ITE 12 – Fundamentals of Programming

Conditions, Logical Expressions and Selection Control Structures

Regien D. Bacabis-Nakazato

Overview

- Control Structures
- Conditions and Logical Expressions
- The if statement
- Nested If Statement
- Decision steps in algorithms
- The If Else statement
- Nested If Else statement

Control Structures

- Sequential
- Selection
- Repetition

Control Structures

- Sequential
- Selection
- Repetition

Control Structures

- Sequential
- Selection
- Repetition

Conditions and Logical Expressions

- True is equal to the integer 1
- False is equal to the integer 0
- Thus, each conditional expression returns the **integer 1** if true and the **integer 0** if false

this expression	is true if
x == y	x is equal to y
x != y	x is not equal to y
x < y	x is less than y
x > y	x is greater than y
x <= y	x is less than or equal to y
x >= y	x is greater than or equal to y

- Conditional operations evaluated left to right
  - Example: x<y<z = (x<y)<z

Conditions and Logical Expressions

Note:

Consider this code

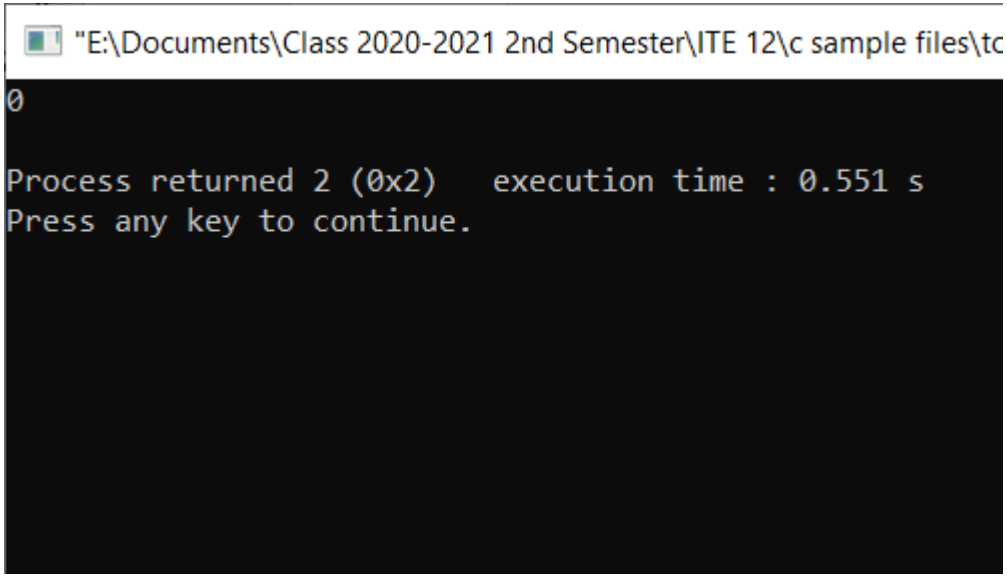
```
1  void main(){
2
3    int x=1, y=2, z=3;
4    printf("%d\n", z>y>z);
5  }
6
```

What do you think is the output?

Conditions and Logical Expressions

Note:

The output is



Why?

x=1, y=2, z=3

z>y>x

(z>y)>x

(3>2)>1

(1) > 1 FALSE

Conditions and Logical Expressions

The 3 logical operators in C

Operator	Meaning
`&&`	AND operator
`||`	OR operator
`!`	NOT operator

Conditions and Logical Expressions

The and operator

Operand 1	Operand 2	Result
true	true	true
true	false	false
false	true	false
false	false	false

Syntax: <Condition1> && <Condition2>&& ... <Condition*n*>

Example:

(a==12) && (b<5)

(a<12) && (b<5) &&(x==z)

x>y && y<z && z==y

Conditions and Logical Expressions

The or operator

Operand 1	Operand 2	Result
true	true	true
true	false	true
false	true	true
false	false	false

