QBASIC techniques

For Beginners

11/13/2007

Pravesh Koirala

About the Author

Name:- **Pravesh Koirala**
Currently:- **Studying**
Interests:- **Computer programming, Electronics, Football and Cricket and reading Books**
Country:- **Nepal**
Mail Address:- **pro_science108@yahoo.com**
Language learnt:- **QBASIC, Visual Basic, Visual Basic.NET and C# (learning)**
Programming interests:- **Designing complete applications, Games etc**

Who can be a good Friend

Anyone who is interested in computer and programming in any language can be a good friend.

Views From Author

Hello everybody! Ok now, actually I thought to write this book keeping in mind, the problems faced by the beginner programmers in Qbasic. Qbasic is an excellent tool to develop a programming concept in students. Actually it was developed for that very purpose only. But in spite of its simplicity, students often feel some problems during their studies. I don't know its exact reason but I guess that the course books just focus on the theoretical studies and don't actually develop the concept in students. A programming just doesn't mean to find "LCM, HCF, Factorials, Prime numbers" and other calculations but the real meaning of programming is to directly communicate with the computer and make it perform some work.

I, being really interested in programming, had some explorations on the net and gathered some basic techniques of this programming language. And seeing the complications of the students during the course, I was inspired to write this booklet. Frankly speaking, QBasic is not that hard. But programming can be really a challenge. QBasic just develops a basic concept, it is not that standard language and programming is not only limited to Qbasic. You will find a lot more in the field of computers; this is just a beginning step. And also it depends on you that how much you want to go deeper into the programming concept. Anyways, I hope that you find it useful and handy. If you still face problems after completing this book then I am always in this world. You can just contact me at my mail address i.e. Pro_science108@yahoo.com. Hope you will enjoy it because I have, my best, tried to use simple language and techniques. And yes, being a student myself I can't assure you that this booklet is 100% error free. But I have tried my best to correct all the errors like spellings, grammar and others. What else can you expect from a student? If you could catch any of them and point out the mistakes then you have got all sort of rights to contact me and inform me about those bugs. Also, if you have some other suggestions for me then you are always welcomed. Don't forget to write a feedback.

Last but not the least, the legal stuffs (☹ Boring!). You can distribute this booklet to any of your friends, students or your children (And please don't forget to mention my small name, will you?) but please never try to misuse it for any commercial purpose. I hate these types of guys, you know!

Intended Audience (Who should read this book?)

I have written this book assuming that readers have some little previous knowledge of the programming language (QB). But complete beginners can also take benefits from this book. However those with little previous knowledge can have advantage. The perfect scenario will be that you have already learnt QBASIC but you don't quite know about how to use the functions and you haven't got the programming language well.

This book has nothing related with advanced topics in QBASIC like Modular programming, File Handling, Graphics and Multimedia etc. If you have already learnt the basic techniques then you got this book accidentally and should distribute to one who needs it more.

Credits

Pravesh Koirala	**(Hey I wrote this book!)**
Prasanna Koirala	**(For inspiration and support)**
Family members	**(For their support)**
Someone else	**(Cause he is the real source of inspiration for me. I don't want mention his name)**
Diwaker Jha	**(A fantastic person, nice friend, and the one who lead me deeper into computers)**

More Reading

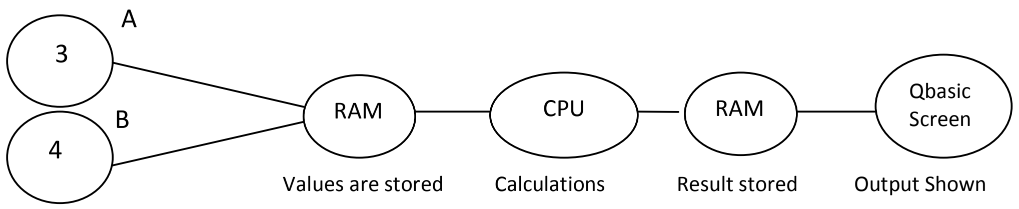
Finished this Book! Please write a feedback to me. Your feedback will prove to be an inspiration to me. If you want some more QBASIC stuffs then you can always search internet. Also, I plan to write next part of this book as "Advanced QBASIC Stuffs" covering concepts like File Handling, Modular programming, Graphics and Sound, Game programming in QBASIC etc etc etc. Let's hope that it will be completed soon. Thanks!

