COMPUTER PROGRAMMING

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
UNIVERSITY OF WEST ATTICA

DEPARTMENT OF INFORMATION AND COMPUTER ENGINEERING

3
CONTROL STRUCTURES

STUDENT DETAILS:

NAME: ATHANASIOU VASILEOS EVANGELOS
STUDENT ID: ice19390005
STUDENT STATUS: GRADUATE
PROGRAM OF STUDY: UNIWA
LABORATORY DEPARTMENT: M2
LABORATORY TEACHER: GEORGIOS MELETIOU
DATE TOTAL COMPLETION OF WORK: 23/11/2021

COMPUTER PROGRAMMING

CONTENTS

COMPUTER PROGRAMMING

COMPUTER PROGRAMMING

ANSWERS TO THEORY SUBJECTS

SUBJECT 1

a) What do you know about the " if " statement? An "

if " statement is a statement that checks the result of an expression to execute the appropriate statements depending on that result. For an expression " p 1" the structure of " if " is as follows:

```
if ( p 1) printf (" Hello world ");
printf (" End of if ");
```

In more detail, if the expression " p 1" produces a " True " result, i.e., a result other than zero (0), then the command inside the " if " will be executed and in this particular case it is " printf (" Hello world ");", where the message " Hello world '. Alternatively, if the result it produces is " False " (0), then the commands, if any, that are not inside the " if " will be executed (In this particular case, the " printf (" End of if ")", where the message " End of if ').
The expression " p 1" can contain any arithmetic, relational, logical, binary operation that produces a value of " False " (0) and one " True " (other than "0"). For example it can be a relational condition (x > y), an arithmetic operation (x + y), a variable assignment (x = 1), a logical condition (x && y), a constant (5), a character ('A'), a string (' Hello world ') and so on

In addition to the " if " command, there is also the " if - else " command, which is also a control command. For an expression " p 1" the structure of " if - else " is as follows:

```
if ( p 1)
printf (" Hello world ");
else
printf (" Goodbye world ");
```

More specifically, if the expression " p 1" produces a result of " True " (different from the zero "0"), then the command inside the " if " will be executed and in this particular case it is " printf (" Hello world ");", where the message " Hello world '. Alternatively, if the result it produces is " False " (0), then the commands inside the " else " will be executed and in this particular case it is " printf (" Goodbye wolrd ");", where the message " Goodbye world ?".

Inside an " if " we can put another " if " or " if - else ".

For example :

```
1 if (x >= 0)
2 {
3 if (x == 0)
4 printf ("zero");
5 else
6 printf ("positive");
7 }
8 else
9 printf (" negative ");
```

In more detail, the " if " of line "1" checks whether the expression " x >=0" produces a result of "0" or different from "0" (" False " or " True " respectively). If it produces a result other than "0" (True), then the command inside the " if " which is another " if " is executed (line 3). The " if " in line "3" checks whether the expression " x == 0" produces a result of "0" or other than "0" (" False " or " True "). If it produces a result other than "0" (True), then the command inside the " if " is executed, which is " printf (" zero ");", where it will print the message " zero " (line 4), that is, that the content of variable " x " is "0". Alternatively, the command inside " else " is executed which is " printf (" positive "?"), where it will print the message " positive ", that is, that the content of the variable " x " is a positive number. On the other hand, if the initial condition of the first " if " (x >= 0) of line "1" produces a result of "0" (False), then the statement inside the " else " of line 9 is executed which is the statement " printf (" negative ");", that is, that the content of the variable " x " is a negative number.

b) What do you know about the " switch " command?

switch " instruction is an instruction that checks the value of an integer constant, in order to execute instructions depending on this value. For an integer constant " x " the structure of " switch " is:

```
switch ( x )
{
case 1 : printf ("Hello");
 case 2 : printf ("world");
   default : printf ("\ n ");
}
```

More specifically, " switch " checks the value of the integer variable " x ". If the variable contains the value "1", then the commands in " case 1", " case 2" and " default " are executed, which in this particular case are " printf (" Hello "?"), " printf (" world ");" and " printf ("\ n ");" respectively, where the messages " Hello ", " world " and the escape character of the change will be printed

line '\ n '. In case the variable contains the value "2", then the commands in " case 2" and " default " are executed. Finally, in the event that the variable contains a value beyond the values contained in " case " (1 and 2), only the commands in " default " are executed.
Additionally, there is the ' break ' command which essentially breaks the flow of the program within the ' switch ' command window and the control is transferred outside of it. For example :

```
switch (x)
{
case 1 : printf ("Hello"); breaks?
case 2 : printf ("world"); breaks?
 default : printf ("\ n ");
}
```

More specifically, " switch " checks the value of the integer variable " x ". In case the variable contains the value "1", then the commands in " case 1" are executed, that is, " printf (" Hello "?"), where the message " Hello " and " break " will be printed. The program flow is then moved outside the " switch " command window between the "{ , }" brackets. The same applies to the case where the variable " x " contains the value "2", where the commands in " case 2" and the command " break ?" will be executed .