Syntax: <Condition1> || <Condition2>|| … <Condition*n*>

Example:

(a==12) || (b<5)

(a<12) || (b<5) || (x==z)

x>y || y<z || z==y

Conditions and Logical Expressions

The not operator

Operand	Result
true	false
false	true

Syntax: !<condition> , !(<condition>)

Example:

!(a==12)

!(a<12)

!x

!(z>x)

Conditions and Logical Expressions

Precedence

the NOT (!) operator has the highest precedence and it associates from right to left. The precedence of AND (&&) operator is higher than OR (||) operator and they both associates from left to right.

Example:

```
int age = 10, height = 45;
(age < 12 && height < 48) || (age > 65 && height > 72);
```

Conditions and Logical Expressions

Precedence

the NOT (!) operator has the highest precedence and it associates from right to left. The precedence of AND (&&) operator is higher than OR (||) operator and they both associates from left to right.

Example:

int age = 10, height = 45;

(age < 12 && height < 48) || (age > 65 && height > 72);

=> (10 < 12 && 45 < 48) || (10 > 65 && 45 > 72)

=> (1 && 1) || (10 > 65 && 45 > 72)

=> 1 || (10 > 65 && 45 > 72)

=> 1 || (0 && 0)

=> 1 || 0

=> 1

Conditions and Logical Expressions

Precedence

the NOT (!) operator has the highest precedence and it associates from right to left. The precedence of AND (&&) operator is higher than OR (||) operator and they both associates from left to right.

Example:

```
int year = 2000;
(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0);

(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0)
=> (2000 % 4 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
```

Conditions and Logical Expressions

Precedence

the NOT (!) operator has the highest precedence and it associates from right to left. The precedence of AND (&&) operator is higher than OR (||) operator and they both associates from left to right.

Example:

```
int year = 2000;
(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0);

(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0)
=> (2000 % 4 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
```

Conditions and Logical Expressions

Precedence

the NOT (!) operator has the highest precedence and it associates from right to left. The precedence of AND (&&) operator is higher than OR (||) operator and they both associates from left to right.

Example:

```
int year = 2000;
(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0);

(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0)
=> (2000 % 4 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 0 != 0 ) || (2000 % 400 == 0)
```

Conditions and Logical Expressions

Precedence

the NOT (!) operator has the highest precedence and it associates from right to left. The precedence of AND (&&) operator is higher than OR (||) operator and they both associates from left to right.

Example:

```
int year = 2000;
(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0);
```

```
(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0)
=> (2000 % 4 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 0 != 0 ) || (2000 % 400 == 0)
=> (1 && 0 != 0 ) || (2000 % 400 == 0)
```

Conditions and Logical Expressions

Precedence

the NOT (!) operator has the highest precedence and it associates from right to left. The precedence of AND (&&) operator is higher than OR (||) operator and they both associates from left to right.

Example:

```
int year = 2000;
(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0);

(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0)
=> (2000 % 4 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 0 != 0 ) || (2000 % 400 == 0)
=> (1 && 0 != 0 ) || (2000 % 400 == 0)
=> (1 && 0 ) || (2000 % 400 == 0)
```

Conditions and Logical Expressions

Precedence

the NOT (!) operator has the highest precedence and it associates from right to left. The precedence of AND (&&) operator is higher than OR (||) operator and they both associates from left to right.

Example:

```
int year = 2000;
(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0);

(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0)
=> (2000 % 4 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 0 != 0 ) || (2000 % 400 == 0)
=> (1 && 0 != 0 ) || (2000 % 400 == 0)
=> (1 && 0 ) || (2000 % 400 == 0)
=> 0 || (2000 % 400 == 0)
```

Conditions and Logical Expressions

Precedence

the NOT (!) operator has the highest precedence and it associates from right to left. The precedence of AND (&&) operator is higher than OR (||) operator and they both associates from left to right.

