KidBasic®

Computer programming for grown ups and kids!

Copyright © 2003-2005, KidBasic Corporation
KidBasic™, Dr. Genius teaches KidBasic, computer programming for kids, so easy even adults can use it!®, Dr. Genius™, KidKoder™, are trademarks or registered trademarks of KidBasic Corporation.
Patents Pending.

KidBasic Corporation
14625 Baltimore Ave, suite 116
Laurel MD 20707

YEAR & COPYRIGHT:
2003-2005; All Rights Reserved, KidBasic Corporation

Forward

Thank you for purchasing KidBasic. We hope you will have a wonderful learning adventure as you use our product.

We are dedicated to creating products that enrich the minds and hearts of people worldwide. Please, look out for more entertaining and educational software from us.

We sincerely hope you will enjoy our product, and would love to hear from you. Visit us at *www.KidBasic.com or www.KidBasic.net*

-Dr. Genius & the folks at KidBasic Corporation

Content

Using this book ..... Page 4

Let's Begin! ..... Page 5

KidBasic Features ..... Page 7

My 1st computer program! ..... Page 9

Make your computer talk! ..... Page 13

Picture and Sounds ..... Page 14

Fun with buttons ..... Page 16

Loops ..... Page 19

Do you compute? ..... Page 21

If, then, end if ..... Page 23

What's in your jar? ..... Page 26

Guess Game ..... Page 32

Cool Projects ..... Page 34

KidBasic Reference ..... Page 39

Using this Book

Icon key

Hint
Term
Quick Quiz

Dr. Genius will guide you through this KidBasic programming adventure.

When you have completed this fun-filled adventure, visit *www.KidBasic.net* with a parent, to register for your official KidBasic Programmer Certificate! You will need a printer to print out your certificate.

You can also submit some of your programs so other KidBasic programmers and end-users can try them out on!

Let's Begin!

Let's Begin!

Welcome to the KidBasic® Adventure

My name is Dr. Genius, and it is my super duper pleasure to teach you KidBasic, a computer programming language that rocks !

You'll use KidBasic to create your own computer programs. KidBasic computer programs are fun for everybody to play with, but what *is* a computer program anyway?

A computer program is one or more instructions that a computer can understand and obey.

I bet you didn't know that a computer cannot think by itself. A computer needs a computer program to tell it how to think or behave. It's a fact, all computers cannot perform any actions unless someone creates a computer program for those actions.

You may have used computer programs many times before, but never had to create one. Have you made your computer save or print work that you created? Believe it or not, you used a computer program to do that! Since you didn't need to create the programs to do any of these things, where did the programs come from?

There are lots of computer programs already on your computer. The one that shuts down your computer when you click on the SHUT DOWN button, is called an operating system. An operating system acts as the brain of your computer, because it controls the way your computer thinks or behaves. An operating system is a very big and difficult computer program to create. Examples of operating systems are: Windows 98, ME, XP, NT or Windows 2000. Computer programmers in a company called Microsoft created all these programs.

Another type of computer program is called an application.
An application is a smaller program than an operating system. Examples of applications are Microsoft Word, and Microsoft Paint.

Other application examples include computer games, such as Minesweeper, Solitaire, and FreeCell.

Since computer programs are instructions a computer can follow, computer programs must contain good instructions, so that computers can understand and carryout the instructions properly. Let's pretend that I want to give you instructions on how to bake cookies. If I give you bad instructions you will bake hard, or even burnt cookies. If I want you to bake tasty, soft cookies, I have to give you good instructions.

Term

A computer programmer is a person that creates computer programs. He or she gives computers instructions to follow. A KidBasic programmer, is someone that knows how to create a computer program with the KidBasic programming language.

A computer program user is anyone that uses or plays with these programs.

Soon, you too will give your computer its first set of instructions by typing a simple computer program into the KidKoder. This will make your computer obey your commands! But first, lets learn about the KidKoder.

KidKoder Features

KidKoder Features

The KidKoder is where you will type all your KidBasic computer programs.

The left part of the KidKoder is called the editor. This is where you will type all your KidBasic computer programs. The right part of the KidKoder is the results screen. This is where you will see the results of your programs once you activate them.

There are six buttons under the results screen in the KidKoder.

The **open button**, the **save button**, the **help button**, the **activate button,** the **clear button,** and the **exit button**. Here's what they do.

The **activate button,** tells the computer to perform the instructions of the program that has been typed inside the editor.

If you make any changes to the instructions in your program, you will need to press the activate button again. This allows the computer to read the changes you made to the instructions in your program.

The **save button,** is used to save the program you typed inside the editor. You decide what name to give your program and then save it

using that name. Try to use names that will help you remember what the computer program instructs the computer to do.

Use the **open button**, to open a program you have already saved. When you open the program, the instructions will appear in the editor. Press the **activate button** to make the computer activate the program.

The **help button**, gives tips on what could be wrong with your program.

If you press the **activate button** and your program does not work the way you think it should, press the **help button**. This will bring up a help window that will tell you what could be wrong with your program.

The **clear button**, clears up your results screen. Sometimes you may want to clear up everything in the results screen, just click on the **clear button** to do this.

Press the **exit button,** to exit KidBasic.

Hint

If you leave your mouse pointer over a button, its name will be displayed.

To start up the KidKoder, go to your computer's start menu, point to programs, point to KidBasic, and then click on KidBasic.

My First Computer Program

My First Computer Program

Type the following computer program into the KidBasic editor:

```
start
write  "Hello KidBasic World!"
stop
```

The color of the ***start***, ***write***, and ***stop*** commands will change to blue, when typed into the editor. All KidBasic commands change to a light blue color when typed in the editor. In fact, if you type a KidBasic command in the editor and its color doesn't change, it means you typed the command incorrectly.

🗁 Hint

If a KidBasic command doesn't change to a light blue color, check for its correct spelling.

Press the **save button**, and the save as box will appear.

