Name _____

Date _____

EXPLORATIONS

Activity 6

Heads Up! (Continued)

In this activity, you will do more experiments with simulations and use a calculator program that will quickly simulate multiple coin tosses.

The Problem

Recall from *Activity 5: Heads Up!* that we used the calculator command **iPart 2rand** to simulate the tossing of a coin. This command produces an output of either **0** or **1**. We chose **0** to represent a tail on the coin and **1** to represent a head. By repeatedly pressing ENTER we were able to simulate many tosses of a coin.

- Before you begin, go to #1 in the **Questions** section of this activity and write what you think the term *simulation* means. Share your ideas with your classmates.

You will now explore how to write and enter a calculator program that will do the button pushing and coin tossing for you. The program you will be entering is shown below. Instructions for entering this program are on the next page.

```
Disp "N="
Input N
For(K,1,N)
iPart 2rand→R
If R=0
Disp "T"
If R=1
Disp "H"
Pause
End
```

This activity continues work begun in Activity 5: Heads Up! It may take two 50-minute class periods to complete.

*Taking the time to get a whole class definition of the term **simulation** gives you an opportunity to hear the students' interpretation of the term. It may be a challenge for some to believe that a simulation is a reliable replacement for actually conducting an experiment.*

If students have never worked with the programming capabilities of the calculator, you may want to begin by having them clear calculator memory. Resetting the memory is not necessary; doing so just removes information that might distract students as they work, since the process erases all programs and data from the calculator.

To reset memory on the TI-80 or TI-82, press 2nd [MEM] ***3:Reset*** , ***2:Reset****. On the TI-83, press* 2nd [MEM] ***5:Reset, 1:All Memory, 2:Reset****.*

Following this, students may have to adjust the display screen by holding down the 2nd *key and pressing* ▲ *or* ▼ *to darken (or lighten) the screen contrast.*

There are several different types of cursors. Normally, the cursor is a solid blinking rectangle . The students should notice when they press the ***ALPHA*** *key that the cursor changes to a blinking* A*. If they press* ***ALPHA*** *again, the cursor returns to the solid rectangle.*

Have the students enter the program first, run it a few times, and then come back to discuss the purpose of each program line.

Using the Calculator

1. Press PRGM and use ▶ to move the cursor over to the **New** menu to prepare the calculator for entry of a program.

2. Press **1:Create New**. Your screen should look like the one at the right with the cursor appearing as an A that is blinking. The blinking cursor indicates that the calculator is in the **ALPHA** mode.

   If you look at your calculator keyboard, you will find letters of the alphabet written directly above and to the right of many of the keys. When the calculator is in **ALPHA** mode, if you press a key with a letter of the alphabet above it, you will get that character on the screen.

3. You are going to enter a program into the calculator that will be named **COIN**. Type in this title by pressing the four keys that have the letters C, O, I, and N above them. Once finished, press ENTER. Your screen should look like the one at the right.

Notice that the screen has changed. The program title is at the top of the screen and the cursor is now blinking on the first line of the program following the colon. This indicates that the calculator is ready to accept the commands that will make up the program instructions. You are ready to enter your **COIN** program.

Table 6.1 provides the keystrokes to enter program commands, an illustration of what the calculator looks like after those commands have been entered, and the purpose of each command in the program. You may want to have a partner help you enter the program so that one of you can read the instructions and the other can enter the commands on the calculator.

Table 6.1. Entering the COIN Program

Keystrokes	Window Display	Purpose
PRGM **I/O** 3:Disp 2nd [A-LOCK] "N 2nd [TEST] 1:= ALPHA " ENTER		Displays **N=** when you run the program. This asks (or prompts) you to enter the number of times (N) you want to toss a coin.
PRGM **I/O** 1:Input ALPHA N ENTER		Takes the value you entered after the prompt **N=** and enters that value for the variable N. This stores the number of times the coin toss will be simulated.
PRGM **CTL** 4:For ALPHA K , 1 , ALPHA N) ENTER		Sets up another variable K to count by one from 1 through N. Each program instruction between the **For** and the **End** statement (see the last line of program code) is carried out N times.
MATH **NUM** 2:iPart 2 MATH **PRB** 1:rand STO▶ ALPHA R ENTER		R represents the result of each roll, either a 0 or a 1. This line of the program stores either a 0 or a 1 into the variable R.
PRGM **CTL** 1:If ALPHA R 2nd [TEST] 1 0 ENTER		The program asks if R (the outcome of the roll) is equal to 0. If the answer is *yes*, the program continues to the next line. If the answer is *no*, the next line of programming is skipped.
PRGM **I/O** 3:Disp 2nd [A-LOCK] "T" ENTER		Displays a T when the value of R is 0. This simulates tossing a TAIL. If R does not equal zero, this line is skipped.