Chapter-1 (Introduction to QB)

Well ready for some QB lessons. Ok now try to learn some fundamentals before actually starting programming concept. Qbasic is a character programming language. (What does it mean?). It means that you cannot implement the windows components like command button and scroll bars that you generally use. Anyways no matter what it is but we have to learn it right? Now you will find that you need to calculate a value like adding, subtracting, divisions etc even for a small program. If you have already programmed a simple application then you ought to know that for multiplication you follow the following steps.

QBasic calculation	Human Calculation
CLS A = 3 B = 4 Print a*b End	3 x 4 (3 times 4 or 4 times 3) i.e. 3+3+3+3=12 or 4+4+4=12 thus, 3 x 4 = 12

Umm, let's think now. What is the difference between this calculation and our human calculations? We have generally mugged up the table of up to 20 even more. Thus it's in our mouth that 3 into 4 equals 12. But the story is different in case of QBasic. First we store 3 in A and 4 in B and we used print a*b. I hope you know the simple functions like Print, CLS and End, If not then wait for Chapter 3 ☺. Anyways let's see what happens in this case. First of all the value 3 is stored in one location of Ram and value 4 in another location. Now it will straightly travel into the processor's ALU (calculation unit). Since, computer is a digital system and it have no techniques like ours, thus, first of all it converts 3 and 4 into binary that is 11 and 4 is 100. Now it will perform binary multiplication (I hope you do know the binary calculations) there and store the result into its memory (No, not RAM, CPU have its own store place called registers). Then it converts the result into decimal and sends the value into a location in Ram (yes, another location!) and then Qbasic access it and prints into the screen. Phew! A long process, isn't it? If you were said to do it then I think it must cost you minimum 50 seconds or more for the longest one. But imagine, the computer does it in a trillionth part of a second, isn't it amazing! Let's have a quick graphical recap.



So next time onwards you will always respect Qbasic, will you? Anyways, since you have learnt the calculating concept, you have rights to ask me that why this is important? For its answer you must wait for the next chapter. If you still don't have a proper idea about this topic then I urge you to read it again and again because this is very important chapter that teaches you a good formula for neat programming.

***Some good practices**

Here are some of the good practices that you can use in your program.

1) Always declare the variables with Dim command before using them. (More details in next lesson)
2) Always try to write remarks in your program using REM function or ' character. Writing remarks makes it easy to understand your program for yourself and other programmers.
3) While writing conditional statement and loops first write their opening and closing command and then insert the statements in between them.
4) Try to select the most suitable loop for your program.

5) If you are writing a lengthy and complicated program then try to break it into small sub procedures.
6) Always select the variable name that is most suitable for the calculation and easy to remember. E.g. If you want store average marks of students in a variable then try to use variable like Avg_Student or Avg_Mark etc.
7) While using Print or Input statement always try to give a proper message to the user otherwise your program couldn't interact with the users well.

***Using Help**
QBASIC is provided with a manual. If in your program anything goes wrong then you are prompted with an error message. If anything like syntax error happens then you can always check out the manual. Just place the cursor below the problem area and press F1 key. That will display the help related to that very command.

***The translator**
You will find that in most of the topics I have mentioned about translator. I think many of you know about it. If you don't then Good! As I mentioned earlier, computer is a digital machine and it knows only 0 and 1. Then the question comes that how does the computer follows our English instructions like Print, End, CLS etc? It's because the English command that you type on the QB editor, is translated to 0 and 1 via either a Compiler or an Interpreter. These Compiler and Interpreter are called translator software.

***Statements**
The statements, in QBASIC, are the operations that give some result. E.G.
a=4, b=b*5, Print "hello" etc

Chapter-2 (Variables and Constants)

Let's recall the simple program that we made in the first lesson.

```
CLS
A=3
B=4
Print a*b
End
```

Now, I think that you do know about the variables and constants but anyways you will have a clear concept about those in this lesson. Lets see "A=3" we already dealt that what it does. It simply stores the value 3 in a location in RAM. Then what is the importance of "A"? Its answer is, the location in which the value 3 will be stored in the RAM will be represented by the keyword "A" so that when you want to retrieve the value of A from Qbasic then the system will search all the values stored and used by the program. Then it will have three locations A, B and another one (I know that you are amazed but yes there is another location with a value. What is it can you find? If you can't then don't worry, I will tell you about it in the last part of this chapter). Within the three locations the system will match the keyword that is A. Now when it confirms the value then it transfers it into the QB and QB displays it into the screen.

