Lecture 8.1

Topics

1. Conditional Statements/Structures – **`if`**, **`if-else`**
2. Extended **`if-else if-else`**

____

1. Conditional Statements/Structures – `if`, `if-else`

There are several conditional structures such as `if`, `if-else`, `if-else-if`, `switch`. Two of these structures are introduced here:

(1) **`if`**, and

(2) **`if–else`**

1.1 Simple `if` Structure – Flowchart

Syntax of `if` Structure

```
if ( testExpression ) {
  /*if testExpression is true, do something here.*/
}
```

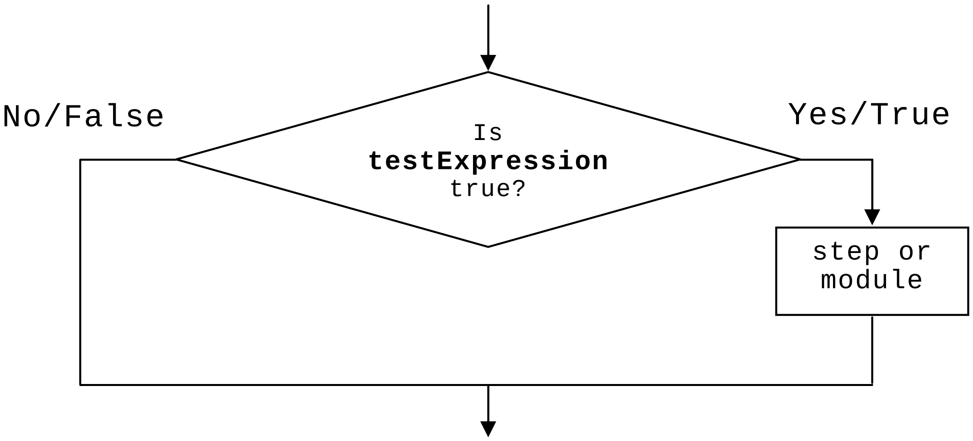

Figure 1 Conditional structure with one option

Example – Conditional **`if`** structure

```
/**
 *Program Name:  cis26L0811.c
 *Discussion:    if Structure
 */

#include <stdio.h>

/*Function prototypes*/
void printPositive( int );
void getPrintPositive( void );

/*Application driver*/
int main() {
  int iA;
  printf( "1234567890123456789012345678901234567890\n" );

  printf( "\nEnter an integer value: " );
  scanf( "%d", &iA );

  printPositive( iA );

  getPrintPositive( );

  printf( "\n" );
  return 0;
}

/*Function definitions*/
/**
 *Function Name: printPositive()
 *Description  : Printing information for an integer
 *Pre          : Integer to be assessed and printed
 *Post         : None
 */
void printPositive( int iOld ) {
  if ( iOld > 0 ) {
    printf( "\n%d is positive.\n", iOld );
  }

  return;
}

/**
 *Function Name: getPrintPositive()
 *Description  : Printing information for an integer
 *Pre          : Nothing
 *Post         : None
 */
void getPrintPositive( void ) {
  int iA;

  printf( "\nEnter an integer value: " );
  scanf( "%d", &iA );

  if ( iA > 0 ) {
    printf( "\n%d is positive.\n", iA );
  }

  return;
}
```

OUTPUT – Run #1

```
1234567890123456789012345678901234567890

Enter an integer value: 8

8 is positive.

Enter an integer value: 9

9 is positive.
```

OUTPUT – Run #2

```
1234567890123456789012345678901234567890

Enter an integer value: -8

Enter an integer value: 9

9 is positive.
```

OUTPUT – Run #3

```
1234567890123456789012345678901234567890

Enter an integer value: -8

Enter an integer value: -9
```

In the above flowchart, if the **`testExpression`** is **false** then no action would take place. However, it is possible to have different action for this (false) option with respect to the other (true) case.

Let’s look at this conditional structure next.

1.2 `if-else` Flowchart

Syntax of `if-else` Structure

```
if ( testExpression ) {
  /*if testExpression is true, do something here.*/
} else {
  /*if testExpression is false, do something else.*/
}
```

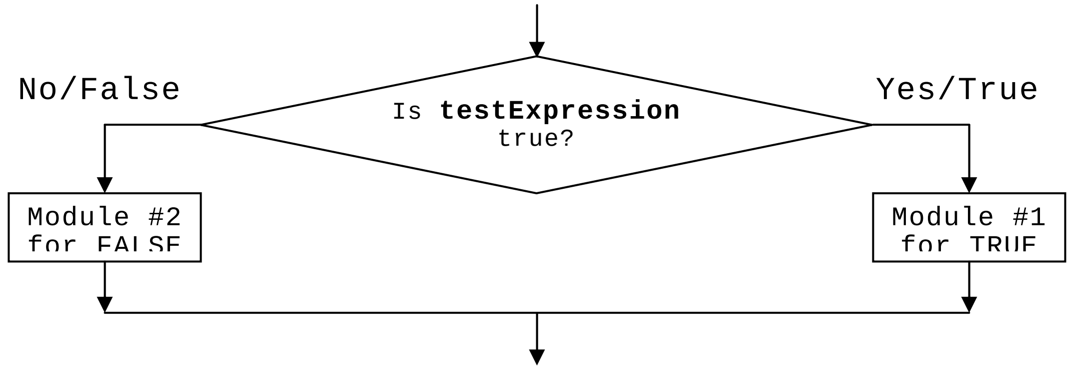

Figure 2 `if-else` conditional structure

Let’s look at a revised version of the above functions.

```
/**
 *Function Name: printPositiveNegative()
 *Description  : Printing information for an integer
 *Pre          : Integer to be assessed and printed
 *Post         : None
 */
void printPositiveNegative( int iOld ) {
  if ( iOld > 0 ) {
    printf( "\n%d is positive.\n", iOld );
  } else {
    printf( "\n%d is non-positive.\n", iOld );
  }

  return;
}

/**
 *Function Name: getPrintPositive()
 *Description  : Printing information for an integer
 *Pre          : Nothing
 *Post         : None
 */
void getPrintPositive( void ) {
  int iA;

  printf( "\nEnter an integer value: " );
  scanf( "%d", &iA );

  if ( iA > 0 ) {
    printf( "\n%d is positive.\n", iA );
  } else {
    printf( "\n%d is non-positive.\n", iOld );
  }

  return;
}
```