Notice that your programs are automatically stored in the KidBasic Programs folder. In the "File Name" area, type **myprogram**. This is the name we will give to this computer program. Then click **save**.

Once the program is saved, press the **activate button**. The computer should show the result below in the results screen.

```
Hello KidBasic World
```

If there are mistakes in your program, the computer will not write "Hello KidBasic World" on the screen. In fact, it might not display anything. Look over your program to see if you typed something wrong or if you forgot to type anything. If you have no mistakes in your program, the computer will activate all your instructions. If you need help press the help button.

We know that computer programs are instructions that tell computers what to do, and computers obey those instructions by doing something. But what language does a computer understand?

Just like people speak English, Spanish, or French, computers have a lot of different languages they speak too. The language we just used and will be using for this adventure is called KidBasic. Each language has its own set of commands. Commands are action words that the computer recognizes.

In English, we understand the words "come, go, sit, stand" and many others, as little words that tell us to perform a specific action. A computer also understands commands like "start, stop, write" and many more that you will see later. Now let's go over the commands in the simple program you just tested.

The ***start*** command tells the computer that you are going to start giving it some instructions.

The ***write*** command tells the computer to write something in the results screen. It writes anything and everything within the double quotes (*""*).

The *""* (double quotes) is the symbol used for conversation in the English language. For example, Dr. G said "You're doing great so far!"

In the KidBasic computer language, the double quotes symbol specifies what the computer should write onto the results screen.

On a Standard English (US) keyboard, double quotes are found next to the "enter" key. (Hold down the shift key while you press the double quotes key to get double quotes.)

The ***stop*** command tells the computer that you are done giving it instructions to follow.

Well, congratulations on coding your very first KidBasic computer program! That was a really easy computer program. Hold on tight, you'll be writing much cooler programs in no time.

Whenever you need help remembering a command, go to the KidBasic Reference Table, which can be found at the back of this manual. It can also help you correct mistakes in your programs faster.

Now, let's try another computer program. Type the following program into the KidBasic editor:

```
start
title “Welcome to my computer
program”
background blue
write “Yipee I am a programmer”
stop
```

Now press the **save button**. You can name this program Yipee.

🗁 Hint

The ***stop*** and ***start*** commands must be at the beginning and end of every KidBasic program. If not, the program will not respond when you click on the **activate button**.

Always remember to save your program after typing it into the KidBasic editor.

Now, click on the **activate button**.

This program introduces two new commands, ***title*** and ***background***.

The ***title*** command tells the computer to write the text in the double quotes as a title. That means whatever is within the double quotes of this command will be shown as the title of the program.

The ***background*** command allows you to change the background color of the results screen. Isn’t that cool? Try other colors using the ***background*** command.

Let’s try another simple program. Type the following computer program and test it.

```
start
title " Welcome to Vikky's computer
program"
write " Be nice to everyone.”
nextline
write " It is good to share!"
nextline
write “Be nice to your friends!”
nextline
stop
```

From the results screen, what do you think the ***nextline*** command does? If you're not sure, remove the ***nextline*** command from the program you typed into the editor. Then click on **activate**.

What did you notice? Without the ***nextline*** command, the computer will try to show the results of everything you type on the same line. The ***nextline*** command tells the computer to move to the next line. Kind of like the "enter" or "return" key on your keyboard.

You can save this program with any name you wish. This program uses some of the commands we have seen before. Go ahead and try out these commands again and again, then move on to the Quick Quiz.

Quick Quiz

A. Use the ***title*** command to make this title appear on the screen:
"Welcome to my new show"
B. Use 2 ***write*** commands to write the following on the screen:
"Don't go anywhere with strangers!"
"KidBasic is so cool!"

Hint

Don't forget to use the ***nextline*** command

If you're done, you can move on to learn about the ***say*** command. But first, why don't you invite your friends or someone from your family to test and play with your computer programs. They can be your first end-users!

Make Your Computer Talk

Make Your Computer Talk

Use the ***say*** command to make the computer say words, phrases, or sentences through the speakers connected to it.

Here's an example, type the program below into the editor and press the **activate button**.

```
start
say "hello"
stop
```

The above program makes your computer say hello through its speakers. Notice that the ***say*** command also uses double quotes, to specify exactly what will be said through its speakers.

🗁 Hint

Only the ***say*** command, ***write*** command, and ***title*** command use double quotes. All other commands do not use double quotes. Later you will also learn how double quotes are used by the ***if*** command to compare words or phrases.

Pictures and Sounds!

Pictures and Sounds

The ***picture*** command is used to show a picture, on the screen. Below is an example:

```
start
picture frog
stop
```

A picture of a frog will appear in the results screen, like the one shown below.

Cute looking frog!

To show lots of pictures, just list them like the sample program below. Type it into the editor, and try it out. What do you see in the results screen?

```
start
picture dog
picture frog
picture cat
stop
```

You can also use the ***nextline*** command to make the images all appear below each other, rather than on the same line.

```
start
picture dog
nextline
picture frog
nextline
picture cat
picture oranges
stop
```

To see a list of pictures available in KidBasic, look under the ***picture*** command in the KidBasic Reference Table.

Now let's try the ***soundfx*** command. Type the program below into the editor, and try it out. How does the computer activate your instructions?

```
start
soundfx moo
stop
```

The ***soundfx*** command is used to make a sound effect through your computer's speakers. A sound effect means a special sound, like the ones you hear playing in the background of your favorite movie. This command plays pre-recorded sounds through your speakers.

moo, **crowd laugh**, **quack**, and **volcano**, are some cool sound effects that you can try out in KidBasic.

For your programming and listening pleasure, there is also the ***music*** command. This command plays pre-recorded music. Its very similar to the ***soundfx*** command only it plays music rather than sound effects. Here is a program that uses the ***music*** command.

```
start
music jazz
stop
```

Do you like jazz music?

Quick Quiz

Use all the commands you have learnt so far in 2 computer programs of your own, and ask your computer program end-users to try them out. See what they think about them.

