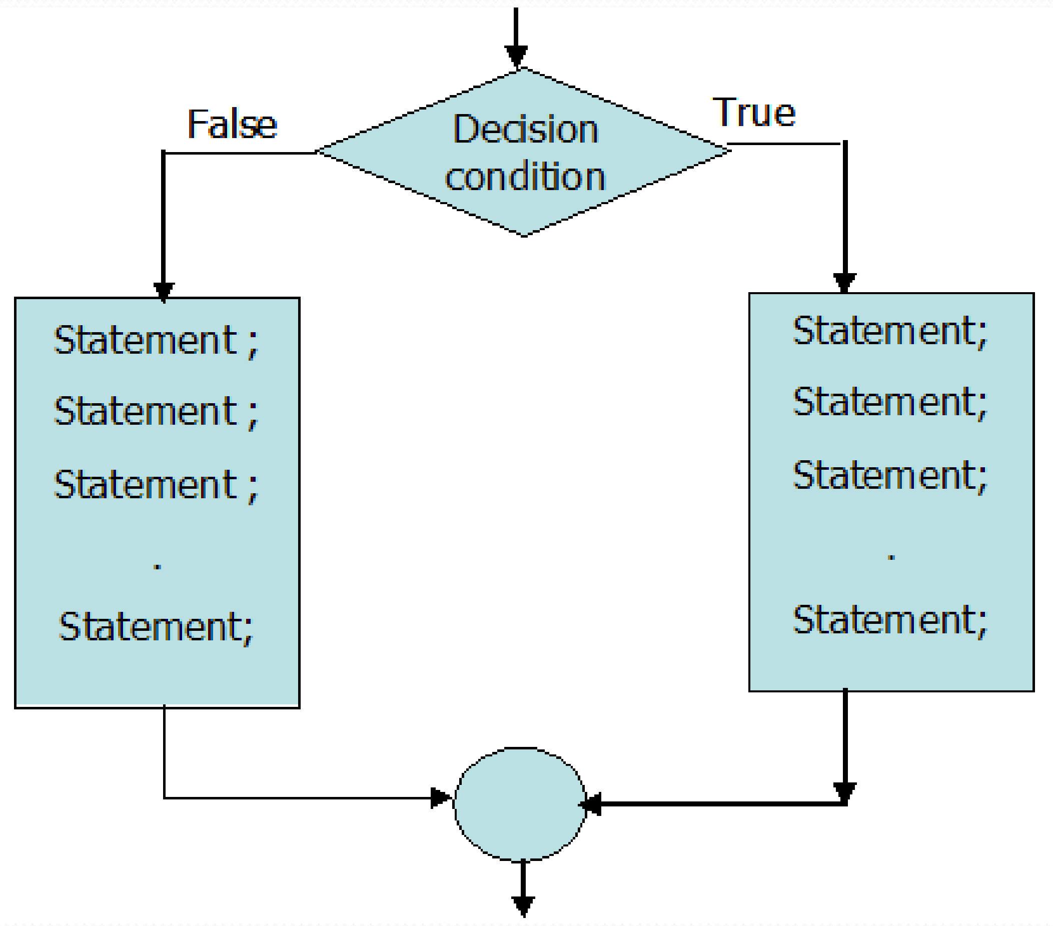
An if-else statement is a compound statement used to make a decision between two alternatives.

Recall, while using a if statement, the statement following the if expression is evaluated if the expression is evaluated to true. If the expression is false, the statement is simply skipped and the control is passed to the immediate next statement.

In the case of a if-else statement, **if the condition is true one or more action statements are performed**.

If the condition is false, then a different action or set of statements are executed.




