COMPUTER PROGRAMMING

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
UNIVERSITY OF WEST ATTICA

DEPARTMENT OF INFORMATION AND COMPUTER ENGINEERING PAPER

1
INTRODUCTION

STUDENT DETAILS:

NAME: ATHANASIOU VASSILEIOS
STUDENT ID: 19390005
STUDENT STATUS: GRADUATE
PROGRAM OF STUDY: PADA
LABORATORY DEPARTMENT: M2
LABORATORY TEACHER: GEORGIOS MELETIOU
DATE TOTAL COMPLETION OF WORK: 29/10/2021

COMPUTER PROGRAMMING

CONTENTS

COMPUTER PROGRAMMING

ANSWERS TO THEORY SUBJECTS

SUBJECT 1

a) <u>What is a program?</u>

A program is a sequence of commands given by the user, which aims to solve a problem that concerns him.

b) <u>What are the steps to develop and run a " C " program?</u>

The steps to develop and run a " C " program are as follows:

a) Editing :

The user, being aware of any syntactic and logical errors, writes his program with the commands he knows through the environment of the " C " language in order to solve the problem that concerns him.

b) Save the draft program (Program . c):

The user, after completing the writing of his program, saves it in ". c » file (eg Program . c), so that the program will work without further problems.

c) Compilation (Compilation):

C " programming language compiler (compiler) then detects any syntax errors in the user's program based on the language's construction rules, if any. Then, it warns him in detail by underlining the incorrect syntax of the commands in red and stating, in a window below the source code, in which lines he found the typical errors (syntax) and what exactly to correct.

d) Compile the corrected program in machine language (Program . obj):

The user corrects the syntax errors detected by the compiler and shown to him. The compiler then compiles the program into machine language (the language the processor understands) and stores it in a file '. obj » (eg Program . obj) so that the computer processor can process the program data.

e) Linking :

The linker connects the user's object program (Program . obj) to other derivative programs. The specific programs execute commands in which the user gets them ready from the libraries he has used in his program (e.g. the " stdio . h " library used for data input/output commands, such as " printf ()" and " scanf ()"). Finally, it takes the '. obj " file and produces the executable file ". exe » (eg Program . exe).

f) Executable file (Program.exe)

The executable file '. exe ", which belongs to the operating system and is ready to record and process data on the object program (.obj) saved by the compiler.

g) Running :

The program runs and all commands written by the user are executed.

c) <u>What are the general categories of program errors?</u>

The general categories of program errors are as follows:

a) Syntax Error (Syntax Error):

This is an error in the syntax of the program that does not conform to the structure rules of the characteristic programming language. For example, in the " C " language the loss of the terminal Greek question mark "?" after each command is a syntax error detected by the compiler .

b) Logical Error (Logic Error):

It is an error in which the compiler does not detect and refers to the rules of logic that the user himself knows. For example, he wants to calculate the addition of two numbers say '5' and '3' and mistakenly types the command '5 - 3'. The compiler will not detect any error and execute the command normally, but it will not give the desired result that the user is looking for, as it wants to calculate the addition of the two numbers (5 + 3 = 8) and will inadvertently calculate the subtraction of the two of numbers (5 – 3 = 2).

c) Runtime error (Runtime Error):

This is the error that is detected during the execution of the program and it is not possible to correct it, since the commands are executed based on the compiled and saved program. Therefore, in order to correct it, the user should go back to the initial step of developing and running a program (Syntax) and repeat the process (see "Topic 1 b " pages "3-5"). For example, he wants to calculate the division " x / y " of two variables " x " and " y ". Assigns the variable ' y ' which is the divisor to the value '0'. The program will execute normally but the result will obviously not be the desired one, as we know from mathematics that division with a divisor of '0' is not specified.

SOURCE CODES

MARK " C 1 aWrong . c » / « C 1 aCorrect . c »

The program " C 1 aWrong . c " is the template with syntax errors, while the program " C 1 aCorrect . c » is the syntax-corrected program.

PROGRAM 'C1aWrong.c'

```
1 #include < stdio.h >
2
3 int main(int argc , int ** argv )
4 {
5 int A, B;
6 int C, D, E, F;
7 system("chcp 1253"); // Delete if running on Linux
8 printf ("Basic Arithmetic Operations with Integers\n");
9 printf("===============================================\n\n") ;
10 printf ("Enter the first number : ");
11 scanf ("%d",&A); // Error on line 12
12 printf ("Enter the second number : );
13 scanf ("%d", &B);
14 C = A + B;
15 D = A - B;
```

```
16 E = A * B;

17 F = A / B;

18 printf (" Sum : %d\n", c); // Error

19 printf ("Difference : %d\n", D);

20 printf ("Product : %d\n", E); // Error

21 prantf ("Quantity : % d \ n ", F ); // Error

22 return 0??
23 }
```

PROGRAM 'C1aCorrect.c'

```
1 #include < stdio.h >

2

3 int main(int argc , int ** argv )

4 {

5 int A, B;

6 int C, D, E, F;

7 system (" chcp 1253"); // Delete if running on Linux

8 printf ("Basic Arithmetic Operations with Integers\n");

9 printf("===============================================\n\n") ;

10 printf ("Enter the first number : ");

11 scanf ("%d",&A);

12 printf ("Enter the second number : "); // Loss of quote (") at the end
of the function parameter

13 scanf ("%d", &B);

14 C = A + B;

15 D = A - B;

16 E = A * B;

17 F = A / B;

18 printf ("Sum : %d\n", C); // Undeclared variable "c" instead of
declared "C" ( case sensitive )

19 printf ("Difference : %d\n", D);

20 printf ("Product : %d\n", E); // Undeclared Greek "E" variable instead
of declared Latin "E"

21 printf ("Quantity : %d\n", F); // Function "prantf" is not declared
instead of "printf" is declared

22 return 0??
23 }
```

COMPUTER PROGRAMMING

ANSWERS TO SOURCE CODE ISSUES

SUBJECT 2

The syntax errors of the program " C 1 a (Wrong). c » are as follows:

a) Loss of quotation mark (") at the end of the function parameter (line 12):

The parameter of the " printf ()" function is enclosed in double quotes ("…"). In line "12" there is a loss of the quotation mark at the closing of the parameter.