ANSWERS TO SOURCE CODE ISSUES

TOPIC 2

HIGHLIGHT « if . c » / « if 1. c » / switch . c

In this section, the comments and observations
regarding the " if " programs are recorded . c ", " if 1. c " and " switch . c ».

DOCUMENTATION « if . c »

if " program . c ” based on “ if - else ” control statements reads
from the “ standard ” input three integers and finds the maximum number, as well as the sequence entered. This check is done with nested " if - else " commands, where in each case these three numbers given by the user are checked to see which is the largest and with which row(s) it was entered or entered, as the same number may have been read more than once. The program succeeds in showing the operation of nested

of " if - else " statements. “

if 1. c ”

DOCUMENTATION

The “if 1. c ” program based on the “ if ” control statement and the ternary operator “? :" performs the same algorithm as that of the program " if . c " (see Program " if . c "). The difference lies in the syntax and control commands. In more detail, in the program " if 1. c " an integer variable " Max " is defined, where the initial value has the value of the first integer read from the " standard " input and indicates the maximum number. Then an " if " checks between " Max " and the second number to see if this is the maximum. By using the ternary operator, the third number is checked to see if it is the maximum. Finally, with the use of " if " it is checked which number has been registered in the variable " Max " and the entered sequence is printed. The check is made for all three variables read from the " standard " input, as the same number may have been read more than once.

DOCUMENTATION « switch . c »

The program based on the control command " switch " succeeds in clarifying the operation of the command " switch " and " break ". In detail, it checks the value of the integer variable " i " which is "7" and executes commands from " case 3 + 4" (case 7), where the message "7" and the newline escape character "\ n " are printed . It then executes from " default " the command, where the message " default " and the newline escape character "\ n " are printed. The same from the " case 1" command, where the message "1" and the newline escape character "\ n " are printed. Finally, " break " is also executed immediately after " case 1" and the program flow is transferred outside the command window of " switch ".

SOURCE CODES / DOCUMENTATION

SUBJECT 3

LABEL '2 ndGradeEquation . c »

The "Program "2 ndGradeEquation . c "" (Source Code) and the “Documentation “2 ndGrade . c "" (Question, Structure, Variables, Traversal, Examples, Remarks) answer the question of “Topic 3”

PROGRAM " 2ndGradeEquation.c"

```
1 #include < stdio.h >
2 #include < math.h >
3
4 int main (int argc , char ** argv )
5
6 {
7
8      system (" chcp 1253");
9
10     double a, b, c; // Declaration variables
11     double x?
12     double D;
13     double root _D ;
14     doubles x1, x2;
15     double x1_2;
16
17     printf ("==========================================================
=====
\n\n");
18     printf ("Solving quadratic equation\ n \ n "); // Title of the
program
19     printf ("==========================================================
=====
\ n \ n ");
20     printf ("Enter the coefficient a : ");
21     scanf ("% lf ", & a ); // Enter the 1st coefficient " a " of
the equation
22     printf ("Enter coefficient b : ");
23     scanf ("% lf ", & b ); // Enter the 2nd coefficient " b " of
the equation
24     printf ("Enter the coefficient c : ");
25     scanf ("% lf ", & c ); // Enter the 3rd coefficient " c " of
the equation
26     printf ("\ n ------------------------------------------------
-- ------\ n \ n ");
27     printf ("Coefficient a : [%20.6 lf ]\ n ", a ); // Print the
1st coefficient " a " of the equation
28     printf ("Coefficient b : [%20.6 lf ]\ n ", b ); // Print the
2nd coefficient " b " of the equation
```

```
29     printf ("Coefficient c : [%20.6 lf ]\ n \ n ", c ); // Print
the 3rd coefficient " c " of the equation

30     if ( a != 0) /* (~) Quadratic equation */

31     {

32           D = ( b * b ) - 4 * a * c ; // Calculate the
discriminant

33           printf ("Equation is quadratic\ n \ n ");

34           printf ("Different : [%20.6 lf ]\ n \ n ", D ); // Print
the discriminant

35           if ( D >= 0) /* (!) Positive differential or equal to
"0" */

36           {

37                 if ( D == 0) /* (+) Discriminator equal to "0" */

38                 {

39                       x 1_2 = - b / (2 * a ); // Calculate the
double real root

40                       printf ("Equation is undefined\ n ");

41                       printf ("Real root x 1_2 is double\ n \ n
");

42                       printf ("Root x 1_2 : [%20.6 lf ]\ n \ n ",
x 1_2); // Print the double real root

43                       printf ("----------------------------------
------------------------- ----\ n \ n ");

44                 }

45                 else /* (+) Discriminating positive */

46                 {

47                       root _ D = sqrt ( D ); // Calculation of
the square root of the discriminant

48                       x 1 = (- b + root _ D ) / (2 * a );
// Calculate the first real root

49                       x 2 = (- b - root _ D ) / (2 * a ); //
Calculate the second real root

50                       printf ("Number of real roots : 2\ n \ n
");

51                       printf ("Root x 1 : [%20.6 lf ]\ n ", x 1);
// Print the first real root

52                       printf ("Root x 2 : [%20.6 lf ]\ n \ n ", x
2); // Print the second real root

53                       printf ("----------------------------------
-------------- ----\ n \ n ");

54                 }

55           }

56           else /* (!) Negative delimiter */
```

```
57            {
58                  printf ("Equation is impossible\ n \ n ");
59                  printf ("Number of real roots : 0\ n \ n ");

60                  printf ("----------------------------------------
------- -----\ n \ n ");
61            }
62      }
63      else /* (~) Primary equation */
64      {
65            x = - c / b ; // Calculate the real root
66            printf ("Equation is prime\ n ");
67            printf ("Number of real roots : 1\ n \ n ");
68             printf ( "Root x : [%20.6 lf ]\ n \ n ", x ); // Print
the actual root
69         printf ("----------------------------------------------------
-------------- ---\ n \ n ");
70      }
71
72      return 0;
73
74 }
```