Decision condition
False
True
Statement ;
Statement ;
Statement ;
.
Statement;
Statement;
Statement;
Statement;
.
Statement;

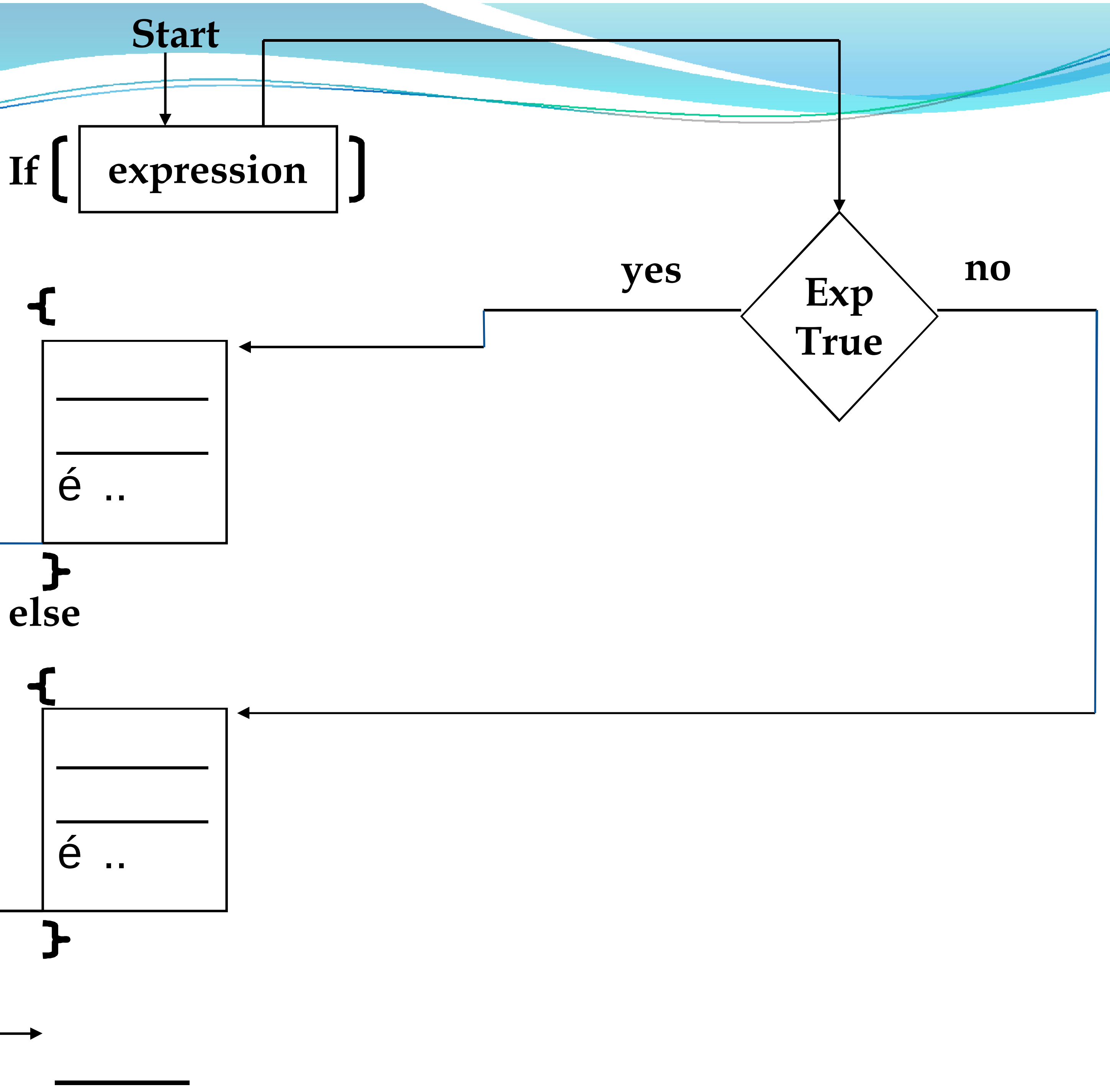
The If-Else Statement

```
    if (expression)
            action statement – 1
    else
            action statement – 2
```

- In this type of if statement, the expression is evaluated first.
- If the expression has a non zero value then action statement 1 will be executed.
- Otherwise action statement 2 will be executed.

NOTE

1. Either action statement 1 or 2 may consist of a single or multiple statements
2. A single statement is terminated with a semicolon.
3. A multiple action statement is enclosed in braces.