Hint

The new commands we have learnt so far are: ***say***, ***picture***, ***soundfx***, and ***music***. Don't forget to use the ***nextline*** command whenever you think its necessary.

Fun with Buttons

Buttons can be lots of fun because they will do something when you click on them. In this section of the KidBasic programming adventure, you will learn how to create buttons.

You have already learnt some basic commands that gave you a good foundation, now its time to build on them. So, if you have forgotten what some commands do, just type them into the editor and see the results. You can also check out the Reference Table at the back of this manual.

How can you create your very own button? Using the ***button*** command of course! In KidBasic, the ***button*** command is used to place a button in the results screen. Your end-users can then click on this button to make the computer do something. The computer will **activate** whatever is inside the button.

The button command works like a button on a shirt pocket.

Inside this pocket is a pack of bubble gum. To get to the bubble gum, you have to unbutton the pocket. The same thing goes for a button in KidBasic.

The instructions you want the computer to perform are located inside the button. But you must click on the button before the computer will activate all the instructions inside that button. Let's walk through the example below to really understand how KidBasic buttons work.

The following program creates a button on the screen. Type it in the KidKoder and see for yourself. You can name this program anything you want. Be sure to save it.

```
start
button doit
end button
stop
```

In the above program, **doit** is the name of the button that you have just created. Now lets tell the computer what to do when someone clicks on the **doit** button. To do this, we have to type instructions inside the ***button*** command.

Take a look at how the ***button*** command is used.

```
start
button doit

Your instructions
here

end button
```

When the doit button is clicked, the computer will activate all the instructions typed inside the button.

The ***end button*** command lets the computer know that we have reached the end of this button. So what does all this mean?

When anyone clicks on the doit button, the computer will do everything from *button* doit to ***end button***. The ***end button*** command is always required when using the ***button*** command because the computer needs to know when the ***button*** command ends.

Here's an example of a program with two buttons in it. Test the following program in the editor to see how it works. Don't forget to save your work. You can call this program anything you want.

```
start

button button1
write "hey you clicked button1"
end button

button button2
write "you just clicked button2"
end button

stop
```

In the program above, each button will only do what is inside it when clicked. The computer will know which instructions to activate for each button by looking at the name of the clicked button. So, every button in a program should have its own unique name. All the button names in a program must be different or the program will not work correctly.

Your computer can do lots of things when a button is clicked. The program below gives an example of this. Test it out and see for yourself. Then ask your end-users to come and check it out too.

```
start

button mybutton
soundfx moo
background yellow
background red
background blue
write "wow! You can do lots with just one
click!"
end button

stop
```

Button Rules

Certain words cannot be used to name KidBasic buttons, these words include, but may not be limited to the following: empty, nothing, true, false, null, name. You can call a button "yourname" or "newname" but not "name". If you do, your button will not work properly. Also the name of a button cannot be made up of **only** numbers and the name of a button cannot begin with a number.

For example the following instructions are bad:

button 2
end button

button 4boys
end button

However, the instructions below are good:

button boys4
end button

button fourboys
end button

Well, that's it for buttons. Try creating a program using the button command in it. Have fun!

Loops

In computer programming, a loop makes the computer do something over and over again. It's like turning arround in circles lots of times, until you feel dizzy. Same thing with a computer. You can tell the computer how many times you want it to turn around, or repeat itself. In KidBasic you use the loop command to do this. Type the program below into the editor, and activate it.

```
start
loop 5
write " I am in control of my
computer "
end loop
stop
```

What do you see in the results screen? The computer writes, I am in control of my computer, on the screen five times!

To use the ***loop*** command, specify how many times you want the computer to repeat the instructions you put between the ***loop*** and ***end loop*** commands. In the example above, 5 was the number of times specified so the loop command repeated what was inside it, five times. You can specify up to 10 times. If you try to specify a number higher than 10, the loop command will only repeat 10 times. The loop command is set that way so that you would not make your computer get too dizzy! Also, you cannot put buttons in loops, you'll find out why when you learn about jars.

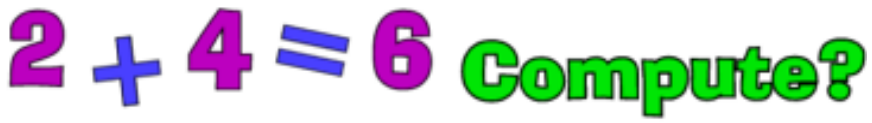
Here are two more sample programs using the ***loop*** command.

Notice that the ***loop*** command always uses the ***end loop*** command to show where the loop ends.

```
start
loop 5
write " This will appear 5
times"
nextline
end loop
stop
```

In the program above, everything between ***loop*** *5* and ***end loop*** will be repeated five times. Here's another program.

```
start
loop 10
say " Cowabunga ! "
end loop
stop
```



Do You Compute?

Computers can do math. Yes, they really can! Let's learn more about writing programs that solve math questions in KidBasic. After all, what's the fun of programming if it can't help you with your homework?
The following are math operators used in KidBasic.

> < <> = + - * ^

By looking at them, I'm sure you will recognize the greater than sign (**>**), the less than sign (**<**), the equal sign (**=**), the plus sign (**+**) and minus sign (**-**). The little star that you see next to the minus sign is the multiply sign (*). The diamond like sign (**<>**) means, "not equal to", this is used when two numbers are not equal in value. For example, 5 does not have the same value as 4. But (4+1) has the same value as 5. So, 4 <> 5, but 4 + 1 = 5.

Take a look at this next program called **mathstuff** to see how some of these signs are used. You don't have to test this one yet.

```
start

if 2=3 then
write "2 is equal to 3"
compute 2+3
end if

if 2=2 then
write "2 is equal to 2"
compute 2–2
end if

if 2 <> 3 then
write "2 is not equal to 3"
compute 2*3
end if

stop
```

The ***compute*** command is used to tell the computer to calculate something. When this command is used, only the result is shown in the results screen.