DOCUMENTATION '2 ndGradeEquation . c »

REQUIRED

The program “2 ndGradeEquation . c "achieves the following functions:

a) Reads from the " standard " input the three coefficients of a quadratic equation in the set of reals (type " double ").

b) Checks the contents of the coefficients.

c) Calculates the roots of the equation according to the values entered in the coefficients.

d) Prints the appropriate messages (for the type of equation and the number of its roots).

STRUCTURE

In order to implement the requested, initially,

the libraries :

a) " stdio . h " : Contains the ready-made functions " scanf ()" and " printf () attached to the input and output channels respectively to read and print contents of the respective variables. Also ' printf ()' was used to print characteristic messages for better understanding of the source code.

b) " math . h » : Contains the ready-made mathematical function " sqrt ()" to calculate the discriminant of the quadratic equation.

Additionally, the characteristic operators were used :

a) numeric : +, -, *, /

b) relational : !=, >=, ==

c) assignment : =

d) & operator : For the variable address as the second argument of the function " standard " input, " scanf ().

The control commands:

a) if - else

VARIABLES

Double-precision real variables (of type " double ")

a (The coefficient " a " of the quadratic equation)

b (The coefficient " b " of the quadratic equation)

c (The coefficient " c " of the quadratic equation)

x (The root of the equation when " a = 0")

D (The discriminant)

root _ D (The root of the discriminant)

x 1 (The first root of the equation, when " a != 0" and " D > 0)

x 2 (The second root of equation, when " a != 0" and " D > 0)

x 1_2 (The double root of the equation, when " a != 0" and " D == 0)

DIVISION

Declaration of variables (lines 10-15)

The program starts with the declaration of variables on lines "10-15" (for details see .subsection “Variables”).

Program title (line 18)

In line "18" with a " printf ()" function, the title of the program (a*x^2 + b*x + c = 0) is printed from the " standard " output, as well as two escape characters of this line break (\ n).

Input of the 1st coefficient "a" of the equation (line 21)

In line "21" with the function " scanf ()" the first real coefficient " a " of the equation is read from the " standard " input . With the " printf ()" function, the corresponding message is printed from the " standard " output (line 20).

Input of the 2nd coefficient " b " of the equation (line 23)

In line "23" with the function " scanf ()" the second real coefficient " b " of the equation is read from the " standard " input . Then, with the " printf ()" function, the corresponding message is printed from the " standard " output (line 22).

Input of the 3rd coefficient " c " of the equation (line 25)

In line "25" with the function " scanf ()" the third real coefficient " c " of the equation is read from the " standard " input . Then, with the " printf ()" function, the corresponding message is printed from the " standard " output (line 24).

Printing the 1st coefficient "a" of the equation (line 27)

In line "27" the content of the variable " a " (the first coefficient of the equation) is printed from the " standard " output with a function " printf ()" accompanied by the appropriate message. It is worth noting that the

alphanumeric format is "%20.6 lf ", that is, the result will be printed immediately after "20" blank characters with "6" decimal places, since this is a real double precision. This is intended to evenly align the de-results.

Printing the 2nd coefficient " b " of the equation (line 28)

In the line "2 8" the content of the variable " b " (the second coefficient of the equation) is printed from the " standard " output with a function " printf ()" accompanied by the appropriate message. It is worth noting that the alphanumeric format is "%20.6 lf ", that is, the result will be printed immediately after "20" blank characters with "6" decimal places, since this is a real double precision. This is to align the results evenly.

Printing the 3rd coefficient " c " of the equation (line 29)

In the line "2 9" the content of the variable " c " (the third coefficient of the equation) is printed from the " standard " output with a function " printf ()" accompanied by the appropriate message. It is worth noting that the alphanumeric format is "%20.6 lf ", that is, the result will be printed immediately after "20" blank characters with "6" decimal places, since this is a real double precision. This is to align the results evenly.

(~) Quadratic equation (lines 30-62)
(~) Primary equation (lines 6 3-70)

In lines "30-70" a check for the content of the first coefficient " a " of the equation is implemented with an " if - else " control command .
In detail, if " a " does not contain the value "0", then the statements of the subsection "Quadratic Equation (lines 30-62)" will be executed, otherwise if it contains the value "0", then the statements of the subsection " Primary equation (lines 63-70)'.

