Tutorial 5 M150

Tutor : Rifat Hamoudi
Staff No. : 00567451
Pager No. : 07669-801 509

I have put this tutorial on the web. This tutorial can be viewed and downloaded from http://www.users.totalise.co.uk/~rifat then selecting M150 Tutorials then Tutorial 5.

1) What is assembly language?

2) Write using assembly language a program that adds two numbers. Using your program describe what is meant by **operator** and **operand**. Also describe what happens to in the computer hardware as the program is executed.

   In this case you can assume that the first number is stored at memory location 400 is 3 and the second number which is stored at 401 is 4. Also assume that the starting memory locaion for your program is 700.

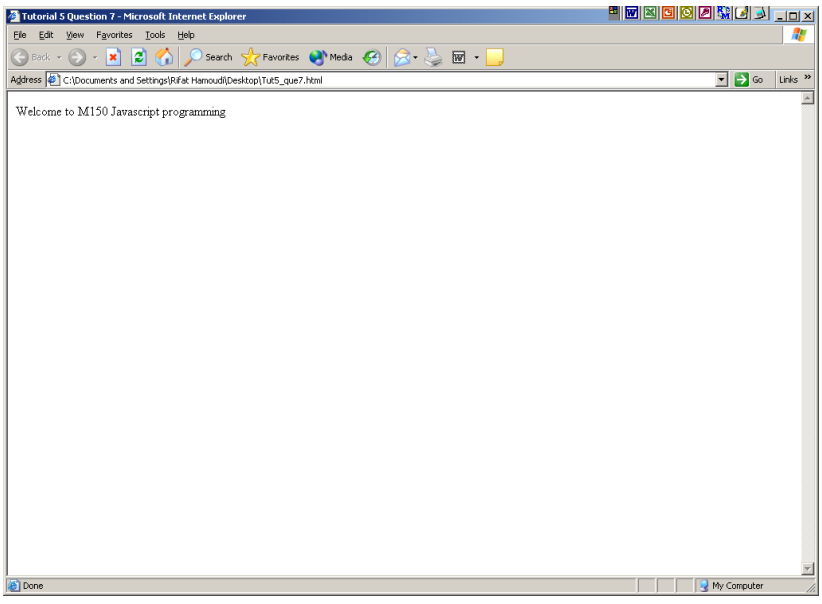
3) Define virtual and cache memory?

4) Describe giving examples the meaning of low and high level languages?

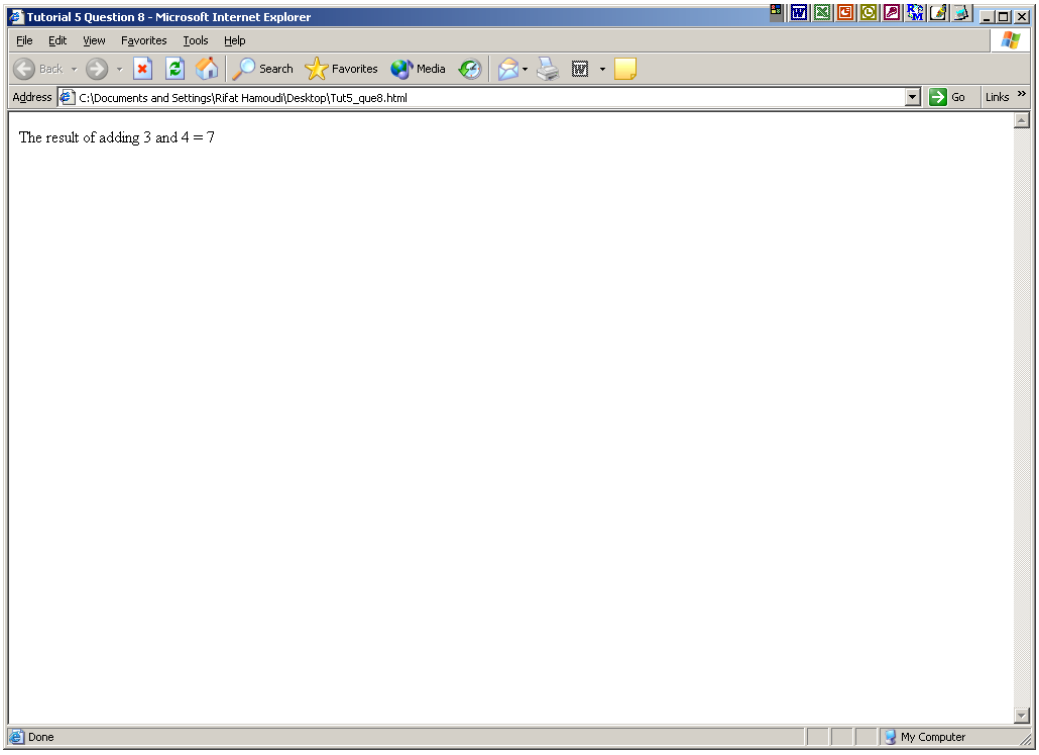
5) What is meant by assembler, compiler and an interpreter?

