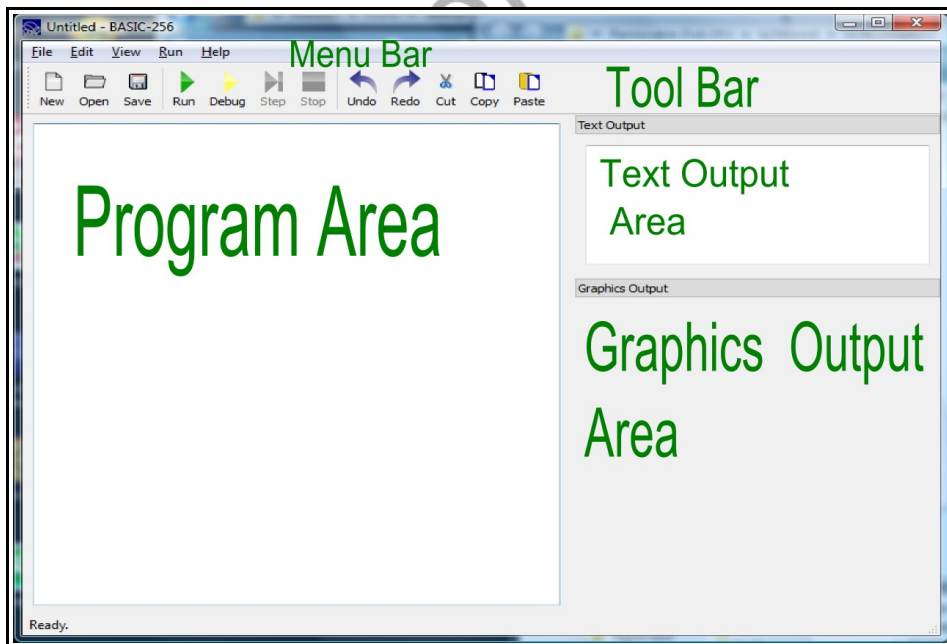


# Chapter 1: Meeting BASIC-256 – Say Hello.

This chapter will introduce the BASIC-256 environment using the **print** and **say** statements. You will see the difference between commands you send to the computer, strings of text, and numbers that will be used by the program. We will also explore simple mathematics to show off just how talented your computer is. Lastly you will learn what a syntax-error is and how to fix them.

## The BASIC-256 Window:

The BASIC-256 window also known as an Integrated Development Environment (IDE) is divided into five sections: the Menu Bar, Tool Bar, Program Area, Text Output Area, and Graphics Output Area (see Illustration 1: The BASIC-256 Integrated Development Environment (IDE) below).













*Illustration 1: The BASIC-256 Integrated Development Environment (IDE)*




## Menu Bar:

The menu bar contains several different drop down menus. These menus include: "File", "Edit", "View", "Run", and "About". The "File" menu allows you to save, reload saved programs, print and exit. The "Edit" menu allows you to cut, copy and paste text and images from the program, text output, and graphics output areas. The "View" menu will allow you to show or hide various parts of the BASIC-256 window. The "Run" menu will allow you to execute and debug your programs. The "About" menu option will display a pop-up dialog with information about BASIC-256 and the version you are using.

## Tool Bar:

The menu options that you will use the most are also available on the tool bar.

-  New – Start a new program
-  Open – Open a saved program
-  Save – Save the current program to the computer's hard disk drive or your USB pen drive
-  Run – Execute the currently displayed program
-  Debug – Start executing program one line at a time
-  Step – When debugging – go to next line
-  Run to Break Point – When debugging – run to the next line marked as a break point
-  Stop – Quit executing the current program
-  Undo – Undo last change to the program.
-  Redo – Redo last change that was undone.

-  Cut – Move highlighted program text to the clipboard
-  Copy – Place a copy of the highlighted program text on the clipboard
-  Paste – Insert text from the clipboard into program at current insertion point

### Program Area:

Programs are made up of instructions to tell the computer exactly what to do and how to do it. You will type your programs, modify and fix your code, and load saved programs into this area of the screen.

### Text Output Area:

This area will display the output of your programs. This may include words and numbers. If the program needs to ask you a question, the question (and what you type) will be displayed here.

### Graphics Output Area:


BASIC-256 is a graphical language (as you will see). Pictures, shapes, and graphics you will create will be displayed here.

## Your first program – The *say* statement:


Let's actually write a computer program. Let us see if BASIC-256 will say hello to us. In the Program Area type the following one-line program (you will see the line number in BASIC256 but you should not type it):

```
1 say "hello"
```


*Program 1: Say Hello*


Once you have this program typed in, use the mouse, and click on "Run" in the tool bar. 


Did BASIC-256 say hello to you through the computer's speakers?