Example:

```
int year = 2000;
(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0);

(year % 4 == 0 && year % 100 != 0 ) || (year % 400 == 0)
=> (2000 % 4 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 2000 % 100 != 0 ) || (2000 % 400 == 0)
=> (0 == 0 && 0 != 0 ) || (2000 % 400 == 0)
=> (1 && 0 != 0 ) || (2000 % 400 == 0)
=> (1 && 0 ) || (2000 % 400 == 0)
=> 0 || (2000 % 400 == 0)
=> 0 || (0 == 0)
=> 0 || 1
=> 1
```

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

1. `z>y>z<x>y<z`
2. `(z>y)&& !(z==x) ||!(y<z)`
3. `(z>y&&z%y)&& !(y>(z==x)) ||!(z%x || !(y<z))`

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

1. `z>y>z<x>y<z`

330 > 22 > 330 < 100 > 22 < 330

1 >330<100>22<330

0<100>22<330

1>22<330

0<330

1

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

2. `(z>y) && !(z==x) ||!(y<z)`

(330 > 22) && !(330==100) || !(22<330)

1 && !(0) || !(1)

1 && 1 || 0

1||0

1

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

1. `(z>y&&z%y) && !(y>(z==x)) ||!(z%x || !(y<z))`

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

1. `(z>y&&z%y) && !(y>(z==x)) ||!(z%x || !(y<z))`

330>22 && 330%22

1 && 0

0

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

1. `(z>y&&z%y) && !(y>(z==x)) ||!(z%x || !(y<z))`

   0

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

1. `(z>y&&z%y) && !(y>(z==x)) ||!(z%x || !(y<z))`

0

!(22>(330==100))

!(22>0)

!(1)

0

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

1. `(z>y&&z%y) && !(y>(z==x)) ||!(z%x || !(y<z))`

   0 0

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

1. `(z>y&&z%y) && !(y>(z==x)) ||!(z%x || !(y<z))`

   0 0

!(330%100 || !(22<330))
!(30||!(1))
!(30||0)
! 1
0

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

1. `(z>y&&z%y) && !(y>(z==x)) ||!(z%x || !(y<z))`

   0 0 0

Conditions and Logical Expressions

Drill!

Evaluate the following,

```
int x=100, y=22, z=330;
```

1. `(z>y&&z%y) && !(y>(z==x)) ||!(z%x || !(y<z))`

   0 0 0

o && o || o

o

The if Statement

C uses the keyword if to implement the decision control instruction.

Syntax:

```
if ( this condition is true )
     execute this statement ;
```

or

```
if ( this condition is true ){
     execute statement1 ;
     execute statement2;
     .
     .
     execute statement n
}
```