C 1 aWrong . c :

```
12 printf ("Enter the second number : );
```

C 1 aCorrect . c :

```
12 printf ("Enter the second number : ");
```

b) Undeclared variable "c" instead of declared "C" (case sensitive) (line 18):

In line "18" the program calls the function " printf ()" to print the contents of the variable " c ". The compiler does not recognize this variable because it has not been formally declared in the first lines of the code (see lines 5-6). Instead, the variable " C " has been declared and as we know " C " language is " case sensitive " language, that is, the variables " C " and " c " are different. Therefore, there is a syntactical error and in the place of " c " we will put " C ".

C 1 aWrong . c :

```
18 printf ("Sum : % d \ n ", c );
```

C 1 aCorrect . c :

```
18 printf ("Sum : %d\n", C);
```

c) Substitute the undeclared Greek "E" for the declared Latin "E" (line 20):

In line "20" the program calls the function " printf ()" to print the contents of the variable " E ". The compiler does not recognize this variable, because it has not been formally declared in the first lines of the code (see Lines 5-6). In fact, the error presented by the compiler cannot be detected with the "naked eye " . The correct wording of the error is that in the specific program the function " printf ()" in line "20" is called to print the content of the variable "E", where the character "E" is in the Greek version, while in line "6" the corresponding variable " E " has been declared, where the character " E " is in the Latin version. Therefore, there is a syntax error and in line "20" we will make the characteristic change.

C 1 aWrong . c :

```
20 printf ("Product : % d \ n ", E);
```

C 1 aCorrect . c :

```
20 printf ("Product : % d \ n ", E );
```

d) Function "prantf" is not declared instead of "printf" is declared (line 21):

In line '21' the program calls the function ' prantf ()'. The compiler does not detect a syntactic error, but the linker does . In particular, it does not recognize this function from any library imported into the program. The author of the program, in more detail, has introduced the library " stdio . h " which takes the " printf ()" and " scanf ()" data input/output functions ready . So he mistyped the function he wanted to put in, as its purpose would be to print the content of a variable (specifically " F ") and therefore we fix the mistake by replacing " prantf ()" with " printf ()" which leads to the desired result of the editor.

C 1 aWrong . c :

```
21 prantf (" Quotient : % d \ n ", F );
```

C 1 aCorrect . c :

```
21 printf (" Quotient : % d \ n ", F );
```

TOPIC 3

Below are presented three characteristic examples (Example 1, Example 2, Example 3) on the program " C 1 aCorrect ". c » assigning different values to the two variables.

EXAMPLES

Example 1 (A = 10 / B = 5)

Active code page : 1253

Basic Arithmetic Operations with Integers

==

Enter the first number : 10

Enter the second number: 5

Sum: 15

Difference: 5

Product: 50

Quotient : 2

Example 2 (A = 5 / B = 10)

Active code page: 1253

Basic Arithmetic Operations with Integers

==

Enter the first number : 5

Enter the second number: 10

Sum: 15

Difference : -5

Product: 50

Quotient : 0

Example 3 (A = 54645234234 / B = 35453453454)

Active code page : 1253

Basic Arithmetic Operations with Integers

===

Enter the first number: 54645234234

Enter the second number: 35453453454

Sum : -95625528

Difference : 2011911596

Product : -695536596

Quotient: -1

COMMENTS

Example 1 (A = 10 / B = 5)

No interesting observation is observed, as the results in programming and mathematics are expected. More specifically, addition is "10 + 5 = 15", subtraction is "10 – 5 = 5", multiplication is "10 * 5 = 50" and division is "10 / 5 = 2".

Example 2 (A = 5 / B = 10)

Here is an interesting observation that is expected programmatically but not mathematically. More specifically, addition is "5 + 10 = 15", subtraction is "5 – 10 = -5", multiplication is "5 * 10 = 50" and division is "5 / 10 = 0". Mathematically and analytically the division "5 / 10" has a quotient of "0.5". In the program the quotient is "0" because the user has declared the variable to assign the quotient to as an integer (int).

Example 3 (A = 54645234234 / B = 35453453454)

Interesting observations are made here, as the results are unexpected and obviously undesirable. In more detail, the addition is "54645234234 + 35453453454 = -95625528", the subtraction is "54645234234 – 35453453454 = 2011911596", the multiplication is "55645234234 * 35453453454 = -69553". 6596" and the division is "55645234234 / 35453453454 = -1". The explanation behind these strange results is this: in each memory of an " N - bit " processor for each type of data there is a range of values in " bits ". Consequently, these numbers are outside the value range of the " int " type, resulting in problems in entering the data into the memory cells and hence the inaccuracies in the results.

COMPUTER PROGRAMMING

Thank you for your attention.

COMPUTER PROGRAMMING