Phew! Again a long process just to recall a simple value, isn't it? Now think for a while, why in the world is this process important and being mentioned? Imagine that you will try to retrieve the value of one variable and then you get the value of another. This can really be dangerous and it is simply not good. Well we are lucky that the computer does it all otherwise it would have been a tedious task to manage all the locations and values.

And can you answer that why the hell "*" is used instead of "x" for multiplication. Answer is simple, it is because you might want to have a variable "x" and this will create a lot of confusion. Well the developers of the QB are really smart, aren't they? While determining a variable following things should be considered.

1) Variable can have alphabet from A-Z and digits from 0-9 and underscore "_"
2) Variables first character must be non numeric. E.G- pro10 (not 10pro)

A variable has its two characteristics, its type and data hold by it.

***TYPES OF THE VARIABLE**

Consider,

A=3 and A=0.003

In both the cases, there is used a same variable A but there is a lot of difference between their types A=3 represents that A is a variable used for storing an integer type value but on the contrary, A=0.003 represents that A is used for storing a decimal or floating type value. And yes, each variable has its own type that must be declared. The declarations can be carried out by the Function DIM.

E.g.:-

```
Dim a as Integer
Dim b as Long
Dim c as String
Dim a (5) as integer
```

Note the point that a, b, c are the variables and the keywords "Integer", "Long", "String" are their types. The characteristics of the types can be found in any course book. I suggest you to go through it. In summary, each variable has its own memory requirement.

Now the big question comes that though we haven't declared the variables A and B in our previous program, how does the program handles the memory for it? And the small answer is; all the non declared variables are thought to be a numeric variant by the translator. But what is a variant? Variant is also a type but it consumes comparatively a large space in the memory. So, it's not a good practice to use them more often. However, in small programs they can be used. EG:-

***Non declared or implicit variables**

E.g.:-

```
A=3
Print a
A=0.03
Print a
End
```

Output:-

```
3
0.03
```

***Declared or explicit variables**

```
Dim a as Integer
10: A=3
20: Print a
30: A=2.03
40: Print a
50: End
```

Output:-

```
3
2
```

What! A 2! But hey, I said that I want to print a 2.03. Maybe the translator has gone mad! No, it is not the translator. As soon as you run your program, the translator translates the program to machine codes from top to bottom. The translator is instructed that the variable A is an integer and so, it occupies memory space for integer. Up to step 20 it runs smoothly but as soon as it encounters the next step i.e. (A=2.03) then it thinks for a while, “Umm, let’s see what we have, A=2.03. What? Oh god! What’s the programmer trying to do? First he says reserve the space for integer and then wants to keep a floating value 2.03. He must have gone mad! Anyways better I store only the integer 2.”

And so the translator doesn’t store any floating value and variable A becomes 2. So Einstein, from next time onwards never try to store a wrong value in a wrong type. It can really annoy the translator and moreover can effect a mathematical calculation.

***String Type Value**

A string type value resembles a character sequence of up to 32,767 characters and always kept inside the double quotation in not only QB but in other higher programming language too. Strings are handled differently by the memory (RAM). So, if you try to store string in a numeric variable and vice versa then system will feel really ☹. Let’s have an example.

Example-1	Example-2
CLS	CLS
A = “HELLO”	Dim A as String
Print A	A= “HELLO”
End	Print A
	End

What do you think what is the output of Example-1? If you thought about an error then you got it! Certainly, there will be an error i.e. “Type Mismatch”. Well, previously the translator did correct our error because it was related to numerical type mismatch but as I told, the string is completely different from the number in relation to its storing process. Do you know something about ASCII? You must know! There is a standard code for each of character. So when you store some string values then RAM will not actually store the character but its ASCII instead.

But in Example-2 we have already instructed the translator that we need a space for string value represented by the variable A. So there will be no any complications.