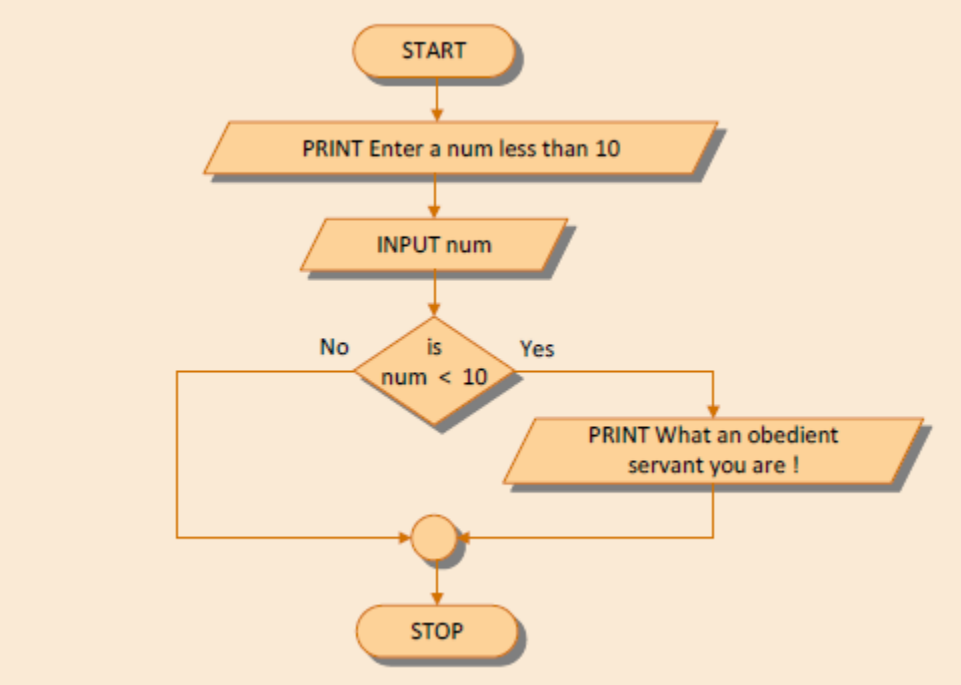
The if Statement

Example:

```
/* Demonstration of if statement */
# include <stdio.h>
int main( )
{
int num ;
printf( "Enter a number less than 10 " ) ;
scanf( "%d", &num ) ;
if ( num < 10 )
printf ( "What an obedient servant you are !\n" ) ;
return 0 ;
}
```

The if Statement

Example:



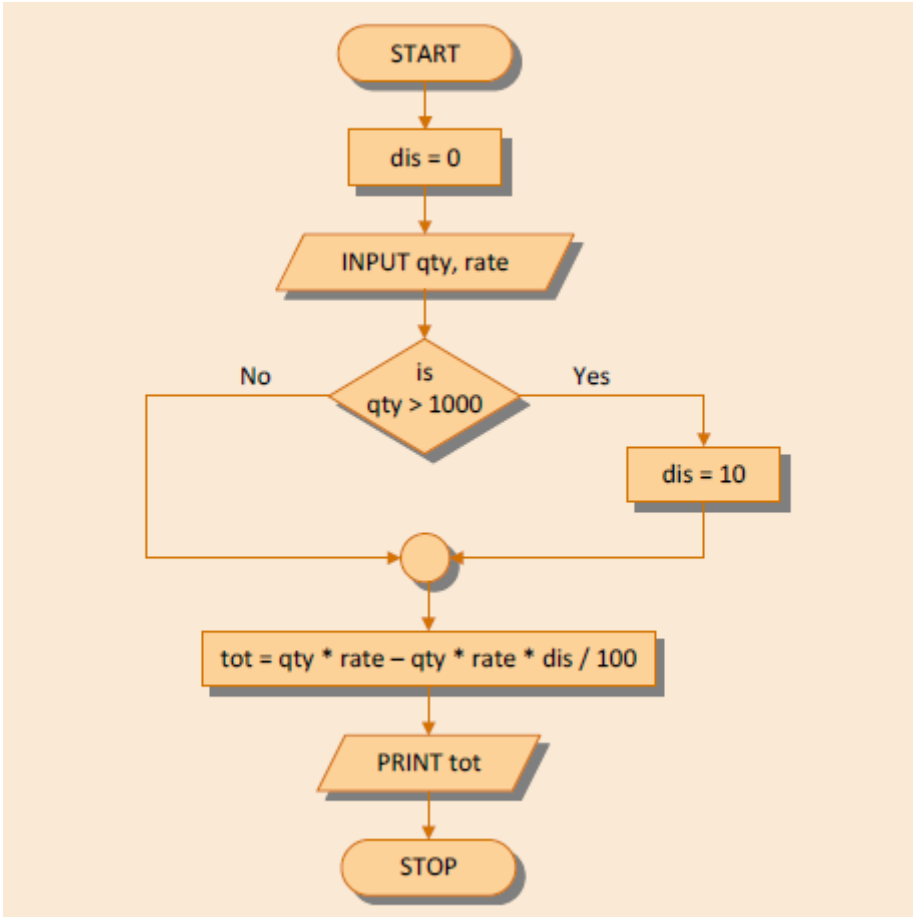
The if Statement

Example:

While purchasing certain items, a discount of 10% is offered if the quantity purchased is more than 1000. If quantity and price per item are input through the keyboard, write a program to calculate the total expenses.