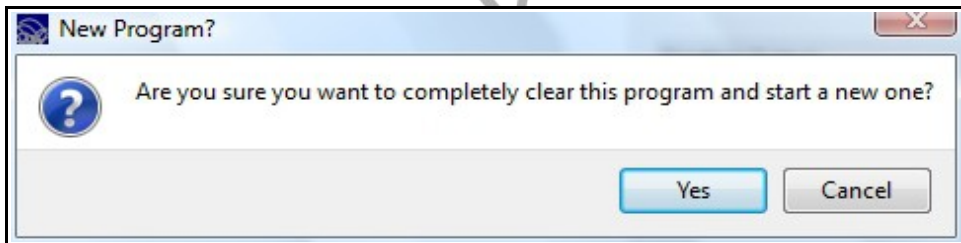
 <p><b>New Concept</b></p>	<p><code>say expression</code></p> <p>The <b>say</b> statement is used to make BASIC-256 read an expression aloud, to the computer's speakers.</p>
---	--

BASIC-256 treats letters, numbers, and punctuation that are inside a set of quotation marks as a block. This block is called a string.

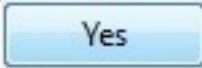
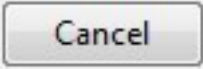
 <p><b>New Concept</b></p>	<p>"letters, numbers 9988, and symbols &amp;%" 'another string with a "quote" inside.'</p> <p>A string may begin with either a single quote mark (') or a double quote mark (") and ends the same as it began. A string surrounded with single quotes may contain double quotes and a string surrounded by double quotes may contain single quotes.</p>
--	---


 <b>New Concept</b>	<p>▶ "Run" on the tool bar - or - "<u>R</u>un" then "<u>R</u>un" on the menu</p> <p>You must tell BASIC-256 when you want it to start executing a program. It doesn't automatically know when you are done typing your programming code in. You do this by clicking on the ▶ "Run" icon on the tool bar or by clicking on "<u>R</u>un" from the menu bar then selecting "<u>R</u>un" from the drop down menu.</p>
---	---

To clear out the program you are working on and completely start a new program we use the  "New" button on the tool bar. The new button will display the following dialog box:



*Illustration 2: BASIC-256 - New Dialog*

If you are fine with clearing your program from the screen then click on the  "Yes" button. If you accidentally hit "New" and do not want to start a new program then click on the  "Cancel" button.


 <p><b>New Concept</b></p>	<p>"New" on the tool bar - or - "File" then "New" on the menu</p> <p>The "New" command tells BASIC-256 that you want to clear the current statements from the program area and start a totally new program. If you have not saved your program to the computer (Chapter 2) then you will lose all changes you have made to the program.</p>
---	---

## Your Second Program – Saying Something Else


You can also have the **say** statement speak out numbers. Try the following program:

```
1 say 123456789
```

*Program 2: Say a Number*

Once you have this program typed in, use the mouse, and click on "Run" in the tool bar. 

Did BASIC-256 say what you were expecting?

 <p><b>New Concept</b></p>	<p><i>numbers</i></p> <p>BASIC-256 allows you to enter numbers in decimal format. Do not use commas when you are entering large numbers. If you need a number less than zero just place the negative sign before the number.</p> <p>Examples include: 1.56, 23456, -6.45 and .5</p>
---	---

## BASIC-256 is really good with numbers – Simple Arithmetic:

The brain of the computer (called the Central Processing Unit or CPU for short) works exclusively with numbers. Everything it does from graphics, sound, and all the rest is done by manipulating numbers.

The four basic operations of addition, subtraction, multiplication, and division are carried out using the operators show in Table 1.

Operator	Operation	Example
	+ Addition	expression1 + expression2
	- Subtraction	expression1 - expression2
	* Multiplication	expression1 * expression2
	/ Division	expression1 / expression2

*Table 1: Basic Mathematical Operators*

Try this program and listen to the talking super calculator.

```
1 say 12 * (2 + 10)
```


*Program 3: Say the Answer*

The computer should have said "144" to you.

1 say 5 / 2

Program 4: Say another Answer

Did the computer say "2.5"?

	<p>+ - * / ( )</p>
<p><b>New Concept</b></p>	<p>The four basic mathematical operations: addition (+), subtraction (-), division (/), and multiplication(*) work with numbers to perform calculations. A numeric value is required on both sides of these operators. You may also use parenthesis to group operations together.</p> <p>Examples include: <math>1 + 1</math>, <math>5 * 7</math>, <math>3.14 * 6 + 2</math>, <math>(1 + 2) * 3</math> and <math>5 - 5</math></p>

## Concatenation:

Concatenation is the operation that joins two strings together to make a longer string. If the strings "abcd" and "xyz" and concatenated together the string "abcdxyz" would be the result. This operation is called concatenation, or "cat" for short.