For example:

```
start
compute 2 + 2
nextline
compute 2 ^ 3
stop
```

The program above will show the number 4 on the screen, move to the nextline, and then show the number 8, like this:

```
4
8
```

In the program we used the, raised to power operator, which looks like this: **^**

The raised to power operator is the same key as the "6" key, on a Standard English (US) keyboard. To type it, hold down the shift key first, then the "6" key.

2 **^** 3 means:
2 raised to the power of 3, or 2 raised to the 3rd power, this means 2 multiplied by itself, 3 times. Like this: 2 * 2 * 2 and that's how we got 8.

More examples:
3 **^** 2 means 3 * 3, and the result is 9
7 **^** 4 means 7 * 7 * 7 * 7, and the result is 2401

Now back to the **mathstuff** program. To really understand the **mathstuff** program, we have to learn about the ***if***, ***then*** and ***end if*** commands. These are new commands. They tell your computer how to make a choice.

Choices, Choices, Choices… if, then, end if commands

Let's pretend that on one Christmas Eve night, you just happened to stay up late enough to actually meet Santa Claus! But since it's so late at night, its dark, and Santa can't tell if you're a boy or girl. He asks you to pick out your very own Christmas gift from his huge sack. Naturally you want a toy.

He gives you a condition. If you are a boy, then you must choose a race car, but if you are a girl, then choose a doll. The type of race car or doll you choose is completely up to you.

The same principle applies to the ***if***, ***then***, ***end if*** commands used in KidBasic to help your computer make a choice. Let's write out the instructions that Santa gave you, using these commands.
It would look something like this:

if you = boy, ***then*** pick a race car
if you = girl, ***then*** pick a doll

If, ***then***, ***end if*** are used in KidBasic to tell a computer how to make a choice about something. Just like Santa used similar words to help you choose either a race car or doll in our story.

Let's look at another example. The example is the program below. You can save the program as **compare**. It uses the ***if***, ***then***, ***end if*** commands. Before you type it into the editor, can you tell what it's going to do? What do you think will appear in the results screen?

```
start
if 2 > 3 then
write "2 is greater than 3"
end if

if 3 > 2 then
write "3 is greater than 2"
end if
stop
```

Now type it into the editor and activate it. Are the results what you expected? In the **compare** program, ***if*** 2 is greater than 3, ***then*** the computer will write "2 is greater than 3", but ***if*** 3 is greater than 2, ***then*** the computer will write "3 is greater than 2".

It's just that simple, pretty neat!

Did you notice the ***end if***? Whenever you have completed the instructions for the ***if*** command, you have to put an end to it using the ***end if*** command. This will let the computer know you are done. The ***end if*** command is required every time you use the ***if***, ***then***, and ***end if*** commands.

Quick Quiz

Type the **mathstuff** program into the editor, to test it. What does it do?

Create a program using the following instructions, and activate the program to see the results.

A. If 3 is less than 2, write "shalom"
B. If 3 is equal to 2, write "3 is equal to two"
C. If 3 is not equal to 2, write " 3 is not equal to 2"

Hint

Don't forget that the not equal to sign is: **<>**

Using the ***if, then, end if*** command with words or phrases

When you want to use the ***if, then, end if*** commands to compare words or letters of the Alphabet, you must use double quotes (**" "**) around the letters of the alphabet. If you are comparing numbers, you don't need to use double quotes, only use double quotes with words or letters. The program below will help us to understand a little better.

The program compares two words to see if they are the same. If they are the same, the computer writes, "Hey the words are the same" in the results screen. If they are not the same it writes, "These words are not the same".

```
start
if “mango”= “mango” then
write “yes, the words are the same”
end if
if “mango” <> “mango” then
write “these words are not the same”
end if
stop
```

Notice that the word “mango” is always in double quotes. This is because it is a word or it is made up of letters of the alphabet. If this had been a number then we would not use double quotes around it.

Test the program in the editor. The computer will only write “Yes, the words are the same” in the results screen, because “mango” is truly equal to “mango”.

Ok, let’s move on, Jars are next.

What's in Your Jar?

What's In Your Jar?

So far in this programming adventure, you have only learned how to create computer programs that will show your end-users something in the results screen. You haven't learnt any commands that can allow your end-users to respond to any of your programs. Your end-users can respond to your programs, by typing information such as words, sentences and numbers into containers called KidBasic jars.

Yes, you did learn how to create buttons that did something or displayed a message when your end-users click on them. But what if you wanted to ask your end-users questions through your computer program? Like, "how old are you" and "can you guess my password?" How would you do it? How would your end-users respond to you? This is where Jars come in handy.

Jars are used to collect your end-user's response to the questions you ask them through your computer program. Just like you would keep cookies in a cookie jar, or coins in a coin jar, you would keep your end-user's response in a jar as well.

You can make the computer create Jars using the ***newjar*** command. When the jar is created, the end-user will see it on the screen and can type numbers or words into it. To use this command, you also have to give your jar a name. So what type of name should you give your jar? Well if you keep cookies in a jar, you would call it a cookie jar, right? So, if you want your end-user to keep their nickname in a jar, you would call that jar a nickname jar; or, if you wanted your end-users to put their age in a jar, you could call that jar an age jar.
In the KidBasic language, you create Jars to keep numbers or words in them. The names of jars must be one word or you must merge the name to become one word. When typing your jar's name into your program, it should not have spaces in it. Here are some examples.

Correct jar names	Wrong jar names	What they do
phonenumber	phone number	This jar keeps your end-user's phone number
Address	Add ress	This jar keeps your end-user's address
Password	Pass word	This jar keeps a password

Yourname	Name	This jar keeps a end-user's name.

You can create as many jars as you want, but every jar in a program must have its own unique name. The names of all the jars in a program must be different or the program will not work correctly. For example, you can create a number jar to keep a number in, or a color jar to keep a color in. It's all up to you.