The if Statement

Example:



The if Statement

Consider this statements

```
if ( 3 + 2 % 5 )
     printf ( "This works" ) ;

if ( a = 10 )
     printf ( "Even this works" ) ;

if ( -5 )
     printf ( "Surprisingly even this
works" ) ;
```

The if Statement

Note that in C a non-zero value is considered to be true, whereas a zero is considered to be false

The if Statement

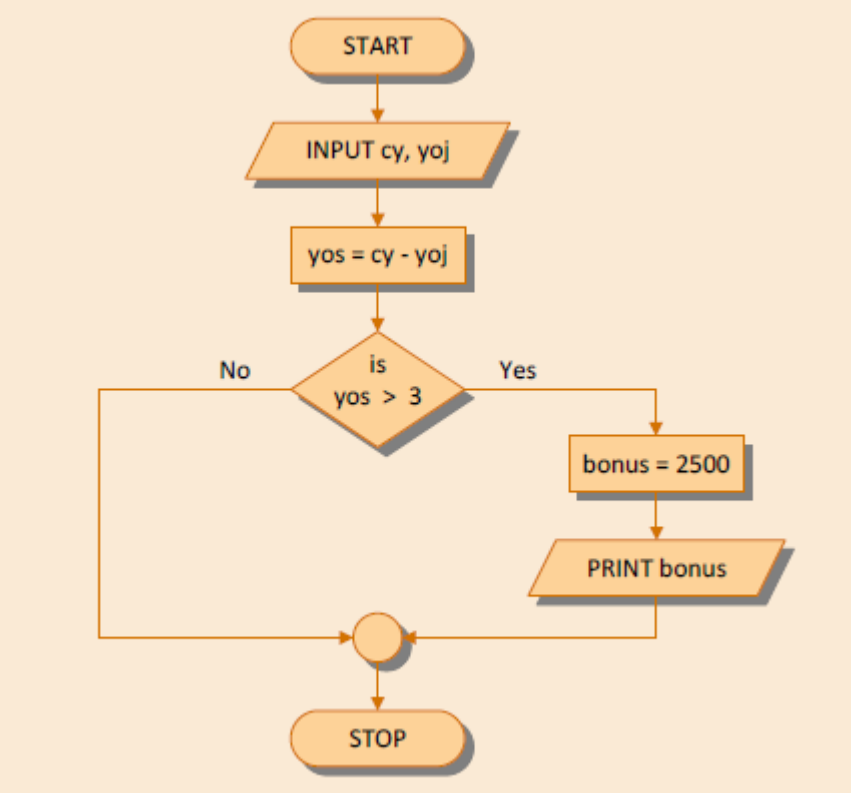
Multiple Statements within if

Example

The current year and the year in which the employee joined the organization are entered through the keyboard. If the number of years for which the employee has served the organization is greater than 3, then a bonus of Rs. 2500/- is given to the employee. If the years of service are not greater than 3, then the program should do nothing.