(~) Quadratic equation (lines 30-62)

In lines "30-62" the commands are executed in the case that the expression in " if " (a != 0) of line "30" produces a result of a value " True " (the coefficient " a " does not contain the value "0 ").

Calculation of the discriminant (line 32) In line "32"

the discriminant for calculating the real roots of the quadratic equation is calculated with the numerical expression "(b * b) – 4 * a * c " and the result is entered in the

variable " D ".

Printing the discriminator (line 34) In line "35"

the content of the variable " D " (the discriminator) is printed from the " standard " output with a function " printf ()" accompanied by the appropriate message. It is worth noting that the alphanumeric format is "%20.6 lf ", that is, the result will be printed immediately after "20" blank characters with "6" decimal places, since this is a real double precision. This is to align the results evenly.

(!) Positive delimiter or equal to "0" (lines 35-56)
(!) Negative delimiter (lines 56-61)

In lines "3 5-61" a control statement " if - else " is implemented for content of the discriminator (variable " D ").
In detail, if " D " contains a value greater than or equal to "0", then the statements of the "Positive discriminant or equal to "0" (lines 35-56)" subsection will be executed, otherwise if it contains a value less than "0 ", then the commands in the “Negative discriminant (lines 56-61)” subsection will be executed.

(!) Positive delimiter or equal to "0" (lines 35-56)

Lines "3 5-56" execute the commands for the case that the expression in " if " (D >= 0) of line "35" produces results in a " True " value (the " D " qualifier contains a value equal to or greater than "0").

(**+) Discriminator equal to "0" (lines 37-44) (+**
) **Discriminator positive (lines 45-54)**

In lines "3 7-54" a check for the content of the discriminant (variable " D ").
In detail, if " D " contains a value equal to "0", then the statements of the subsection "Distinct equal to "0" (lines 37-44)" will be executed, otherwise if it does not contain a value equal to "0", then the statements in the "Discriminative positive (lines 45-54)" subsection will be executed.

(+) Discriminator equal to "0" (lines 37-44)

In lines "3 7-44" the commands are executed for the case that the expression in " if " (D == 0) of line "37" produces a result of " True " value (the " D " delimiter contains a value equal to "0").

Calculation of the double real root (line 39)

In line " 39" the double real root of the quadratic indefinite equation is calculated with the numerical expression "- b / (2 * a)" and entered in the variable " x 1_2".

Printing the double real root (line 42)

In the line "4 2" the content of the variable " x 1_2" (η) is printed from the " standard " output with a function " printf ()" accompanied by the appropriate message. It is worth noting that the alphanumeric format is "%20.6 lf ", that is, the result will be printed immediately after "20" blank characters with "6" decimal places, since this is a real double precision. This is to align the results evenly.

(+) Discriminating positive (lines 45-54)

Lines " 45-54" execute the statements of " else " (line 45) corresponding to the " if " of line "37" for the case that the expression " D == 0" results in a " False " value (the discriminator " D " does not contain a value equal to "0" and is greater than "0").

Calculation of the square root of the discriminant (line 47)

In line "47" the real double-precision square root of the discriminant is calculated with the arithmetic expression " sqrt (D)" and the result is stored in the variable " root _ D ".

Calculation of the first real root (line 48)

In the line "4 8" is calculated with the numerical expression "(- b + root _ D) / (2 * a)", the first double-precision real root of the quadratic equation and the result is recorded in the variable " x 1".

Calculation of the second real root (line 49)

In line " 49" is calculated with the numerical expression "(- b - root _ D) / (2 * a)", the second real root of double precision of the quadratic equation and the result is entered in variable " x 2".

Print the first real root (line 51)

In line "51" the content of the variable " x 1" (the first real root of the quadratic equation) is printed from the " standard " output with a function " printf ()" accompanied by the appropriate message. It is worth noting that the alphanumeric format is "%20.6 lf ", that is, the result will be printed immediately after "20" blank characters with "6" decimal places, since this is a real double precision. This is to align the results evenly.

Printing the second real root (line 52)

In line "5 2" the content of the variable " x 2" (the second real root of the quadratic equation) is printed from the " standard " output with a function " printf ()" accompanied by the appropriate message. It is worth noting that the alphanumeric format is "%20.6 lf ", that is, the result will be printed immediately after "20" blank characters with "6" decimal places, since this is a real double precision. This is to align the results evenly.

(!) Negative delimiter (lines 56-61)

In lines "56-61" the commands of " else " (line 56) are executed in the case that the expression in " if " (D >= 0) of line "35" produces a result of a " False " value (the " D " discriminator does not contain a value equal to "0" and is less than "0"). In lines "58-59" messages about the type of equation (indefinite) and the number of its real roots (0) are printed from the "

standard " output with the help of the " printf ()" function. **(~) Primary equation (lines 63-70)**

In lines "63-70" the commands of " else " (line 63) corresponding to the " if " of line "30" are executed in the case that the expression " a != 0" produces a result of a " False " value (the factor " a " contains the value "0").