Start
If
expression
Exp True
yes
no
{
é ..
}
else
{
é ..
}

Example:

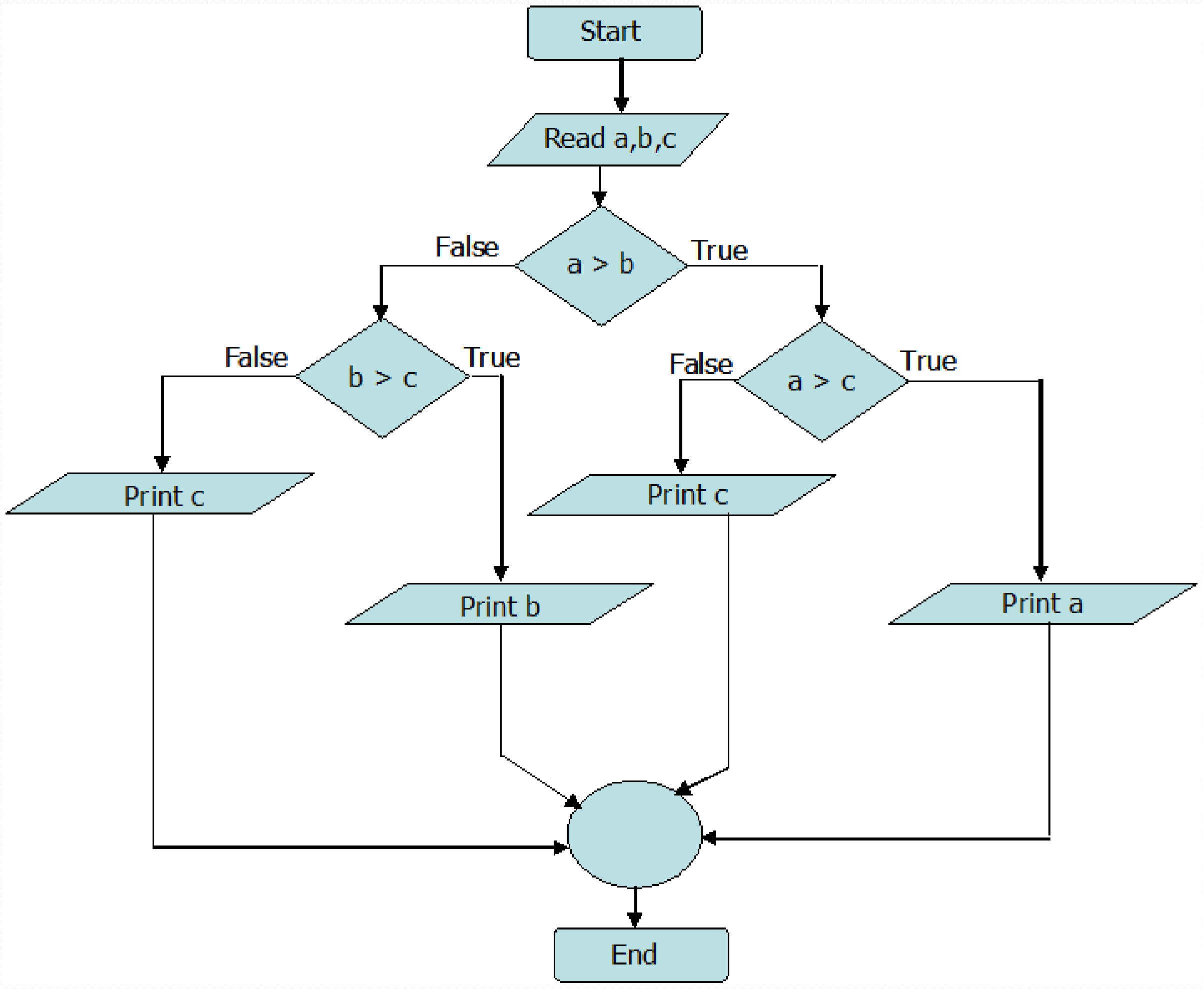
```
int number = 5;
if (number %2 == 0)
    printf("the number %d is even\n", number);
else
    printf("the number %d is odd\n", number);
```

The action statement can be another if – else statement.