The if Statement

Multiple Statements within if



The if else Statement

Syntax

```
if ( this condition is true )
                execute this statement ;
else
                execute this statement
```

or

```
        if ( this condition is true ){
                execute statement1 ;
                execute statement2;
                …
                execute statement n
        }
        else{
                execute statement1 ;
                execute statement2;
                …
                execute statement n

        }
```

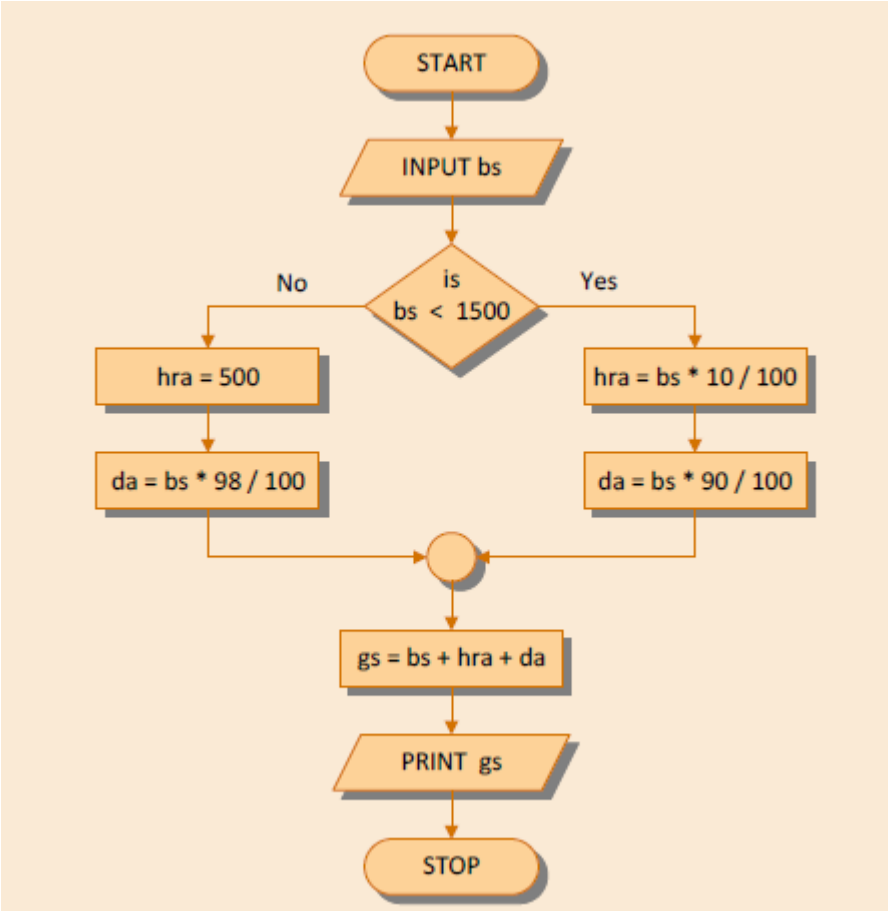
The if else Statement

Example:

If his basic salary is less than Rs. 1500, then HRA = 10% of basic salary and DA = 90% of basic salary. If his salary is either equal to or above Rs. 1500, then HRA = Rs. 500 and DA = 98% of basic salary. If the employee's salary is input through the keyboard write a program to find his gross salary.

The if else Statement

Example:



The if else Statement

Important Notes

(a) The group of statements after the if upto and not including the else is called an 'if block'. Similarly, the statements after the else form the 'else block'.

(b) Notice that the else is written exactly below the if. The statements in the if block and those in the else block have been indented to the right. This formatting convention is followed throughout the book to enable you to understand the working of the program better.

(c) Had there been only one statement to be executed in the if block and only one statement in the else block we could have dropped the pair of braces.

(d) As with the if statement, the default scope of else is also the statement immediately after the else. To override this default scope, a pair of braces, as shown in the above example, must be used.

The nested if Statement

Example:

```
/* A quick demo of nested if-else */
# include <stdio.h>
int main( )
{
          int i ;
          printf ( "Enter either 1 or 2 " ) ;
          scanf ( "%d", &i ) ;
          if ( i == 1 )
                    printf ( "You would go to heaven !\n" ) ;
          else
          {
                    if ( i == 2 )
                              printf ( "You would stay here! \n" ) ;
                    else
                              printf ( "How about mother earth !\n" );
          }
          return 0 ;
}
```

End of Lecture