6) Most of present day software is written using a high level language but what kind of present day software still need to be written using low level languages and what would happen if this software is written using high level language?

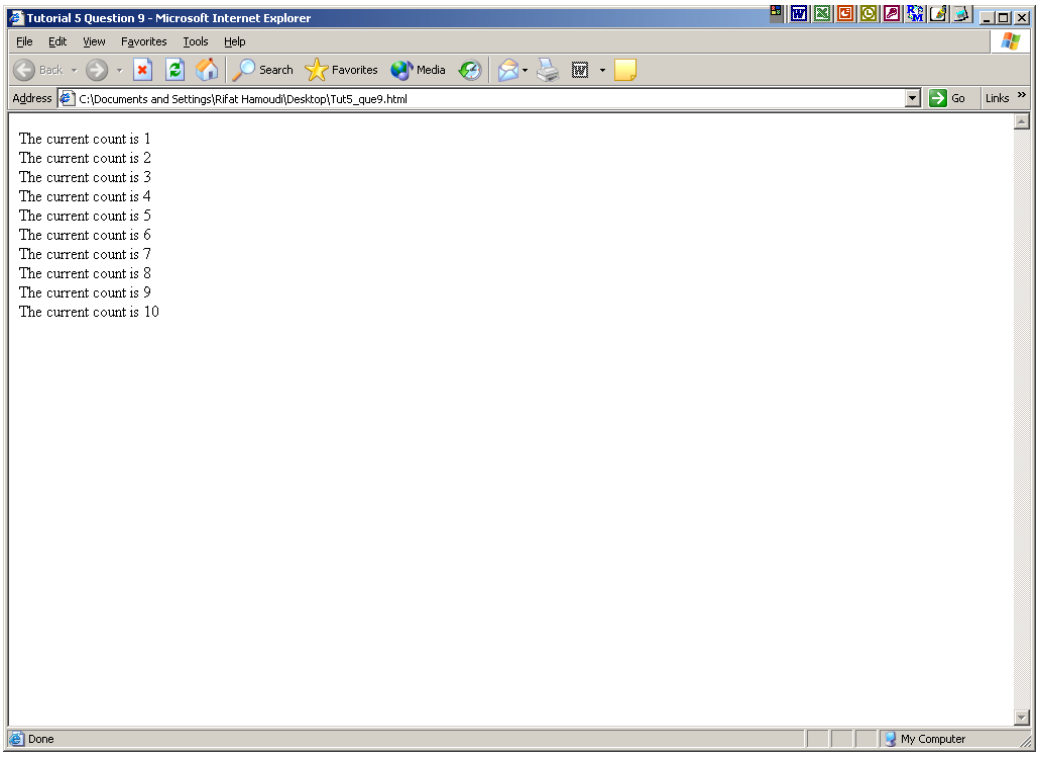
7) Write a program in Javascript to display the sentence "Welcome to M150 Javascript programming" as follows :