Jar Rules

Certain words cannot be used to name KidBasic jars, these words include, but may not be limited to the following: empty, nothing, true, false, null, name. You can call a jar "yourname" or "newname" but not "name". If you do, your jar will not work properly. Also the name of a jar cannot be made up of **only** numbers and the name of a jar cannot begin with a number.

For example the following instructions are bad:
newjar 2
newjar 4boys

However, the instructions below are good:
newjar boys4
newjar fourboys

```
start
say "please enter a nickname into the
nickname jar"
write "please enter an age into the age jar"
newjar nickname
newjar age
stop
```

In the above program, the computer asks an end-user to enter a nickname into the nickname jar. The computer also creates the nickname jar on the screen. An end-user will now be able to type their nickname into the jar.

Now let's learn how to make the computer get the value that an end-user types into the jar. The value of a jar is whatever an end-user types into that jar. For example, if an end-user types "tommy" into a jar, the value of this jar is "tommy". If an end-user types the number 8 inside a jar, the value of that jar is 8.

Get the jar's value

Let's pretend you want to eat one of those delicious cookies from the cookie jar example. To do that, you have to put your hand in the jar and get the cookie. In KidBasic, it would be a value instead of a cookie.
But why would you want to get the value your end-user typed into a jar? Well the same reason for wanting to get the cookie out of the cookie jar. You want to use it in the rest of your program, just like you use cookies to fill your tummy.

For the computer to get the value inside a jar, type the name of the jar and then a dot (**.**) then the word ***value***. So it looks like this.

nickname.***value***

In the sample program above, let's assume your end-user types "Rumplestiltskin" into the nickname jar you created. To get the value out of the nickname jar for use in your program you have to type: nickname***.value***. This is the name of your jar plus the dot value.

If the name of your jar was number, then to get the value entered into the number jar, you must type: number***.value***. Let's test some examples to understand this better. Type the following program into the editor.

```
start
newjar myjar
button mybutton
display  myjar.value
end button
stop
```

Type any number, letters, or words into myjar from the results screen, then press the button to see what happens.

By now you would have noticed that the ***display*** command just shows what is in the jar. This doesn't seem like much until we start doing some really interesting things with jars and buttons. The ***display*** command can also be used like the ***compute*** command with math operators. Notice that the ***display*** command does not use double quotes ("). So, keep that in mind when you are creating your own programs.

🗁 Hint

Display commands should always be inside buttons.
When a button containing a display command is clicked, the computer will show the value that the end-user entered into the KidBasic jar.

Here are some other programs to help you learn how to use KidBasic jars.

1) Enter your age

```
start
write "enter your age"
newjar age
button doit
write "the age you entered is"
display  age.value
end button
stop
```

This is another simple example of jars with buttons. This time, a message is written in the results screen when your end-user clicks on the doit button you created in your program. Let's pretend your end-user entered 40 into the age jar. In the results screen you should see something like this.

the age you entered is 40

2) A hundred years from now

```
start
write "enter your age"
newjar age

button PressMe
write "this is how old you will be 100 years from
now"
display age.value + 100
end button

stop
```

Let's see what your end-user's age will be a hundred years from now! In this program you have instructed the computer to add 100 years to the age the

end-user entered in the age jar, and then display it on screen for all to see. Let's assume your user enters the number 40. In the results screen you should see something like this.

This is how old you will be 100 years from now 140

Isn't it cool that you can add numbers to the value of a jar? As long as the value inside a jar is a number, you can continue to treat it just like a number! You can add to it, subtract from it, multiply it, divide it, and much more. This is very exciting because you don't have to know what number your end-user enters. They can use any number, and you can do lots with it.

3) Fun with numbers

```
start
write "Please enter a number"
nextline
write "Press mybutton when you're done"

newjar myjar

button mybutton
write "The number you entered, plus 2 is"
display  myjar.value + 2
write "The number you entered, multiplied by 2
is"
display  myjar.value * 2
write "The number you entered, minus 2 is"
display   myjar.value – 2
write "The number you entered, divided by 2
is"
display   myjar.value / 2
end button
stop
```

See? This is awesome! Test this program out and see what it does with the number your end-user entered. Save it with any name you want and let other end-users play with it. If a end-user enters the number 25 into myjar, here is what you will see in the results screen.

The number you entered, plus 2 is 27
The number you entered, multiplied by 2 is 50
The number you entered, minus 2 is 23
The number you entered, divided by 2 is 12.5

4) Adding Jars

```
start
newjar firstjar
newjar secondjar

button doit
display firstjar.value + secondjar.value
end button

stop
```

I'm sure you can guess what this program does without even trying it out. But let's pretend that your end-user enters 20 into the firstjar and 25 into the secondjar. You would get this in the results screen.

45

Yep, you've had enough training on jars. Remember, if the value inside a jar is a number, then it can be treated just like any number! And as you have seen, that can open up a world of possibilities! So go ahead and get creative!

There is just one thing to be aware off while your creative mind is spinning. You cannot put jars or buttons in loops, and here's why.

Remember, every button must have its own name, and every jar must also have its own name. These names have to be unique. This means you cannot have two jars or buttons with the same name in a program.
Well, if you put a jar or button in a loop, you'll create many jars or buttons with the same name. This won't work since their names are not different. Make sense?

You can continue being creative now!

Guess Game

Let's write a computer program called **guess**. When your end-user tests this program, they will enter a word inside a guess jar and press a button called test. If the word they entered is the same as the secret word inside your program, then the computer will ***write*** and ***say*** "You guessed right!" and it will play a ***soundfx*** called **crowd cheer**.

The results screen should also show the ***picture*** of a happy **man**. If the word they entered is not the same as the hidden word inside your program, then the computer will ***write*** "wrong guess" and play a ***soundfx*** called **buzzer**, and show the ***picture*** of a **sad man**.

The ***title*** of the program should be "Welcome to my guessing game program", but you can modify it to include your name if you want. In this program, you will need a jar named guess, and a button named **test**.