Calculation of the real root (line 65)

In line "65" the real root of the primary equation is calculated with the numerical expression "- c / b " and the result is entered in the variable " x ".

Print the actual root (line 68)

In line "6 8" the content is printed from the " standard " output

of the variable " x " (the real root of the quadratic equation) with a function " printf ()" accompanied by the appropriate message. It is worth noting that the alphanumeric format is "%20.6 lf ", that is, the result will be printed immediately after "20" blank characters with "6" decimal places, since this is a real double precision. This is to align the results evenly.

EXAMPLES

Example 1 (a == 0)

========== ==========

Solving a quadratic equation

========== =

Enter the factor a : 0.0

Enter coefficient b : -11.0

Enter the factor c : 3.0

---------- ----------

Factor a : [0.000000]

Factor b : [-11.000000]

Coefficient c : [3.000000]

The equation is primary

Number of real roots: 1

Root of x : [0.272727]

---------- ----------

Example 2 (a != 0 / D >= 0 / D == 0)

================================ ==================

Solving a quadratic equation

== =

Enter the factor a : 2.0

Enter the factor b : 4.0

Enter the coefficient c : 2.0

---------- ----------

Factor a : [2.000000]

Factor b : [4.000000]

Coefficient c : [2.000000]

The equation is quadratic

Distinguishing : [0.000000]

The equation is indefinite

The real root x 1_2 is double

Root x 1_2 : [-1.000000]

---------- ----------

Example 3 (a != 0 / D >= 0 / D != 0)

================================== ============

Solving a quadratic equation

========== =

Enter the factor a : 2.8

Enter coefficient b : -7.1

Enter the coefficient c : -1.6

---------- ----------

Factor a : [2.800000]

Factor b : [-7.100000]

Coefficient c : [-1.600000]

The equation is quadratic

Distinctive : [68.330000]

Number of real roots: 2

Root x 1 : [2.743964]

Root x 2 : [-0.208249]

---------- ----------

Example 4 (a != 0 / D <0)

========== =======

Solving a quadratic equation

========== =

Enter the factor a : 32.679

Enter the factor b : 1.0

Enter the coefficient c : 16.21456

---------- ----------

Factor a : [32.679000]

Factor b : [1.000000]

Coefficient c : [16.214560]

The equation is quadratic

Distinctive : [-2118.502425]

The equation is impossible

Number of real roots: 0

---------- ----------

COMMENTS

The correctness of the results in the above examples was also checked using a standard calculator and no difference was found.
In detail, we have four examples where each one corresponds to special conditions found inside the corresponding " if - else " statements:

Example 1 (a == 0)

In "Example 1" the first coefficient " a " of the equation (0), the second coefficient " b " (-11) and the third " c " (3) are read from the " standard " input . These three numbers are correctly printed, the first check is made for the content of " a " which contains the value "0", so it is a primary equation with a real root and this root (x) is correctly printed. The results :

a : 0.000000

b : -11.000000

c : 3.000000

x : 0.272727

Example 2 (a != 0 / D >= 0 / D == 0)

In "Example 2" the first coefficient " a " of equation (2), the second coefficient " b " (4) and the third " c " (4) are read from the " standard " input . These three numbers are printed correctly, the first check is made for the content of " a " that does not contain the value "0", so this is a quadratic equation. The result of the discriminant (0) is correctly calculated and printed, the second check is made for the content of " D " which contains a value greater than or equal to "0", so it definitely has real roots. The third check is made for the content of " D " which contains the value "0", so it is an indefinite equation with a double real root (x 1_2). The results :

a : 2.000000

b : 4.000000

c : 2.000000

D : 0.000000

x 1_2 : -1.000000

Example 3 (a != 0 / D >= 0 / D != 0)

In "Example 3" it is read from the " standard " input the first coefficient " a " of equation (2.8), the second coefficient " b " (-7.1)

and the third " c " (-1.6). These three numbers are printed correctly, the first check is made for the content of " a " that does not contain the value "0", so this is a quadratic equation. The result of the discriminant (68.33) is correctly calculated and printed, the second check is made for the content of " D " which contains a value greater than or equal to "0", so it definitely has real roots. The third check is made for the content of " D " which does not contain the value "0", so it is an equation with two real roots (x 1, x 2). The results :

a : 2.800000

b : -7.100000

c : -1.600000

D : 68.300000

x 1 : 2.743964

x 2 : -0.208249

Example 4 (a != 0 / D < 0) :

In "Example 4" read from " standard " input the first coefficient " a " of the equation (32.679), the second coefficient " b " (1) and the third " c " 16.21456). These three numbers are printed correctly, the first check is made for the content of " a " that does not contain the value "0", so this is a quadratic equation. The result of the discriminant (-2118.502425) is correctly calculated and printed, the second check is made for the content of " D " containing a value less than "0", so it is an impossible equation without real roots. The results :

a : 32.679000

b : 1.000000

c : 16.214560

D : -2118.502425

TOPIC 4

LABEL " MaxInteger . c »

The "Program " MaxInteger . c "" (Source Code) and the "Documentation " MaxInteger . c "" (Question, Structure, Variables, Traversal, Examples, Remarks) answer the question of "Topic 4".