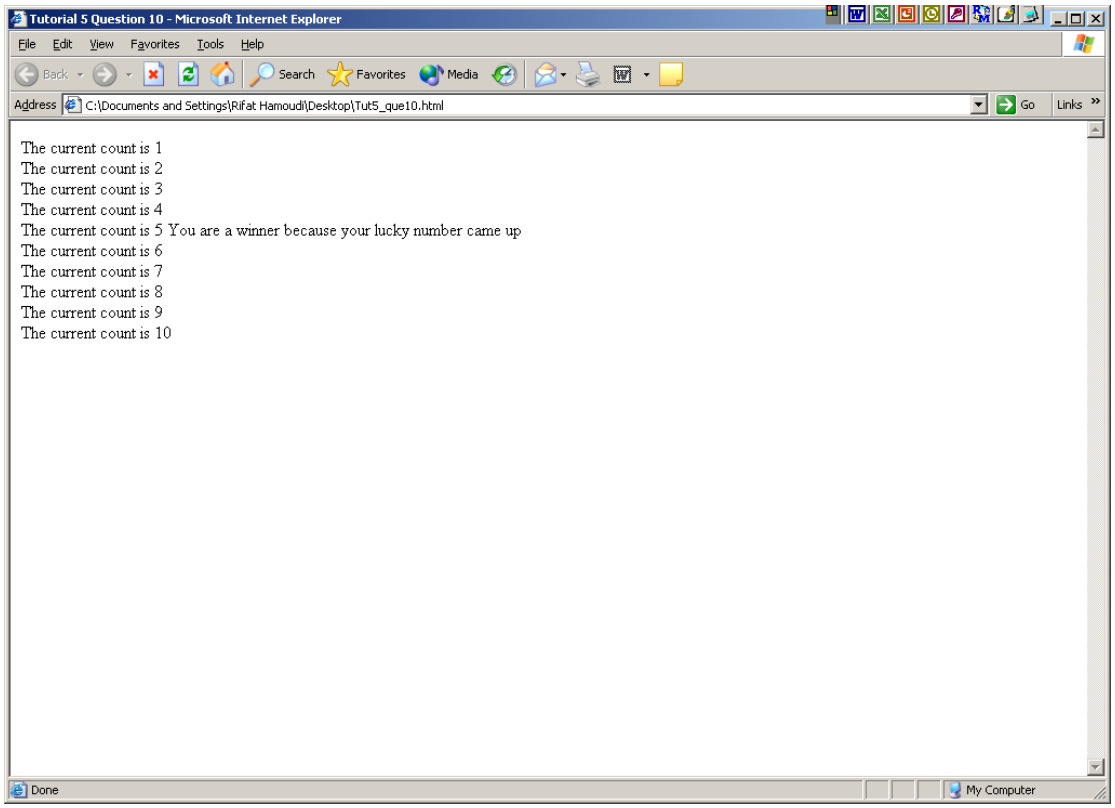
8) Write a program in Javascript that adds two numbers; 3 and 4 and displays the result. Example of output is shown as follows :



9) Write a Javascript program that counts a range of numbers from 1 to 10. Example of output is shown as follows :



10) Modify question 9 so that the program displays the following text "You are a winner because your lucky number came up" next to number 5 (which in this case is suppose to be the lucky number). An example output is as follows :



Answer to question 1

It is a low level language in which each instruction exactly corresponds to one hardware operation of the processor.

Answer to question 2

The assembly language program is as follows :

Memory location	Instruction	Comment
700	LDA 400	Load the content of memory location 400 into the accumulator
701	ADD 401	Add to the accumulator the contents of memory location 400 (which results in the accumulator holding the result of the addition)
702	STO 402	Store the contents of the accumulator in memory location 402
703	HLT 000	End the program

Its location in memory is as follows :

Memory Location	Memory content
400	3
401	4
402	?
. . . .	
700	LDA 400
701	ADD 401
702	STO 402
703	HLT 000

The alphabetic part is called the **operator**, which identifies the instruction (e.g. LDA for load into the accumulator). The operator usually has a name that is mnemonic, i.e. the name gives a clue about what the operator does.

A numerical part, called the **operand**, which specifies a memory location, such as 401. Usually in assembly the operator acts upon the operand.

Running the program

As the program is executed, or run, the effects are as follows:

a) a copy of the data stored in main memory location 400 is placed into the accumulator;

b) a copy of the data stored in main memory location 401 is then added to the contents of the accumulator, so now the accumulator contains the result of the addition;

c) the contents of the accumulator (i.e. the result) are copied to main memory location 402;

d) the program halts.

You can see that before this program is executed, the data needed (the two numbers to be added) must already be stored in memory locations 400 and 401. Once the program has been executed the result will be stored in memory location 402.

Answer to question 3

Virtual memory is when part of storage media such hard disk or optical disk is used as RAM.

Cache memory is intermediate memory that is used to synchronise the speed of RAM with the clock speed of the CPU. Usually the clock speed of the CPU is faster than RAM so this can lead to slowing down of processing of data.