The computer will write the following on the results screen. "Enter your guess then press the test button." You will also need to write a hint for the end-user to help them guess the hidden word because it will be too hard without a hint. The computer should write a hint underneath the ***title***. In this example, the hidden word is orange. So, the hint is: "Hint: A fruit, round and orange in color!" Let's take a look at the guess game.

Let the Games Begin!

The guess game!

```
start
title "welcome to my guessing game program "
write "Enter your guess then press the test
button”
write "hint: a fruit, round and orange in color!"

newjar guess

button testbutton

if guess.value = "orange" then
write "you guessed right!"
soundfx crowd cheer
picture happy man
end if

if guess.value <> “orange” then
write "wrong guess"
soundfx buzzer
picture sad man
end if

end button
stop
```

Did you notice that the ***if*** choice statements were inside the button? That makes sense because the program will not compare the end-user’s guess to your hidden word until the **test *button*** is pressed.

Hint

You should put the ***if***, ***then*** and ***end if*** commands inside a button, when you want to use them to compare values that an end-user will enter into a jar. The ***if***, ***then*** and ***end if*** commands will compare the end-user’s value, when the button is clicked.

Cool Projects!

You will find four cool projects in this section. The guess game is an example of how to work on these projects. They're simple, and all they do is apply the cool things you have learned about KidBasic. Each project will have little hints to help you along. Take your time, and remember to have lots of fun!

When you are done with each project, test your program to see how well it works, then invite your friends and family to play with your programs!

Project 1, Cracking the Password

You can name this program **password.** The object of this program is to have the end-user crack your password. The end-user will enter a word inside a password jar. If the word they entered is the same as the secret word inside your program, then the computer will ***write*** and ***say*** "Password correct. Access Granted", and play a ***soundfx*** called **crowd laugh** and show the ***picture*** of a **roller coaster**.

If the word they entered is not the same as the hidden word inside your program, then the computer will ***write***, "Intruder Alert! You do not have the secret password, please stay out". The computer should also ***say***, "intruder alert" three times using the ***loop*** command. The computer should then show the ***picture*** of a **secret agent**.

The ***title*** of the program should be "Please enter the secret password and press the authorize button". You may modify this title if you wish.

🗁 Hint

In this program, you will need a jar named password, and a button named authorize. In this program, you don't need to write a hint for the end-user but that's up to you. For example, if the hidden word is your nickname, then you can make the computer ***write*** a hint for the end-user like this: "Hint: My nickname"

Project 2, Investment advice business

For this project you will start your own investment advice business. The ***title*** of this program will be "Abby the Investment Advisor". Change the name, Abby to your name, of course if your name is Abby don't change it. In this

program, you will ask the end-user to enter a stock price. ***write*** "Sir or Madam, Please enter the final stock price for last month and final stock price for this month".

Prompt the end-user to click on a button with another ***write*** command. It could go something like this: ***write*** " Press the button for advice on this stock."

Hint

You will need to create two jars for this program, they are: thismonth and lastmonth. You will also need a button called, button. The following is what should happen when the button is clicked:

if thismonth.value is greater than lastmonth.***value then*** the computer should ***write*** "Buy this stock, it is likely to go up". The computer should also show a ***picture*** of an **arrow up**, pointing upwards. The computer should play the ***soundfx*** called **cheerleader.**

However, ***if*** lastmonth.***value*** is greater than thismonth.***value***, ***then*** the computer should ***write*** "Sell this stock, it is likely to go down". The computer should show the ***picture*** of an **arrow down**, pointing downwards. The computer should also play a ***soundfx*** called **buzzer**.

Project 3, Sales Tax

In this project you will create your own software that calculates how much customers must pay in taxes each time they buy a product from the store. Let's pretend your end-users want to buy some of the programs you have written in KidBasic. Yes, they like them that much!

For the Sales Tax program, you will need a jar called price, and a button called tax. You should make the computer ***write*** "Please enter the price you wish to pay for my computer programs." The end-user will enter the price they wish to pay. When the end-user clicks the tax button, the computer should ***display*** the value the end-user entered multiplied by 5% sales tax. 5% is 5 divided by 100, or 0.05. You would use the ***display*** command like this:

```
display price.value * 0.05
```

The ***title*** of your program should be "Ashley's Retail Store". Again, you can change Ashley to your real name. Using the ***write*** command, make the computer ***write*** "Total tax on all items in this state is 5%". Then use the ***nextline*** command to make your text show up on the next line.
Use another ***write*** command to make the computer ***write*** "We will multiply your total purchases by 0.05". Then use the ***nextline*** command again.

Use another ***write*** command to make the computer ***write*** "Please enter the price of the item you want to purchase. When done, click on the tax button".

Use the ***nextline*** command again. Now, when the end-user clicks the button, the computer should ***write*** "The tax added to your purchase price is." Now use the ***display*** command as explained above to show the value of the price jar multiplied by 5% sales tax (0.05).

Project 4, The Power of compound interest

For this project, you will start your own Savings Bank Corporation. The project will also teach you about the advantages of saving some of your money in a bank. If you save your money in a real bank, it will accrue interest - the bank will add more money to it. Yep! Money can actually grow! One of the ways your money can grow is if you put it into a type of bank account called a Certificate of Deposit, CD for short.

Money grows through a compounded interest rate. The "rate" is a percentage of the total amount of money in your bank account. Sometimes its 4 or 5 %, and it determines how fast your money will grow every year.

For example, if the rate is 5%, then the bank will add 5% to the total amount of money in your account. So, your bank account increases by 5% every year. If you have $20.00 in your account, at the end of the year, the bank will add $1.00, which is 5% of $20.00.

Next year, the bank will add 5% of $21.00 to your account, which is $1.05. This brings your total to $22.05.

So, the higher the rate, the faster your money grows. Also, the longer you leave your money in the Certificate of Deposit, the more your money will grow! Make sense?

Create a program that will tell us how much money we will have in the bank if we put $10.00 in our Certificate of Deposit, and if it grows at the rate of 5%, and we leave the money in there for 30 years.