*Tell students that selecting or highlighting a submenu (for example the **I/0** submenu found under the **PRGM** menu) is accomplished by using the blue arrow keys.*

*For the TI-80, the **Disp** command is option **2** under **I/O** menu. Also, the alpha-lock stays on after you press ENTER. You must press the ALPHA key to return to the standard cursor.*

*For the TI-83, **iPart** is option **3** in the **NUM** menu. Recall also that the TI-83 automatically inserts a left parenthesis in the expression.*

Table 6.1, Continued

Keystrokes	Window Display	Purpose
PRGM **CTL** 1:If ALPHA R 2nd [TEST] 1:= 1 ENTER	PROGRAM:COIN :Input N :For(K,1,N) :iPart 2rand→R :If R=0 :Disp "T" :If R=1 :	The program asks if *R* (the outcome of the roll) is equal to 1. If the answer is *yes*, the program continues to the next line. If the answer is *no*, the next line of programming is skipped.
PRGM **I/O** 3:Disp 2nd [A-LOCK] "H" ENTER	PROGRAM:COIN :For(K,1,N) :iPart 2rand→R :If R=0 :Disp "T" :If R=1 :Disp "H" :	Displays an *H* when the value of *R* is 1. This simulates tossing a HEAD. If *R* does not equal 1, this line is skipped.
PRGM **CTL** 8:Pause ENTER	PROGRAM:COIN :iPart 2rand→R :If R=0 :Disp "T" :If R=1 :Disp "H" :Pause :	This pauses the program allowing you to view the display screen until you press ENTER again.
PRGM **CTL** 7:End	PROGRAM:COIN :iPart 2rand→R :If R=0 :Disp "T" :If R=1 :Disp "H" :Pause :End	Identifies the end of the loop of commands.

*For the TI-80, the **Pause** command is option **6** and **End** is option **5** under the **CTL** menu.*

Once the program is entered, have the students run it as it is written a few times so that they see the randomness of the tosses just like they would if they were really tossing a coin.

Later, students will modify the program to insert a counter. Thus, the "view" of the individual tosses is lost. When they see the results 47 heads and 53 tails, they need to know that the calculator didn't just have a run of 47 heads or 53 tails but that these outcomes did indeed occur randomly and that in the long run, we expect about half to be heads and half to be tails.

*If students cleared their calculator memories, **COIN** should be the only program in the calculator. If not, they may have several other programs,*

Once you have finished entering the program, press 2nd [QUIT]. The cursor will return to the home screen. You are now ready to run the program to see what happens!

1. Press PRGM and select the **COIN** program by entering the number of the line on which the name of the program occurs. Press ENTER and the calculator will begin to run the program.

   The prompt **N=** should appear, waiting for you to enter how many times you want to toss a coin.

2. Enter 5. The result of the first toss should appear on the screen.

3. Remember that there is a **Pause** command in the program, so you need to press [ENTER] four times to see all five tosses.

4. Press [ENTER] one last time to get the calculator message **DONE**, which indicates that all programming commands have been run. (If you press [ENTER] again, the **COIN** program will run again.)

and they may need to scroll down the screen with [▼] to get to the line containing the program name. If they scroll to select a program, they can press [ENTER] to select the program and [ENTER] again to start the program execution.

"So what's the big deal?" you may ask. For one thing, your calculator tosses coins much more quietly than using the real thing. But more importantly, you have the beginnings of a program that can allow you to do some other interesting things more efficiently than tossing coins. You have also begun to think about how to communicate with and instruct a programmable calculator. You have to write a program carefully so that the calculator will do exactly what you want it to do. This can get complicated because you need to include every important detail when writing a program.

After running the program, some students may not be impressed. "We can do this with a real coin so why have the calculator do it?" This is a valid point, and we want students to make good choices on appropriate times to use technology. When conducting simulations, students should conduct many trials to increase the reliability of the probability estimate. Technology pays off here, since they will probably not want to toss a coin 600 times.

✎ Before going further, go back and study the last column of the table you used when entering the program. Discuss each line of command with your group or class and try to make sense of the programming lines. Now answer #2 in the **Questions** section of this activity.

*At this point, encourage students to think about the functions of the program lines. Ask them to remove the **Pause** line and notice the impact. Your biggest challenge is not to tell the students what will happen. Let them experiment with changes, run the program, and tell* you *what happens.*

Now you are going to delete the **Pause** command and run the program to check your prediction.

1. Press [PRGM], press [▶] to select **EDIT,** and press [ENTER].
2. Press [2nd] and use [▼] to move to the line containing the **Pause** command.
3. Press [DEL] to remove this line.
4. Press [2nd] [QUIT] to save your change.
5. Press [ENTER] to run the program and check your prediction.

