Unit 4: Loops | **Skill Builder 3: The Loop loop**

In this lesson, you will learn about the all-purpose **Loop…EndLoop** structure.

Objectives:

- Use the **Loop…EndLoop** structure
- Use the **Exit** statement to get out of a loop

What is this thing called Loop…EndLoop?

The **Loop…EndLoop** structure creates a more flexible (but possibly hard-to-follow) loop. It repeatedly executes the statements in the loop body. Note that this loop will be executed endlessly, unless an **Exit** statement is executed somewhere inside the loop body.

If no **Exit** statement is encountered, then the loop will be an 'infinite' loop. If you accidentally run into an infinite loop on the handheld, press and hold [on] until the program 'breaks.'

The **Loop…EndLoop** structure is processed at least once since there's no condition to be met for its entry.

Exit forces program flow to the statement immediately after **EndLoop**.

Example: How many random numbers from 1 to 6 can be generated before two consecutive values are the same?

In the program shown to the right*, observe the use of **Loop…EndLoop** with a *conditional* **Exit** statement. When the random integer generator **randInt(1,6)** produces two consecutive identical values, the loop **Exit**s to process the **Disp** statement at the bottom of the program.

The reserved word **Exit** is located in **menu > Transfers**.

```
Define loop_endloop()=
Prgm
Local t,c,n
t:=randInt(1,6)
c:=1
Loop
n:=randInt(1,6)
If t=n Then
   Exit
Else
   t:=n
   c:=c+1
EndIf
EndLoop
Disp c
EndPrgm
```

The same effect could be accomplished with a **While** loop. Think about how you could do this.

** An elongated image was used here to show the entire program which would not fit in one screen.*

Program: Guess My Number

To demonstrate the use of **Loop…EndLoop**, we'll develop a two-player game to guess a random number from 1 through 10. When a player guesses the number, then the program ends with a 'winner' message. Sample output of such a program is shown to the right.

1. Start a new program called **guess**.
2. Start by setting up (initializing) the game. Identify three variables, the player number (*player*), the computer's or handheld's number (*number*), and the player's guess (*guessnum*).

The computer picks a random number from 1 to 10. We start with the player numbered 0. This makes it easy to change the player number as we'll soon see.

3. Next, build the loop where we first ask the player to enter a guess. The complete statement is:

   Request "Player " & string(*player*) & " guess?", *guessnum*

The '**&**' symbol is for *concatenation* of strings. The prompt portion of a **Request** statement must be a string so the *player* variable (a number) is converted to a string with the **string( )** function found in the Catalog.

4. Next, build the **Exit** condition. When a player guesses the number, then the loop will **Exit**. We can use the primitive **If** statement here:

   If guessnum = number
   Exit

Recall that **If** without **Then** processes the next statement when the *condition* is true; otherwise, the next statement is skipped.

These two statements can be placed on the same line by using a colon (**:**) to separate the statements: **If guessnum = number : Exit**

5. To switch players, use the statement:

   player:= 1 - *player*

This unique statement <u>switches</u> the value of *player* between 0 and 1. That is, when *player* is 0 the statement changes it to 1 and when *player* is 1 the statement changes it to 0. Try it!

6. Finally, add in a **Text** statement after the loop to congratulate the winner:

   Text “Player ” & string(player) & “ wins!!”

Tip: If you’d rather not see ‘player 0’ and ‘player 1,’ just add 1 to the player variable in this statement and in the **Request** statement. The user sees a 1 or a 2 even though the computer uses 0 and 1 for the players.

7. Before running the program, save the document in case the program becomes stuck in an infinite loop. Once the program begins, you must play until there’s a winner. There is no escaping the Request statement.

©2018 Texas Instruments Incorporated