***Signs used for types**
Imagine that you are designing a small program and you don't want to spend time writing declarations for variables (However writing declarations is a good practice ☺.) Then you have the options to use some special signs that will tell the translator about the types of the variable. E.g.:-

```
CLS
A$= "Pravesh"    'Instead of declaring A as string you can directly use signs
Print A$         'But remember you must always assign the sign in the program with that variable
End
```

*Following are some of the signs that can be used for different var(variable) types

i) $	String
ii) %	Integer
iii) !	Single Floating Point
iv) #	Double Floating Point
v) &	Long Integer

***Declaring a default Type**
Now think for a while. You want to work only with one type i.e. string. I mean that you want to use only string variables in your program. Then there are three ways.
i) Declare all variables that you want to use as String
ii)Use sign $ for each variable
iii)Declare the default type as String

I am sure that out of these steps you are familiar with the first two. So, lets learn how to declare a default variable. By default, in Qbasic, each variable is a numeric variant unless declared using Dim statement or using signs. But we can declare any other type as default according to our wish using the DEF-Type keyword. E.g.:- DEFStr A-Z
Lets break the statement. DEF + Str + A-Z are our results. Clearly, DEF is a function used to declare the default type. Str is acronym for String and A-Z is the range. What this function does is, it declares all the variables from A to Z as **String** unless separately declared using Dim or any sign. Let's have a short example.

```
CLS
DEFStr A-W              'Now lets declare all variables from a to w as string
Dim C as integer        'Lets separately declare C as integer
V= "Hello World!"       'Put "Hello World" in variable V
C = 8                   'Put 8 in C
D! = 9.09               'Store 9.09 in D!
Print V , C ,D%         'Print all vars
End
```

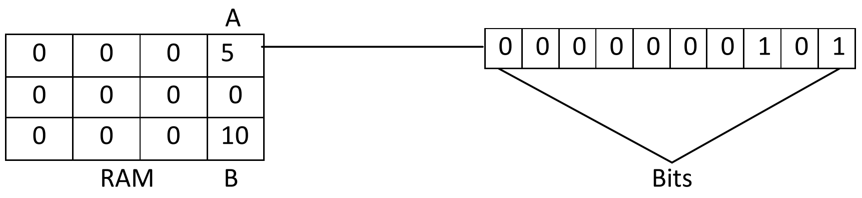
Understood? What! No? Ok ok don't panic, Lets deal it in detail. We declared all values from a to w as string. So, we need not use $ sign with the variables now onwards. But wait, we also declared c separately as an integer. And why we have used D! while storing 9.09 in it? So confusing right? Answer is, we are storing an integer value in c i.e. 8. But since the default type is string, thus, Qbasic would have treated it as a string and you know that 8 is not a string so there would have an error. For D we have stored 9.09 and again the QB would have tried to store it as a string but since we provided it with the symbol ! so the translator will now store it as a single precision. Clear as mud, right? ☺

***CONSTANTS**

Since you are now clear with the variables then lets head towards Constants. You already know that variable is a location in memory that stores a data. And Constant is that very data which is being stored. E.G.

```
A% = 5
B% = 10
```

With this simple command, the following things happen inside the memory.



I think that since you began to learn computer, you were taught about 1 byte is equivalent to 8 bits. If you were not quite clear about that then the figure illustrates it all. Bit is the smallest part of memory. And the group of 8 bits make 1 Byte.

Every thing is alright except that all other locations(or bytes) are 0. Do you know that the memory stores the data in form of electric charges?

Now when you store 5 in A then translator occupies a memory location and system charges the memory electrically and 5 is stored in form of electricity. But consider this small program.

```
CLS
Dim C as integer
Print C
End
```

What do you think about the output? Certainly, there will be a zero. You did declare space for C but you forget to store any value. It means that the memory won't be charged and as a result you will get a big 0. But in other programming language like C++, the default value of a variable is undefined.

***Updating the value of constants during program execution**

In QBASIC the value of constants can be updated during run time. Well, if it can be changed then why is it named Constant? Strange, right?

Anyways, updating the value makes it possible to carry out different calculations. E.G

```
CLS
A=5
A = A+2
Print A
End
```

Output

```
7
```

I think that I need not mention what happened here.

***Updating Variable with loops**

For updating variable repeatedly, we must use it with loops. Loops are described in forthcoming chapters.

***What was that another location mentioned earlier?**
Well I mentioned about three locations at the start of the chapter but there were only two viz. (namely) A and B. Then what am I gone mad? No certainly not, if I were mad then how could have I wrote this book. Anyways, So what was that third location? The answer is, after the calculation of 3*4 the result i.e. 12 is also stored inside the RAM. So, it is also a location, isn't it?

So, we came to the end of this chapter. See you in next chapter. Oh! I was about to leave without giving you an information. See, how dumb I am! The thing that I wanted to inform you was that I didn't mention some more topics about the variable types like User-defined types. Remember what I said earlier, it depends upon you that how deeper you want to go inside the programming concept. So, if you want to explore more then just check out some other books and explore. Otherwise, leave it!