PROGRAM " MaxInteger.c "

```
1 #include < stdio.h >
2
3 int main (int argc , int ** argv )
4
5 {
6
7      system (" chcp 1253");
8
9      int a, b, c; // Declaration variables
10
11     printf   ("=================================================
=====
\n\n");
12     printf ("Find max number and row entered\ n \ n "); // Program
title
13     printf   ("=================================================
=====
\ n \ n ");
14     printf ("Enter the 1st integer : ");
15     scanf ("% d ", & a ); // Input the first integer
16     printf ("Enter the 2nd integer : ");
17     scanf ("% d ", & b ); // Input the second integer
18     printf ("Enter the 3rd integer : ");
19     scanf ("% d ", & c ); // Input the third integer
20     printf ("\ n ------------------------------------------------
-- ------\ n \ n ");
21     printf ("The 1st integer : [%20 d ]\ n ", a ); // Print the
first integer
22     printf ("The 2nd integer : [%20 d ]\ n ", b ); // Print the
second integer
23     printf ("The 3rd integer : [%20 d ]\ n \ n ", c ); // Print the
third integer
```

```
24      ( a > b && a > c ) ? printf ("The largest number is the number
% d and entered 1st.", a ) : printf ("\ b "); // a > b AND a > c

25      ( a < b && b > c ) ? printf ( "Largest number is number % d and
entered 2nd.", b ) : printf ("\ b "); // a < b AND b > c

26      ( a < c && b < c ) ? printf ("The largest number is the number
% d and entered 3rd.", c ) : printf ("\ b "); // a < c AND b < c

27      ( a == b && a > c ) ? printf ("The largest number is the number
% d and 1st and 2nd entered.", a ) : printf ("\ b "); // a == b > c

28      ( a < b && b == c ) ? printf ("The largest number is the number
% d and 2nd and 3rd entered.", b ) : printf ("\ b "); // a < b == c

29      ( a > b && a == c ) ? printf ("The largest number is the number
% d and 1st and 3rd were entered.", c ) : printf ("\ b "); // a == c >
b

30      ( a == b && b == c && a == c ) ? printf ("The largest number is
the number % d and 1st, 2nd and 3rd were entered.\ n ", a ) : printf
("\ n "); // a == b == c

31

33      return 0;

33

34 }
```

DOCUMENT “ MaxInteger . c »

REQUIRED

The program “ MaxInteger . c " achieves the following functions:

a) Reads from the " standard " input three numbers in the set of integers (type int).

b) Compares the three numbers, which are stored in the contents of three variables.

c) Prints the appropriate messages clarifying which of the three numbers is the largest, as well as in which order it was entered or entered in the event that the same number is entered in more than one variable.

STRUCTURE

In order to implement the request, the following libraries were used, initially:

a) " stdio . h ” : Contains the ready-made functions “ scanf ()” and “

printf () connected to the input and output channels respectively for read-

meaning and printing the contents of the corresponding variables. Also ' printf ()' was used to print characteristic messages for better understanding of the source code.

Additionally , the characteristic operators were used:

a) condition: {? :}

b) relational : <, >, ==

c) assignment : =

d) operator & : For the variable address as the second argument of the " standard " input function, " scanf ().

VARIABLES

Integer variables (of type int)

a (the first integer)

b (the second integer)

c (the third integer)

DIVISION

Declaration of variables (line 9)

The program starts with the declaration of variables on line "9" (for details see subsection "Variables") .

Program title (line 12)

On line "12" with a " printf ()" function, the title of the program (Finding the maximum number and sequence entered) is printed from the " standard " output, as well as two escape characters of this line break (\ n).

Input of the first integer (line 15)

In line "15" with the function " scanf ()" the first integer is read from the " standard " input. Then, with the " printf ()" function, the corresponding is printed from the " standard " output

message (line 14).

Input of the second integer (line 17)

In line "17" with the function " scanf ()" the second integer is read from the " standard " input. Then, with the " printf ()" function, the corresponding message

is printed from the " standard " output (line 16). **Input of the third integer (line 19)**

In line "19" with the function " scanf ()" the third integer is read from the " standard " input. Then, with the " printf ()" function, the corresponding message

is printed from the " standard " output (line 18). **Print the first integer (line 21) In line "21" the content of the variable "**

a " (the first integer) is printed from the " standard " output with a function " printf ()" accompanied by the appropriate message. It is worth noting that the alphanumeric format is "%20 d ", that is, the result will be printed immediately after "20" blank characters. This is to align the results evenly.

Printing the second integer (line 22) In line "22" the content of the variable "

b " (the second integer) is printed from the " standard " output with a function " printf ()" accompanied by the appropriate message. It is worth noting that the alphanumeric format is "%20 d ", that is, the result will be printed immediately after "20" blank characters. This is to align the results evenly.