In this program, you will have your own savings bank where an end-user can come to save some of their money in a CD.

The ***title*** of the program will be "Victoria & Friends International Bank." Again, change Victoria to your name. Use the picture command to show a picture called, **treasure.** Use the write command like this, ***write*** "Today, our CD rate is 5%, so come bank with us!" , ***write*** "Please type 50 into the CD jar." Again, ***write*** "Please enter 30 in the years jar" and then ***write***, "Then click on the Deposit Button ."

You will need to create a CD jar for this program. Remember that CD is short for Certificate of Deposit. You also need to create a years jar. Create a ***button*** called **deposit**. When the end-user clicks on the **deposit** button, this is what should happen:
The computer should ***write*** "If you save your money in a Certificate of Deposit for this many years". Your computer should also ***write*** "Your money will grow to". The computer should then use the *display* command like this:

```
display cd.value *((1 + 0.05) ^ years.value)
```

The way the display command is used above means: the value of cd jar multiplied by 1**+** 5%, *raised to the power of* the number of years it is saved in the bank.

This is actually the math formula to calculate compound interest (1 + R)^Y.

Use the ***write*** command to ***write*** "dollars". For an account of 50 dollars after 30 years, you will exactly have $216.097118757533 dollars. This is approximately $216.09 dollars.

After you have tested your program and it works, change what the computer writes under the ***title*** to "Please enter your money into the cd jar", instead of "Please enter 50 into the cd jar". This way your end-users can enter any amount they wish to save. Also change the next ***write*** command to "Please enter the number of years in the years jar" instead of "Please enter 30 in the years jar". This way, your end-users can enter how long they wish to keep their money in your bank.

When you have tested this program, you'll see how big your money can grow when you save it in a bank. Think how much you could have if you keep putting money in your CD. It pays to save!

You're Done!

Well that's it for this version of KidBasic. But don't stop here, think about new programs you can create. Test your new programs and invite end-users to play with them. You can even modify the programs you have already done so that the computer will act differently each time you make changes.

KidBasic Reference Table

start

This is always the first command in any KidBasic program. It tells the computer to start activating the instructions in your program.

stop

This is always the last command in any KidBasic program. It tells the computer it has reached the end of a computer program. There are no longer instructions for it to do.

title “”

This will place a nice title in your program. Example of how it is used:

title “ Welcome to my program”

write “ ”

This command writes to the results screen. Example of how it is used:

write “ This will be written in the result window”

background

Changes the background color of the results screen. Example of how it is used:

background Blue

soundfx

Plays a pre-recoded sound effect file though the computer’s speakers. Example of how it is used:

soundfx **honk**

The following sound effects are available in KidBasic:
buzzer, crowd cheer, honk, phone ring, school bell, siren, alert, puddle, zap, chirp, meow, cluck, moo, croak, bleat, bark, neigh, howl, hyenas, lion, pant, woof, rooster, tiger, whale, hurricane, jungle, boing, blast, critters, bulletin, slide, waves, crowd laugh, delicious, glass break, horse, howl, laser, pour drink, blob, swoosh, tictoc, toy laser, zap

music
Plays pre-recorded music files through attached computer speakers. Example of how it is used:

music **birthday**

The following music files are available in KidBasic: *disco, jazz, birthday, classical, blues, hiphop, lullaby, techno, wedding, reggae, and country*.

nextline
The nextline command, tells the computer to show command results on the next line. Example of how it is used:

write "What is your name?"
nextline
say "where are you from?"
nextline

picture
Shows pictures in the results screen. Example of how it is used:

picture aloha

The following pictures are available in KidBasic:
aloha, angel, antelope, apples, arrow down, arrow up, astronaut, baby boy, baby girl, bananas, ben franklin, book, bull, bumper car, birthday party, bear, bunny, butler, cabin, carrots, cashier, castle, cat, chihuahua, cow, cheer leader, cherries, clown, coconuts, comet, computer, Dr. Genius, daises, doctor, dog, dolphins, elephant, elephant ride, fathers day, ferris wheel, fireman, fish, flag, fox, frog, giraffe, grapes, graduate, ground hog day, happy birthday, happy man, horse, hula, judge, jumping man, ketchup, lemons, lifeguard, lion, mail man, merry christmas, milk man, monkey, moon, mothers day, mustard, new year, noel, oranges, oil rig, otter, pears, penguin, pharaoh frogs, pig, plague flies, platypus, police car, police man, radishes, reporter, robin, roller coaster, sad man, santa, scuba diver, secret agent, sheep, snow mobile, space scooter, strawberries, sun, swat, tax day, tax time, teacher, thanksgiving day, tiger, tomatoes, turtle, treasure, tropical girl, watermelon, wedding, wedding bells, windmill, wolf.

newjar
Creates a jar in the results screen that can hold numbers or characters. The default value inside the jar is 0. That value will be replaced with what ever is entered into the jar. Example of how this command Is used:

newjar number

say “”

Makes the computer say words and sentences through the computer speakers. Example of how it is used:

say “hello”

button

Creates a button in the results screen. When the button is clicked, the computer begins to activate all the instructions typed inside the button. For more examples, see “Fun with buttons”. Example of how it is used:

```
button button1
say “I am inside button1”
end button
```

end button must be at the end of every button command.

display

Displays the value in a jar on the screen. See “What’s in your Jar” for more examples. Example of how it is used:

display number jar.***value***

spacebar

Acts like the space bar on your keyboard.
You can use this command to push pictures, buttons or words to the right of your screen.