Example:

```
          if (rain == 0)
             if (fish == 0)
                 printf("It is not raining you do not have a fish\n");
            else
                printf("It is not raining you got a fish\n");
```

Flowchart to find the greatest integer given three integer



```
/*Program to find the greatest integer given three integer */
#include <stdio.h>
void main(void)
{
int a, b, c,
scanf("%d%d%d", &a &b &c);
if (a>b)
        {   if (a>c)
                        printf("%d is greater , "a);
                      else
                        printf("%d is gretaer", c);
         }
else
        {   if (b>c)
                        printf("%d is greater  ",b);
                    else
                        printf("%d is greater  ",c);
         }
return;
}
```

Example:

What is the output:

```
int x, y;
x = 3;
y = 5;
if (x < 2)
        printf(“%d\n”, x);
else
        printf(“%d\n “,y);
```

Result: 5

```
/* Program to determine the entered number is odd or even */
#include<stdio.h>
int main(void)
{
int number;
/* Get the number from the user */
printf("Enter a Number ");
scanf("%d",&number);
/* Check the number is odd or even */
if(number % 2 == 1)
  printf("%d is an odd number\n",number);
else
  printf("%d is an even number\n", number);
return 0;
}
```

```
Enter a number 10
10 is an even number
```

What is the syntax error?

a)
```
if (x < 2) then
    printf (%d\n”, x);
```

b)
```
if x <2
    printf (%d\n”, x);
```

```
if (x < 2) then
    printf (%d\n”, x);
```

Double quote is missing

Parenthesis () missing

```
if x <2
    printf (%d\n”, x);
```

Double quote is missing

What is the output?

```
code = 1
if(code == 1) {
printf(“Mathematician\n”);
if (code = 2)
        printf(“Engineer\n”);
        else
        printf(“Scientist\n”);
        }
else
        printf(“doctor\n”);
```

NESTED IF – ELSE STATEMENT

The general syntax of a nested – else statement is

```
if(expression-1)
        action statement-1;
else if (expression-2)
        action statement-2;
else if (expression-3)
        action statement-3;
else if (expression-i)
        action statement-i;
else if (expression-n)
        action statement-n;
```

Find the first expression that is true (if-any). If expression i is the first true condition, (if expression 1, 2, 3… are false), the action statement- i is executed and all the other action statement are skipped. If no action is true, no statement will be executed.

ELSE-IF-ELSE STATEMENT

```
if(expression-1)
        action statement-1;
else if (expression-2)
        action statement-2;
else if (expression-3)
        action statement-3;
else if (expression-i)
        action statement-i;
else if (expression-n)
        action statement-n;
else
        action n + 1;
```

Find the first expression that is true, if any. If expression i is the first condition, then action statement i will be executed and all the other action statements are skipped.

If no expression is true, then the action statement n+1 will be executed and all other statements are skipped.

Example:

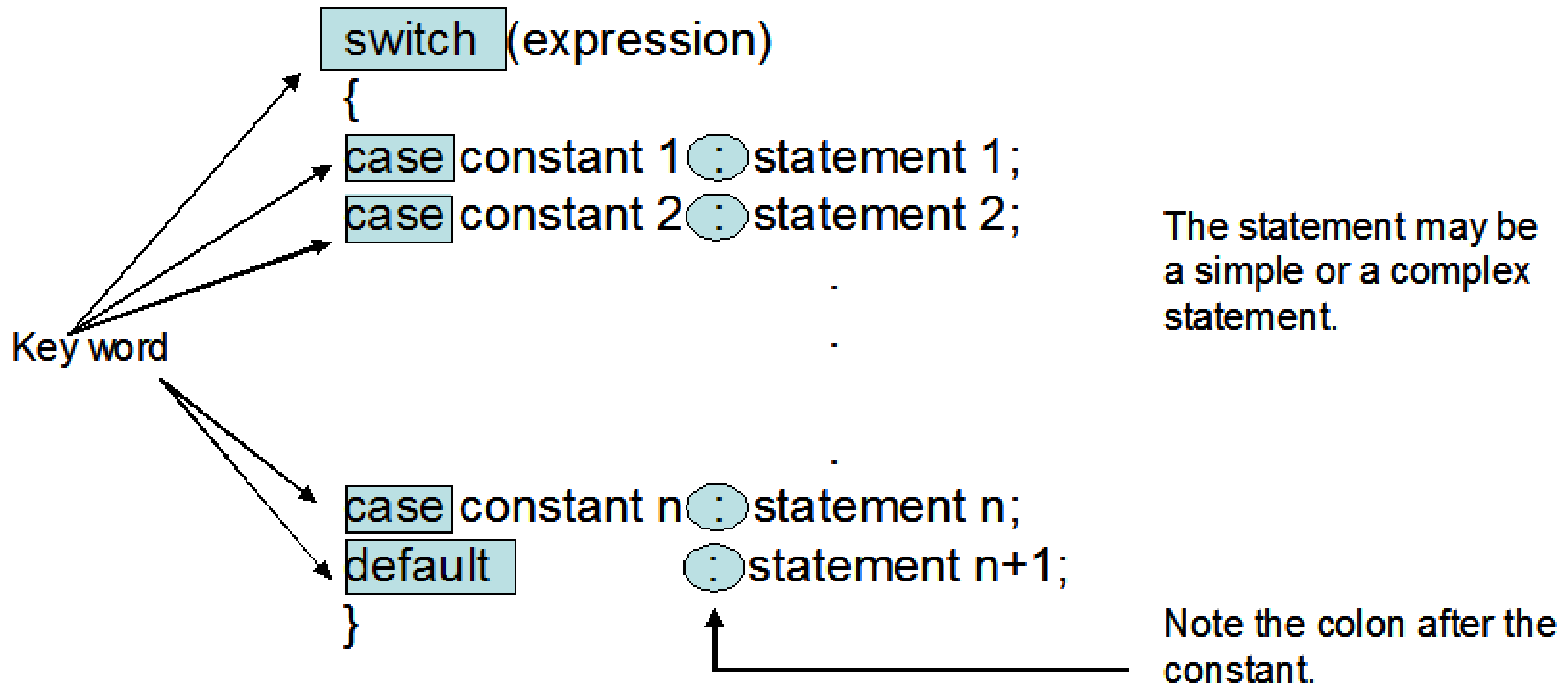
```
if ((time >= 0.0) &&(time < 12.0))
          printf("Good Morning\n");
else if ((time >=12.0) &&(time < 18.0))
          printf("Good Afternoon\n");
else if ((time >= 18.0) &&(time < 24.0))
          printf("Good Evening\n");
else
          printf("Time Out of Range\n");
```

Switch Statement

Switch statement can regarded as a special instance of if else, if-else-if, else-if-else statement.

The condition for branching in switch statement is by integer values. The general form of switch statement is