Printing the third integer (line 23)

In the line "2 3" the content of the variable " c " (the third integer) is printed from the " standard " output with a function " printf ()" accompanied by the appropriate message. It is worth noting that the alphanumeric format is "%20 d ", that is, the result will be printed immediately after "20" blank characters. This is to align the results evenly.

a > b AND a > c (line 24)

In line " 24" the case is drawn up, where the older

number is entered in the variable " x " and entered first. The use of the ternary condition operator '? :" is justified as follows:

Since the expression " a > b && a > c " returns a value of " True " (a value other than "0"), that is, the variable " a " contains a value greater than those of the variables " b " and " c " , then, with " printf ()" from the " standard " output the content of " a " is printed accompanied by the appropriate message and for the entered order. Otherwise, that is, the expression " a > b && a > c " returns a value of " False " (a value equal to " 0 ") and therefore, the variable " a " contains no value greater than those of the variables " b " and " c ", then the escape character of that one position back (\ b) is printed by " printf ()" from the " standard " output .

a < b AND b > c (line 25)

In line " 25" the case is written, where the largest number is entered in the variable " b " and entered second. The use of the ternary condition operator '? :" is justified as follows:

Since the expression " a < b && b > c " returns a value of " True " (a value other than "0"), that is, the variable " b " contains a value greater than those of the variables " a " and " c " , then, with " printf ()" from the " standard " output the content of " b " is printed accompanied by the appropriate message and for the entered order. Otherwise, that is, the expression " a < b && b > c " produces a result of " False " (value equal to " 0 ") and therefore, the variable " b " contains no value greater than those of the variables " a " and " c ", then the escape character that one position back (\ b) is printed by " printf ()" from the " standard " output .

a < c AND b < c (line 26)

In line "20" the case is written, where the largest number is entered in the variable " z " and a third one is entered. The use of the ternary condition operator '? :" is justified as follows:

Since the expression " a < c && b < c " returns a value of " True " (a value other than "0"), that is, the variable " c " contains a value greater than those of the variables " x " and " y " , then, with " printf ()" from the " standard " output, the contents of " c " are printed accompanied by the appropriate message and for the entered order. Otherwise, that is, the expression " a < c && b < c " returns a value of " False " (a value equal to " 0 ") and therefore, the variable " c " contains no value greater than those of the variables " a " and " b ", then the escape character that one position back (\ b) is printed with " printf ()" from the " standard " output .

a == b > c (line 27)

In line "27" the case is written, where the largest number is entered in the variables " a " and " b " and entered first and second. The use of the ternary condition operator '? :" is justified as follows:

Since the expression " a == b && b > c " returns a value of " True " (a value other than "0"), i.e., the variables " a " and " b " contain the same value which is greater than of the " c " variable , then, " printf ()" prints from the " standard " output the contents of " a " (or " b ") accompanied by the appropriate message and for the entered order. Otherwise, that is, the expression " a == b && b > c " returns a value of " False " (a value equal to "0") and therefore, the variables " a " and " b " do not contain a value greater than that of the variable " a " and " b ", then, with " printf ()" from the " standard " output the escape character that one position back (\ b) is printed.

a < b == c (line 28)

In line "2 8" the case is written, where the largest number is entered in the variables " b " and " c " and a second and third one is entered. The use of the ternary condition operator '? :" is justified as follows :

Since, the expression " a < b && b == c " produces a result of a value " True " (a value other than "0"), that is, the variables " b " and " c " contain the same value that is greater than that of the variable " a ", then, with " printf ()" from the " standard " output the contents of " b " (or " c ") are printed with the appropriate message and for the order that was introduced. Otherwise, i.e., the expression " a < b && b == c " returns a value of " False " (a value equal to "0") and therefore, the variables " b " and " c " do not contain a value greater than that of the variable " a ", then, the escape character that one position back (\ b) is printed with " printf ()" from the " standard " output .

a == c > b (line 29)

In line "2 9" the case is drawn, where the largest number is entered in the variables " a " and " c " and entered first and second. The use of the ternary condition operator '? :" is justified as follows :

Since, the expression " a > b && a == c " produces a result of a value " True " (a value other than " 0 "), that is, the variables " a " and " c "

contain the same value that is greater than that of the variable " b ", then, print with " printf ()" from the " standard " output the contents of " c " (or " a ") accompanied by the appropriate message and for the row entered. Otherwise, that is, the expression " a > b && a == c " returns a value of " False " (a value equal to " 0 ") and therefore, the variables " a " and " c " do not contain a value greater than that of the " b " variable , then, with " printf ()" from the " standard " output, the escape character that one position back (\ b) is printed.

a == b == c (line 30)

In line " 30" the case is written, where the same number is entered in all three variables " a ", " b ", " c " and entered first, second and third. The use of the ternary condition operator '? :" is justified as follows :
Since, the expression " a == b && b == c && a == c " results in a value " True " (a value other than " 0 "), that is, the variables " a ", " b " and " c " contain the same value, then " printf ()" prints from the " standard " output the contents of " a " (or " b " or " c ") accompanied by the appropriate message and for the row entered. Otherwise, i.e., the expression " a == b && b == c && a == c " produces a result value of " False " (a value equal to "0") and thus, the variables " a ", " b " and " c " do not contain the same value, then the newline escape character (\ n) is printed by " printf ()" from the " standard " output .

