CS 105

Introduction to Scientific Computing

Topic #11 – For Loops and Tracing

Matt Burlick
Stevens Institute of Technology

ASSIGNMENT 7

- Determine how many times the command *ires=ires+index2* is run and what the final value of *ires* is for the following script:
-
```
ires = 0;
for index1 = 10:-2:4
        for index2 = 2:2:index1
                if index2 == 6
                        break
                end
                ires = ires + index2;
        end
end
```

NEEDED SKILLS

- How to create loops in Matlab
- How to trace loops in Matlab

TOPICS

1. What are loops and when to use them?
2. Break and Continue Statements
3. Tracing Scripts

READING

- Section 4.2: The For Loop
- Robert Talbert's videos on loops are:
    - http://www.youtube.com/watch?v=5a3bpKuBpgo (for loops)
    - http://www.youtube.com/watch?v=O6vD-E3AZoo (while loops #1)
    - http://www.youtube.com/watch?v=LZY-MubpShg (while loops #2)

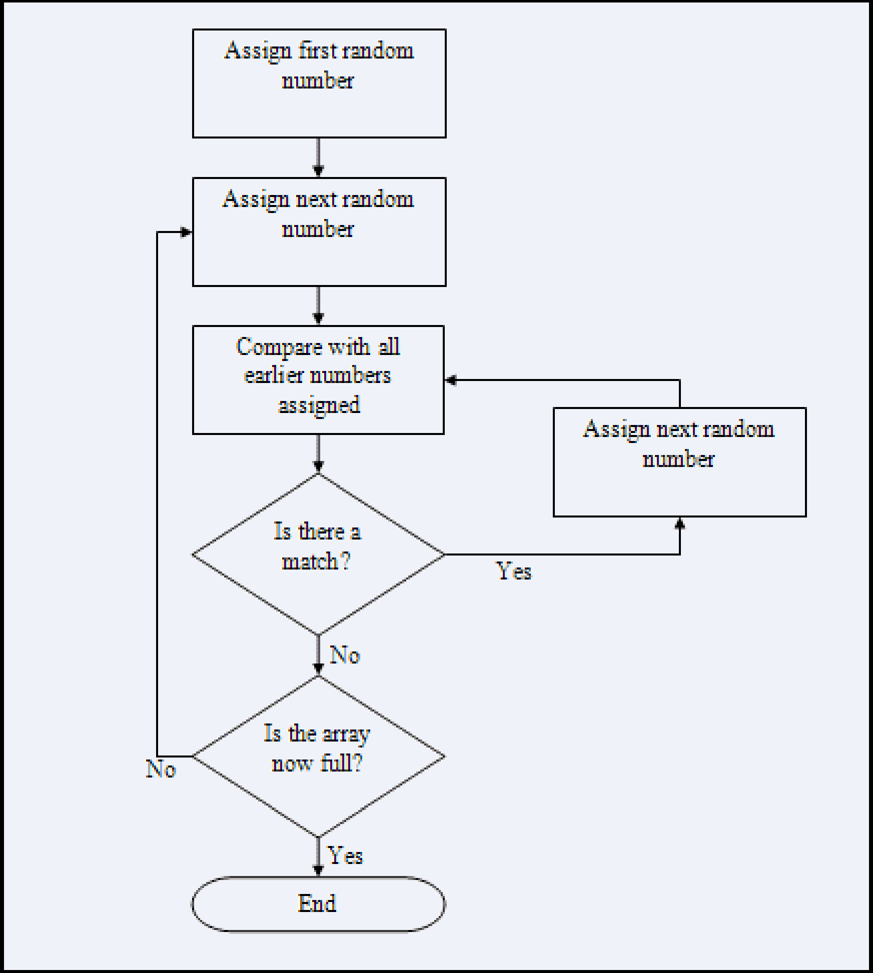
PROGRAM FLOW

- Thus far all of our program run every command *at most once*.
    - And they do them in order, from first to last
    - Order's important!
        - Must assign a variable before you use it
- But more sophisticated programs should be able to do stuff like
    - Only run certain commands
    - Run certain commands several times

PROGRAM FLOW

- Last chapter we introduced branching
  - This allows us to run only certain commands
  - But we still only run commands *at most* once
- What if we want to run commands several times
  - How many times?
    - Depends!
      - Conditions!

FLOW DIAGRAMS


Assign first random number
Assign next random number
Compare with all earlier numbers assigned
Assign next random number
Is there a match?
Yes
No
Is the array now full?
No
Yes
End

LOOPS

- There are 2 common types of loops
  - *for* loops
    - Do some block of code *for* a set of values
  - *while* loops
    - Do some block of code *while* a condition is true

SPECIAL BLOCK STATEMENTS

- We can force premature behavior in loops
- *continue*
  - Skip the rest of the body of the loop and do next *iteration*
- *break*
  - Get out of the block ASAP!
  - Works for *any* conditional block (including if/elseif/else statements)
- *return*
  - Exits scripts (or exits function)

FOR LOOPS

- Do a body/block *for* a set of values
- Useful when we know the set of values to do before hand
  - Like vectors and arrays!

-
  ```
  for i=[1 3 4]
          disp(i);
  end
  ```

-
  ```
  for i=1:10
          disp(i);
  end
  ```
- Example 1: Print out all odd numbers between 1 and 80

TRACING

- Tracing an program/script means to observe how variables change as the program is executed
- It is basically stepping through the program by hand
- It is useful when you program isn't producing the correct output.
  - It can help you figure out where things went wrong!
- It can also help with understanding *why* a program is working correctly

TRACING PRACTICE

- Trace the following:

-
  ```
  X=[4 3 6 9]
  Y = zeros(1,length(X)-1);
  for i=1:length(X)-1
          Y(i) = X(i) + X(i+1);
  end
  ```

FOR LOOPS: FINAL REMARKS

- Similar to "foreach" loops in other languages
- Good for repeating a process when:
  - We know the number of times to do it beforehand

    ```
    for i=1:n
            %stuff
     end
    ```
  - We know we want to do it for each element of an array/matrix

    ```
    for loc=1:length(X)
            %stuff
    end
    ```