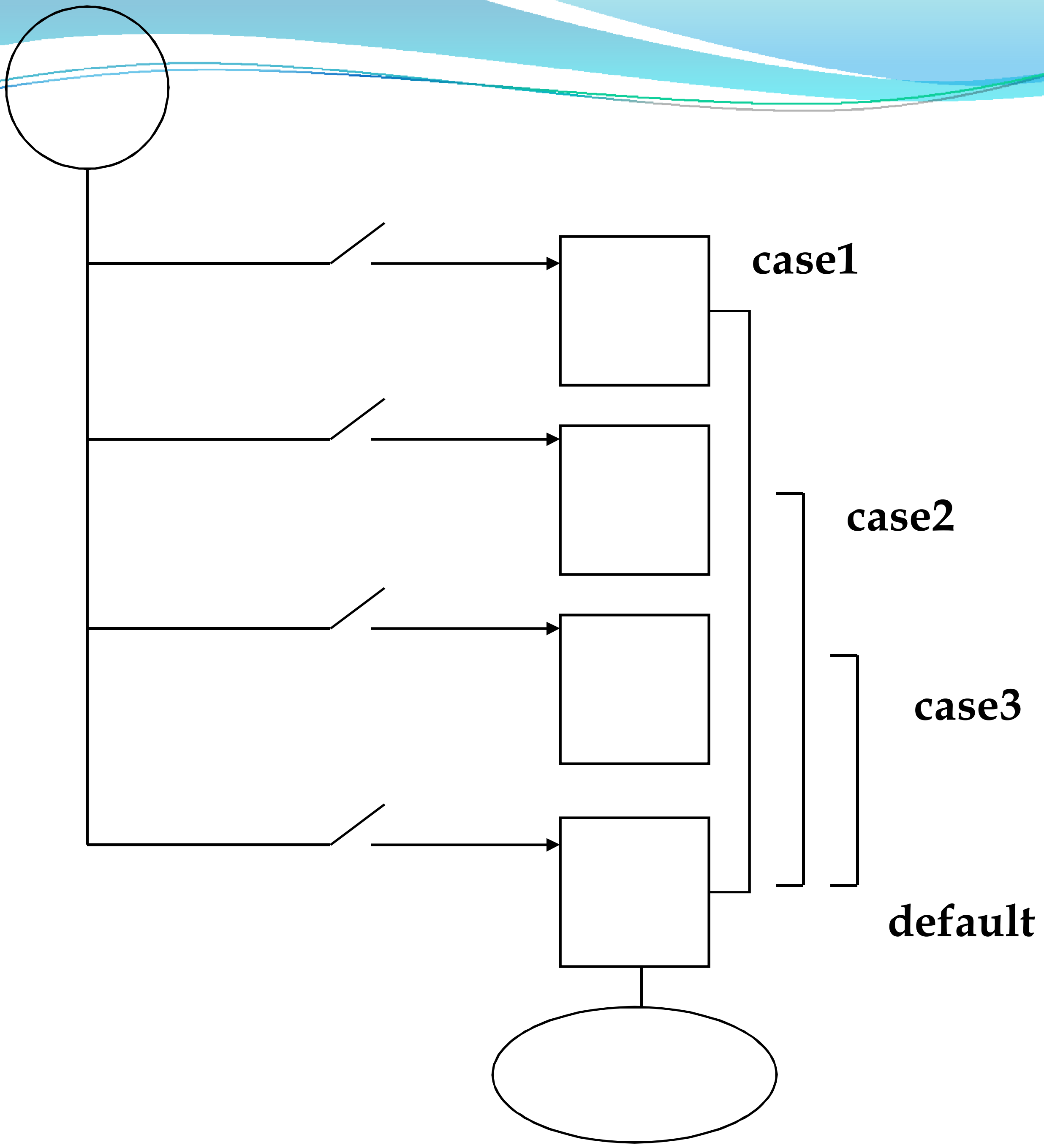
```
switch (expression giving integer value)
{
 case constant 1:             statement 1;
 case constant 2:             statement 2;
 case constant n:             statement n;
 default:                    statement n+1 ;
}
```



```
switch (expression)
{
case constant 1 : statement 1;
case constant 2 : statement 2;
                          .
                          .
                          .
case constant n : statement n;
default         : statement n+1;
}
```

The statement may be a simple or a complex statement.

Note the colon after the constant.


case1
case2
case3
default

First the expression is evaluated . The value of the integer expression is compared with the constant 1 then 2, 3, and so on. [All the constant must be different].

If the integer expression does not equal to any of the constant, execution will execute the statements in the default class.

If the value of integer expression equals constant I, execution will begin with statement i. [ie once the entry point has been located by the switch statement, all following are executed.]

Example:

```
int code = 2;
switch (code)  {
        case 1:
                printf(“Bahasa Malaysia\n”);
        case 2:
                printf(“Mandarin\n”);
         case 3:
                printf(English\n”);
        case 4:
               printf(“Tamil\n”);
      default: printf(“Not a language\n”);
}
```

Result:

```
Mandarin
English
Tamil
Not a language
```

BREAK STATEMENT

A break statement causes an immediate exit from the innermost while, for, do while and switch statement.

If when a break statement is encountered at the end of a case, it causes immediate exit from the while, for, do, do while and switch statement.

Example:

```
int movie=1;
switch (movie)
{
case 1:     printf("Die Another Day\n");
            break;
case 2:     printf("Lord of The Rings\n");
            break;
case 3:     printf("The Ring\n");
            break;
default:    printf("No Movie\n");
}
```

Result: Die Another Day

THANK YOU