EXAMPLES

Example 1 (a > b AND a > c / 1st)

================================== ===============

Find maximum number and row entered

== =

Enter the 1st integer : 8

Enter the 2nd integer : -2

Enter the 3rd integer : 0

---------- ----------

The 1st integer : [8]

The 2nd integer : [-2]

The 3rd integer : [0]

The largest number is the number 8 and was entered 1st.

Example 2 (a < b AND b > c / 2nd)

========== ==========

Find maximum number and row entered

========== =

Enter the 1st integer : 2

Enter the 2nd integer : 5

Enter the 3rd integer : 4

---------- ----------

The 1st integer : [2]

The 2nd integer : [5]

The 3rd integer : [4]

The largest number is number 5 and was entered 2nd.

Example 3 (a < c AND b < c / 3rd)

========== ==========

Find maximum number and row entered

========== =

Enter the 1st integer : -15

Enter the 2nd integer : -13

Enter the 3rd integer : -9

---------- ----------

The 1st integer : [-15]

The 2nd integer : [-13]

The 3rd integer : [-9]

The largest number is the number -9 and was entered 3rd.

Example 4 (a == b > c / 1st , 2nd)

========== ==========

Find maximum number and row entered

========== =

Enter the 1st integer : 32

Enter the 2nd integer : 32

Enter the 3rd integer : 1

---------- ----------

The 1st integer : [32]

The 2nd integer : [32]

The 3rd integer : [1]

The largest number is number 32 and was entered 1st and 2nd.

Example 5 (a < b == c / 2nd , 3rd)

================================ ================

Find maximum number and row entered

== =

Enter the 1st integer : -20

Enter the 2nd integer : -4

Enter the 3rd integer : -4

---------- ----------

The 1st integer : [-20]

The 2nd integer : [-4]

The 3rd integer : [-4]

The largest number is the number -4 and was entered 2nd and 3rd.

Example 6 (x == z > y / 1st , 3rd)

================================ ================

Find maximum number and row entered

== =

Enter the 1st integer : 18

Enter the 2nd integer : 4

Enter the 3rd integer : 18

---------- ----------

The 1st whole number : [18]

The 2nd integer : [4]

The 3rd integer : [18]

The largest number is number 18 and was entered 1st and 3rd.

Example 7 (a == b == c / 1st , 2nd , 3rd)

========== ==========

Find maximum number and row entered

========== =

Enter 1st integer : 100

Enter the 2nd integer : 100

Enter the 3rd integer : 100

---------- ----------

The 1st integer : [100]

The 2nd integer : [100]

The 3rd integer : [100]

The largest number is the number 100 and it was entered 1st, 2nd and 3rd.

REMARKS

In detail, we have seven examples where each corresponds to special conditions contained within the corresponding ternary condition operators '? :» :

Example 1 (a > b AND a > c / 1st) :

Represents the case where the variable " a " contains a number greater than the counterparts of " b " and " c " and was entered first. The results :

a : 8

b : -2

c : 0

The largest number is the number 8 and was entered 1st.

Example 2 (x < y AND y > z / 2nd) :

Represents the case where the variable ' b ' contains a number greater than the corresponding ones of ' a ' and ' c ' and was entered second. The results:

a : 2

b : 5

c : 4

The largest number is the number 5 and was entered 2nd.

Example 3 (a < c AND b < c / 3 rd) :

Represents the case where the variable " c " contains a number greater than the counterparts of " a " and " b " and a third one has been entered. The results:

a : -15

b : -13

c : -9

The largest number is the number -9 and was entered 3rd.

Example 4 (a == b > c / 1st , 2nd) :

Represents the case where the variable ' a ' and ' b ' contain the same number which is greater than the corresponding one of ' c ' and entered first and second. The results :

a : 32

b : 32

c : 1

The largest number is the number 32 and was entered 1st and 2nd.

Example 5 (a < b == c / 2nd , 3rd) :

Represents the case where the variables " b " and " c " contain the same number that is greater than the corresponding number of " a " and a second and third one were entered. The results:

a : -20

b : -20

c : -4

The largest number is the number -4 and was entered 2nd and 3rd..

Example 6 (a == c > b / 1st , 3rd) :

Represents the case where the variables " a " and " c " contain the same number that is greater than the corresponding " b " and entered first and third. The results :

a : 18

b : 4

c : 18

The largest number is the number 18 and was entered 1st and 3rd.

Example 7 (a == b == c / 1st , 2nd , 3rd) :

Represents the case where the variables " a ", " b " and " c "

contain the same and he was introduced first, second and third. The results :

a : 100

b : 100

c : 100

The largest number is the number 100 and 1st, 2nd and 3rd were entered.

Thank you for your attention.