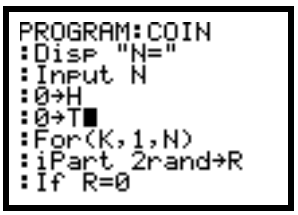
Students are instructed here to use the [DEL] key, but the [CLEAR] key will also clear the line. Later, they will delete program lines and the [CLEAR] key will not suffice.The students do not have to clear the program lines here, but can simply type over them.

If you wanted to toss a coin 600 times, you would probably choose a calculator to do the tossing. In fact, it would be nice if the calculator could also do the recording of the tallies so that you could spend your time analyzing and making sense of the results. Go back into the program and make a few more changes so that the calculator will also count how many heads and tails come up.

1. Press PRGM, select **EDIT**, and select the **COIN** program.
2. Use ▼ to move down to the third line **:For(K,1,N)** and then press 2nd [INS] ENTER ENTER to insert two blank lines.
3. Press ▲ twice to move back up to the first blank line. Press 0 STO▶ ALPHA H ENTER.
4. In the second blank line, type in 0 STO▶ ALPHA T.

   These commands set the values of the counters **H** and **T** to zero each time the program is run.

   Your calculator screen should look something like the one at the right.

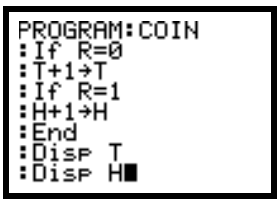


Note that when you want the calculator to display alphabetic characters on the screen, you need to enclose these characters inside quotation marks. For example, ***Disp "YOU"*** *will display the word* ***YOU****. However,* ***Disp YOU*** *will display the value of the product* ***Y*O*U****. If no values have yet been stored in any of these variables, the product will be* ***0****.*

5. Use ▼ to move down to the line **Disp "T"** and place your cursor on the D of **Disp**.
6. Press CLEAR ALPHA T + 1 STO▶ ALPHA T ENTER.

   This command means that every time R=0 (a tail is tossed), the value of T will be increased by 1. In other words, this line will count the number of times a tail is tossed.
7. Make a similar change to the line **Disp "H"** by pressing CLEAR ALPHA H + 1 STO▶ ALPHA H ENTER. You do not need the **Pause** command, so if you haven't deleted it yet, do so now by pressing DEL twice.
8. After the **End** command in the program, add the two new lines **Disp T** and **Disp H**. These commands will display or show the *values* of the variables (our counters) H and T instead of the letters themselves.

When finished, your display should look like the one at the right.



9. Press [2nd] [QUIT] to exit the program.

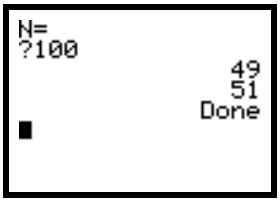
You are now ready to test the program to see if it will count the heads and tails.

1. To run the program again, press [PRGM], the number of the **COIN** program, and [ENTER]. Press 5, and you should see the counts for tails and heads displayed on the screen.

2. Press [ENTER] again to run the program one more time. This time enter in 100 tosses.

Notice that it takes the calculator a little bit of time to *toss* the coin. You can tell the calculator is working by watching the movement of the vertical segment in the upper right corner of the display.

A sample run of the program is given at the right. Your results will probably differ from ours.



✎ Go to the **Questions** section and answer #3 through #6.

Questions

1. What does the term *simulation* mean?

_____

_____

_____

_____

➧ *Return to page 51.*

2. What do you think would happen if you deleted the **Pause** command from the program?

_____

_____

➧ *Return to page 55, step 1.*

3. Does the calculator toss the coin faster than you could?

_____

_____

4. Are your results from 100 tosses close to what you would predict?

_____

The activity simulated the tossing of a coin using random numbers.

5. Describe the simulation you conducted in this activity.

_____

_____

_____

_____

The program allows one to conduct the simulation many times, quickly, and displays the results for analysis.

6. How did the program help with the simulation?

_____

_____

_____

Problems for Additional Exploration

1. We began our study of calculator simulations by starting with the command **iPart 2rand** to simulate the tossing of a coin. How could you modify this command to simulate rolling a standard six-faced die? Try out your suggestion on your calculator.

_____

_____

_____

*The roll of a standard die could be simulated by using **iPart 6rand+1**. Several rolls could be simulated by repeatedly pressing the ENTER key.*

The following program is one example that could be used to simulate several tosses of a standard die:

```
Disp "N="
Input N
For(K,1,N)
iPart 6rand+1→R
Disp R
Pause
End
```

2. Use some of the ideas you learned from the original **COIN** program to write a **DICE** program that will simulate rolling a standard six-faced die several times and display the outcomes from each of the rolls. It is not necessary that your program count and display the number of each outcome over all of the tosses.