Answer to question 4

As with machine language, an assembly language is a language in which each instruction exactly corresponds to one hardware operation of the processor. Any computer programming language with this one-to-one correspondence (including the machine language itself) is referred to as a low-level language. In programming languages such as JavaScript, Java, C++ and Smalltalk, one instruction may be equivalent to many machine language instructions. These programming languages are referred to as high-level languages. Because of the one-to-one correspondence between a low-level language instruction and a machine language instruction, a program written in assembly language can directly take advantage of all the features and instructions available on the processor. This is not the case with high-level

languages, where a single instruction may be translated into a sequence of low-level instructions in a number of different ways – something over which the programmer has no control.

Answer to question 5

Assembler translates assembly language into machine code. Compiler translates source code of high level language into machine code, some compilers translate source code to assembly language and use the assembler to translate that into machine code. An interpreter translates each instruction in the source code into machine language as the program is running, and then immediately executes it. There is never a complete translation of the whole of the source code into machine language, and so no machine language program is generated or saved.

The advantage of an interpreted language is that the potentially lengthy process of compile/execute does not need to be gone through for each small change in the source code. Interpreted languages lend themselves to situations where small incremental changes to a program are being made, and need to be tested as quickly as possible. The disadvantage is that the translation process must take place every time a program is run, resulting in a slower execution process. Languages, such as JavaScript, Perl and Basic are especially designed to be interpreted, whereas languages such as C, C++ and Java are designed to be compiled.

Answer to question 6

Any safety critical and real time software is best written in low level language like assembly language such as Motorolla 68000 or Zilog or 80486 ..etc as safety critical software need fast manipulation of the variables within CPU registers to keep up with the demand. Example of this is National Air Traffic Control Software, software that track cruise missile and software that monitors vital signs in intensive care unit. For such software real time processing is very important and they are safety critical so if the data is not processed quickly it can lead to two planes crashing in the sky, the cruise missile falling in the wrong place or the patient dying by giving him too much oxygen for example.

Answer to question 7

```
<HTML>
<HEAD>
<TITLE> Tutorial 5 Question 7 </TITLE>
<SCRIPT LANGUAGE = "JavaScript" type = "text/javascript">



/*
 *    Company Database Software
 *
 *    Written by : Rifat Hamoudi
 *
 *    Tutorial 5 Question 7
 */



document.write('Welcome to M150 Javascript programming');


</SCRIPT>
</HEAD>
</HTML>
```

Answer to question 8

```
<HTML>
<HEAD>
<TITLE> Tutorial 5 Question 8 </TITLE>
<SCRIPT LANGUAGE = "JavaScript" type = "text/javascript">

/*
 *    Company Database Software
 *
 *    Written by : Rifat Hamoudi
 *
 *    Tutorial 5 Question 8
 */

var first_number;
var second_number;
var result;

first_number = 3;
second_number = 4;

result = first_number + second_number;

document.write('The result of adding ' + first_number);
document.write(' and ' + second_number);
document.write( ' = ' + result);


</SCRIPT>
</HEAD>
</HTML>
```

Answer to question 9

```
<HTML>
<HEAD>
<TITLE> Tutorial 5 Question 9 </TITLE>
<SCRIPT LANGUAGE = "JavaScript" type = "text/javascript">

/*
 *    Company Database Software
 *
 *    Written by : Rifat Hamoudi
 *
 *    Tutorial 5 Question 9
 */


var first_number;
var last_number;
var index;

first_number = 1;
last_number = 10;

for (index = first_number; index <= last_number; index = index + 1)
{
     document.write('The current count is ' + index);
     document.write('<BR>');
}


</SCRIPT>
</HEAD>
</HTML>
```

Answer to question 10

```
<HTML>
<HEAD>
<TITLE> Tutorial 5 Question 10 </TITLE>
<SCRIPT LANGUAGE = "JavaScript" type = "text/javascript">

/*
 *     Company Database Software
 *
 *     Written by : Rifat Hamoudi
 *
 *     Tutorial 5 Question 10
 */


var first_number;
var last_number;
var index;

first_number = 1;
last_number = 10;

for (index = first_number; index <= last_number; index = index + 1)
{
     document.write('The current count is ' + index);

     if (index == 5)
     {
       document.write(' You are a winner because your lucky number came up');
     }

     document.write('<BR>');
}


</SCRIPT>
</HEAD>
</HTML>
```