Example of how it is used:

```
spacebar 5
picture frog
```

These spacebar command above, will push the picture of the frog 5 spaces to right of the results screen.

compute

Uses math operators for calculations. Example of how it is used:

compute 5 *5

Examples of Math operators are:

<> not equal to; means that two numbers, words, or letters are not equal to each other. Example: 2 **<>** 4 This says 2 is not equal to 4
= equal to; example: 2 **=** 2 means 2 is equal to 2
< less than; example: 2 **<** 3 means 2 is less that 3
> greater than; example: 2 **>** 1 means 2 is greater than 1
^ raised to power; example: 2 **^** 3 means 2 raised to the power of 3

If, then, end if

These are used for comparing values in KidBasic. See "Choices, choices, choices... ***If***, ***then***, and ***end if*** commands" for detailed explanation. Example of how these commands are used:

```
if 10 > 2 then
write "10 is greater than 2"
write "Yep. 10 is NOT less than 2"
end if
```

end if must be at the end of every if statement.

END-USER LICENSE AGREEMENT FOR KIDBASIC SOFTWARE AND KIDBASIC MANUAL

This KidBasic Corporation End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and KidBasic Corporation for the KidBasic Corporation software product identified above, which includes computer software and may include associated media, printed materials, and "online" or electronic documentation ("SOFTWARE PRODUCT"). The SOFTWARE PRODUCT also includes any updates and supplements to the original SOFTWARE PRODUCT provided to you by KidBasic Corporation. Any software provided along with the SOFTWARE PRODUCT that is associated with a separate end-user license agreement is licensed to you under the terms of that license agreement.

SOFTWARE PRODUCT LICENSE
The SOFTWARE PRODUCT is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE PRODUCT is licensed, not sold.

1. GRANT OF LICENSE.
This EULA grants you the following rights:

- Systems Software.
  You may install and use the SOFTWARE PRODUCT on multiple computers, including workstations, terminals or other digital electronic devices ("COMPUTER"). Feel free to share this product with anyone, in any way. However you may not reverse engineer, decompile, or disassemble the SOFTWARE PRODUCT OR MANUAL.

2. DESCRIPTION OF OTHER RIGHTS AND LIMITATIONS.

- Limitations on Reverse Engineering, Recompilation, and Disassembly.
  You may not reverse engineer, decompile, or disassemble the SOFTWARE PRODUCT.

- You may make copies of this product. You may transfer any copies of this product.

- Separation of Components.
  The SOFTWARE PRODUCT is licensed as a single product. Its component parts may not be separated for use on more than one computer.

- Rental.
  You may not rent, lease or lend the SOFTWARE PRODUCT.

- Trademarks.
  This EULA does not grant you any rights in connection with any trademarks or service marks of KidBasic Corporation.

- Termination.
  Without prejudice to any other rights, KidBasic Corporation may terminate this EULA if you fail to comply with the terms and conditions of this EULA. In such event, you must destroy all copies of the SOFTWARE PRODUCT and all of its component parts.

3. COPYRIGHT.

All title and intellectual property rights in and to the SOFTWARE PRODUCT (including but not limited to any images, photographs, animations, video, audio, music, text, incorporated into the SOFTWARE PRODUCT), the accompanying printed materials, and any copies of the SOFTWARE PRODUCT are owned by KidBasic Corporation or its suppliers. All title and intellectual property rights in and to the content which may be accessed through use of the SOFTWARE PRODUCT is the property of the respective content owner and may be protected by applicable copyright or other intellectual property laws and treaties. This EULA grants you no rights to use such content. All rights not expressly granted are reserved by KidBasic Corporation.

4. DUAL-MEDIA SOFTWARE.

You may not loan, rent, lease, sell other mediums of the software product or manual to another user.

5. COPY and REDISTRIBUTION.

You may copy the CD and redistribute it as a whole unit but you must not charge a fee in any way. You may NOT redistribute parts of the CD contents separately. You may share your CD but you must not charge a fee for it. You are not allowed to charge any costs for the CD, you are not allowed to re-sell KidBasic in whole or in part. To do this would be in violation of this EULA.

6. U.S. GOVERNMENT RESTRICTED RIGHTS.

The SOFTWARE PRODUCT and documentation are provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1) and (2) of the Commercial Computer Software--Restricted Rights at 48 CFR 52.227-19, as applicable. Manufacturer is KidBasic Corporation/

KidBasic Corporation
14625 Baltimore Ave, suite 116
Laurel MD 20707

7. EXPORT RESTRICTIONS.

You agree that you will not export or re-export the SOFTWARE PRODUCT to any country, person, entity or end user subject to U.S. export restrictions. You specifically agree not to export or re-export the SOFTWARE PRODUCT: (i) to any country to which the U.S. has embargoed or restricted the export of goods or services, which currently include, but are not necessarily limited to Cuba, Iran, Iraq, Libya, North Korea, Sudan and Syria, or to any national of any such country, wherever located, who intends to transmit or transport the products back to such country; (ii) to any end-user who you know or have reason to know will utilize the SOFTWARE PRODUCT or portion thereof in the design, development or production of nuclear, chemical or biological weapons; or (iii) to any end-user who has been prohibited from participating in U.S. export transactions by any federal agency of the U.S. government.

MISCELLANEOUS

- If you acquired this product in the United States, this EULA is governed by the laws of the State of Maryland.
- If you acquired this product in Canada, this EULA is governed by the laws of the Province of Ontario, Canada. Each of the parties hereto irrevocably atones to the jurisdiction of the courts of the Province of Ontario and further agrees to commence any litigation which may arise hereunder in the courts located in the Judicial District of York, Province of Ontario.
- If this product was acquired outside the United States, then local law may apply.

Should you have any questions concerning this EULA, or if you desire to contact KidBasic Corporation for any reason, please write:

KidBasic Corporation Sales Information Center
14625 Baltimore Ave, suite 116
Laurel MD 20707

NO WARRANTIES.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, KIDBASIC CORPORATION AND ITS SUPPLIERS DISCLAIM ALL WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, WITH REGARD TO THE SOFTWARE PRODUCT and MANUAL, AND THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES.

NO LIABILITY.
TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL KIDBASIC CORPORATION OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE PRODUCT OR MANUAL OR THE FAILURE TO PROVIDE SUPPORT SERVICES, EVEN IF KIDBASIC CORPORATION HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.