Examples
Using `if-else`

Outline

- **Example 1:** Computing square root
- **Example 2**: Finding the number of real roots of a quadratic equation.
- **Example 3**: Output three numbers in non-descending order.

Example 1

- **Objective:** To write a program to output the square root of a number.

- This example illustrates also how to use a pre-defined function to compute the square root of a number.

Example 1: Solution #1

```c
#include <stdio.h>
#include <math.h>                  // Need this line to use sqrt()

int main( void )
{
   double num , result ;

   scanf( "%lf" , & num );

   result = sqrt( num );
   printf( "The square root of %.4f is %.4f.\n" , num , result );

   return 0 ;
}
```

sqrt(num) computes the square root of num and yields the result in type double.

Example 1: Solution #2

```c
#include <stdio.h>
#include <math.h>                   // Need this line to use sqrt()

int main( void )
{
   double num , result ;

   scanf( "%lf" , & num );

   if ( num >= 0 )
   {
      result = sqrt( num );
      printf( "The square root of %.4f is %.4f.\n" , num , result );
   }
   else
      printf( "Can't compute square root for negative number.\n" );

   return 0 ;
}
```

Avoid taking square root on a negative number:

IND means “indeterminate”

Example 2

- **Objective:** Write a segment of code to output the number of real number solutions of a quadratic equation in the form $ax^2+bx+c = 0$. The code will read a, b, and c from the user. We assume $a \neq 0$.

- **Approach:**
  - Compute discriminant as b^2 - $4ac$
  - discriminant > 0 ➔ 2 real number solutions
  - discriminant = 0 ➔ 1 real number solution
  - discriminant < 0 ➔ 0 real number solutions

Example 2: Solution #1

```c
double a , b , c ;          // To store the coefficients
scanf( "%lf%lf%lf" , & a , & b , & c );

if ( b * b – 4 * a * c > 0 )
    printf( "# of real number solutions: 2\n" );

if ( b * b – 4 * a * c == 0 )
    printf( "# of real number solutions: 1\n" );

if ( b * b – 4 * a * c < 0 )
    printf( "# of real number solutions: 0\n" );
```

Any bug in the code?

Any issue in the code? ⇨ How to fix the issues?

Example 2: Solution #2

```
double a , b , c ;          // The coefficients
double dis ;                // The discriminant
int    sol ;                // # of real number solutions

scanf( "%lf%lf%lf" , & a , & b , & c );
dis = b * b – 4 * a * c ; // Compute discriminant once

if ( dis > 0 )
   sol = 2 ;
else
if ( dis == 0 )
   sol = 1 ;
else                        // Otherwise dis < 0
   sol = 0 ;

printf( "# of real number solutions: %d\n" , sol );
```

Notes about the issues in Example 2.

- **Efficiency**: In solution #2, `b*b-4*a*c` is only evaluated once, and thus the amount of computation is reduced.
- **Efficiency**: by using “else,” testing twice is sufficient for mutually-exclusive situations.
- **Code Maintenance**: Solution #2 uses one `printf()` to output the result. The advantage is, if we need to change the output format, we only need to change one `printf()` statement.

Example 3

- **Objective:** Write a segment of code to read three integers from a user and output them in non-descending order.
  - Assume the values are stored in variables **x**, **y**, and **z**

- Approach #1:
  - For each of the six possible arrangements, output the result accordingly: (1) **x** ≤ **y** ≤ **z**, (2) **x** ≤ **z** ≤ **y**, (3) **y** ≤ **x** ≤ **z** , (4) **y** ≤ **z** ≤ **x**, (5) **z** ≤ **x** ≤ **y**, and (6) **z** ≤ **y** ≤ **x**

- Approach #2:
  - Sort the values of **x**, **y**, and **z** so that **x** ≤ **y** ≤ **z**

Example 3: Solution #1.1

```
int x , y , z ;                  // To store input values
scanf( "%d%d%d" , & x , & y , & z );

if ( x <= y && y <= z )
   printf( "%d %d %d\n" , x , y , z );
if ( x <= z && z <= y )
   printf( "%d %d %d\n" , x , z , y );
if ( y <= x && x <= z )
   printf( "%d %d %d\n" , y , x , z );
if ( y <= z && z <= x )
   printf( "%d %d %d\n" , y , z , x );
if ( z <= x && x <= y )
   printf( "%d %d %d\n" , z , x , y );
if ( z <= y && y <= x )
   printf( "%d %d %d\n" , z , y , x );
```

Any issue in the code?

Without "else", this would produce multiple outputs when two or more inputs have the same value.

e.g., when x, y, z are all 6, the output "6 6 6" would appear six times.

Example 3: Solution #1.2

```c
int x , y , z ;                    // To store input values
scanf( "%d%d%d" , & x , & y , & z );

if ( x <= y && y <= z )
   printf( "%d %d %d\n" , x , y , z );
else
if ( x <= z && z <= y )
   printf( "%d %d %d\n" , x , z , y );
else
...
else
if ( z <= x && x <= y )
   printf( "%d %d %d\n" , z , x , y );
else
   printf( "%d %d %d\n" , z , y , x );
```

Example 3: Solution #2

```c
int x , y , z , tmp ;
scanf( "%d%d%d" , & x , & y , & z );

// First, make sure x holds the smallest value
if ( x > y ) {                        // If y is smaller
   tmp = x ; x = y ; y = tmp ;        // swap x and y
}
if ( x > z ) {                        // If z is even smaller
   tmp = x ; x = z ; z = tmp ;        // swap x and z
}

// Next, make sure y <= z
if ( y > z ) {                        // If z is smaller
   tmp = y ; y = z ; z = tmp ;        // swap y and z
}

printf( "%d %d %d\n" , x , y , z );
```

Example 3: Solution #3

```c
int x , y , z , sum , maxV , minV ;
scanf( "%d%d%d" , & x , & y , & z );

// There are many other ways to solve this problem, e.g.,

......

printf( "%d %d %d\n" , minV , sum-minV-maxV , maxV );
```