BASIC-256 has three different operators that will concatenate strings, but they perform differently when the expressions are numbers. The `;` operator will convert expressions to strings and always concatenate, the `+` operator will numerically add two numbers but concatenate if either are strings, and the `&` operator will perform a 'bit-wise and' if both are numbers but will otherwise concatenate.



Let's try it out:

```
1 say "Hello " ; "Mary."
```

*Program 5: Say Hello to Mary*

The computer should have said hello to Mary.

Try another.


```
1 say 1 ; " more time"
```

*Program 6: Say it One More Time*

In the last example concatenation was performed with a number and a string. The number was first converted to a string "1" and then BASIC-256 was able to concatenate.

```
1 Say 1 + 2
2 say '1' + 2
3 say 1 ; 2
```

The computer should have said "three", "twelve", and "twelve". In the first line, the plus operator adds the numbers 1 and two. In line 2, the plus operator concatenates the string 1 to the string 2 (the number is converted). In the last line the semicolon operator converted both numbers to strings and concatenates.


 <p><b>New Concept</b></p>	<pre data-bbox="339 186 618 287">; (concatenate) + (concatenate) &amp; (concatenate)</pre> <p data-bbox="339 336 1243 442">The semicolon (;) is used to tell the computer to concatenate (join) strings together. If one or both operands are numeric they will be changed to strings before concatenation.</p> <p data-bbox="339 486 1222 592">The + and &amp; operators perform concatenation if either or both expressions are strings. If both are numbers then they perform other actions.</p>
---	---


## The text output area - The print statement:

Programs that use the Text to Speech (TTS) **say** statement can be very useful and fun but it is also often necessary to write information (strings and numbers) to the screen so that the output can be read. The **print** statement does just that. In the Program Area type the following two-line program:

```
1 print "hello"
2 print "there"
```

*Program 7: Print Hello There*


Once you have this program typed in, use the mouse, and click on  "Run" in the tool bar. The text output area should now show "hello" on the first line and "there" on the second line.

 <p><b>New Concept</b></p>	<pre>print <i>expression</i> print <i>expression</i>;</pre> <p>The <b>print</b> statement is used to display text and numbers on the text output area of the BASIC-256 window.</p> <p>The print statement, by default, advances the text area so that the next print is on the next line. If you place a ; (semicolon) on the end of the expression being printed, it will suppress the line advance so that the next print will be on the same line.</p>
---	---

The **print** statement, by default, advances the text area so that the next **print** is on the next line. If you place a ; (semicolon) on the end of the *expression* being printed, it will suppress the line advance so that the next **print** will be on the same line.

```
1  cls
2  print "Hello ";
3  print "there, ";
4  print "my friend."
```

Program 8: Many Prints One Line


 <p><b>New Concept</b></p>	<pre>cls</pre> <p>The <i>cls</i> statement clears all the old displayed information from the text output area.</p>
---	--


## What is a "Syntax error":

Programmers are human and occasionally make mistakes. "Syntax errors" are one of the types of errors that we may encounter. A "Syntax error" is generated by BASIC-256 when it does not understand the program you have typed in. Usually syntax errors are caused by misspellings, missing commas, incorrect spaces, unclosed quotations, or unbalanced parenthesis. BASIC-256 will tell you what line your error is on and will even attempt to tell you where on the line the error is.

Free eBook Edition

**Exercises:**

 <p><b>Word Search</b></p>	<pre> z a h d g p b a n n q m c j g j r o i q l o c q o x r u n t u u n i l c n s z v w s y o b s s k c y l l e n a t i s s p a n p a x r s e p e q r t t f r p t r b k r y o e a r m m r a o r p i g n x d o i f n i r x n r a y t i h l n a f e g a t m d w n v e d g i t m i a c v c e i j f d n b o t c c a u s o r c i s n a m z i z i g n c p r u </pre> <p>cls, concatenation, error, expression, print, program, quote, run, say, stop, string, syntax</p>
---	--

 <p><b>Problems</b></p>	<ol style="list-style-type: none"> <li>1. Write a one line program to say the tongue twister 'Peter Piper picked a peck of pickled peppers.'</li> <li>2. Add a second line to Problem 1 to also display that sentence on the screen.</li> <li>3. Use the computer as a talking calculator to solve the following problem and to say the answer: Bob has 5 pieces of candy and Jim has 9. If they were to share the candy evenly between them, how many would they each have (average).</li> <li>4. Use the computer as a talking calculator to solve the following problem and to say the answer: You want 5 model cars that each cost \$1.25 and one model boat that costs \$3.50. How much</li> </ol>
--	---

	<p>money to you need to make these purchases.</p>
--	---

	<p>5. Write a one line program to say “one plus two equals three” without using the word three or the number 3.</p>
--	---

Free eBook Edition