2. Extended Conditional Structure – `if-else-if-else`

In many scenarios, there may be more than two choices to be considered; for examples, determining one of the seven (7) days in a week, twelve (12) months in a year, etc.

In these cases, the **extended form** of **`if-else`** structure can be used.

The flowchart is given as follows,

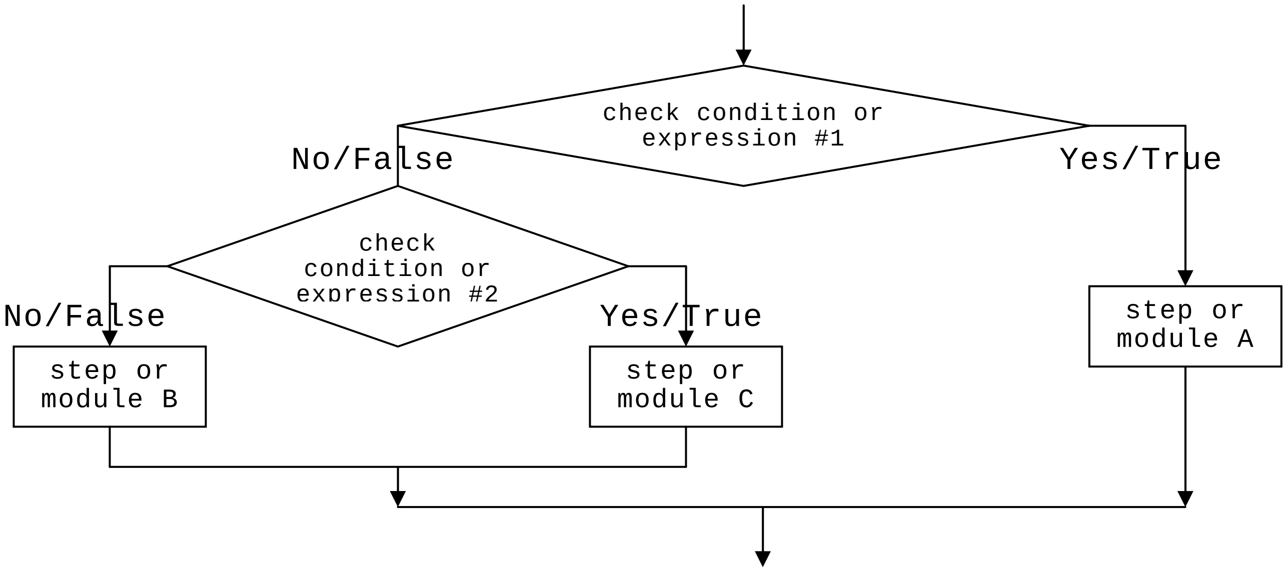

Figure 3 Extended `if-else-if` conditional structure

The general syntax of the extended **`if-else-if`** structure is given as follows,

```
if ( testExpression1 ) {

  //if testExpression1 is true, perform option #1 here.

} else if ( testExpression2 ) {

  //if testExpression2 is true, perform option #2 here;
  //  this also means that testExpression1 is false.

} else {

  //if testExpression2 is false, perform last option;
  //  this also means that
  //    testExpression1 and testExpression2 are both false.

}
```

The above extended structure **can also be extended further and further** as needed. The code structure below illustrates the idea.

```
if ( testExpression1 ) {

  //if testExpression1 is true, perform option #1 here.

} else if ( testExpression2 ) {

  //if testExpression2 is true, perform option #2 here;
  //  this also means that testExpression1 is false.

} else if ( testExpression3 ) {

  //if testExpression3 is true, perform option #3 here;
  //  this also means that
  //    testExpression1 and testExpression2 are both false.

} else {

  //if testExpression3 is false, perform last option;
  //  this also means that ALL test expressions are false.

}
```

The following example with function `printDay()` shows how the `if-else-if` extension can be written.

```
void printDay( int iDay ) {
  if ( iDay == 1 )   {
    printf( "\nIt is Sunday!" );
  } else if ( iDay == 2 ) {
    printf( "\nIt is Monday!" );
  } else if ( iDay == 3 ) {
    printf( "\nIt is Tuesday!" );
  } else if ( iDay == 4 ) {
    printf( "\nIt is Wednesday!" );
  } else if ( iDay == 5 ) {
    printf( "\nIt is Thursday!" );
  } else if ( iDay == 6 ) {
    printf( "\nIt is Friday!" );
  } else if ( iDay == 7 ) {
    printf( "\nIt is Saturday!" );
  } else {
    printf( "\nIt is an INVALID selection!" );
  }

  return;
}
```

Note that the test expression of **`iDay`** can be of **any value** of type **`int`**. This would mean that there are many values falling to the last option group (i.e., the last **`else`** block).

In particular, if the **same expression** (such as **`iDay`**) is used to test against different integral values (e.g., expressions of **`if`**s as in the above) then one may want to consider the use of a **`switch`** structure.

We will discuss **`switch`** structure next.