CS1132 Lab Exercise 1

1 while Loops

Let’s recall the lecture video. When is it appropriate to use a loop? When should I use a `while` loop, and when should I use a `for` loop?

There are often three parts that need to work together for a loop to work properly:

1. A loop counter initialization - tells us where the loop counter starts
2. A loop condition - the loop runs while this is true
3. An update to the loop counter - otherwise, the loop runs forever, since we never move closer to making the loop condition false.

In the code below, label the three parts of the loops:

```matlab
i = 1;
while i <= 12
    disp('meow meow')
    i = i + 3;
end
```

For comparison, where are these three parts in the `for` loop? Note that its behavior is identical to the `while` loop above.

```matlab
for i = 1:3:12
    disp('meow meow')
end
```

Another construct is when you want to make sure that you do not go out of bounds of an array, or when you want to stop a loop early. Below is an example of a loop that does both:

```matlab
array = [1, 5, 3, 8, 5, 9, 4];
i = 1; % Note that an array indexes in MATLAB start at 1!
valueToFind = 8;
found = false;
while i <= length(array) && ~found % Unlike Java or Python, the last index is actually
    if array(i) == valueToFind % equal to the length of the array
        found = true;
    end
    i = i + 1;
end
fprintf('The value was found at index %d\n', (i-1));
```

And here is a better way to do it:

```matlab
array = [1, 5, 3, 8, 5, 9, 4];
i = 1;
valueToFind = 8;
while i <= length(array) && array(i) ~= valueToFind
    i = i + 1;
end
fprintf('The value was found at index %d\n', i);
```
2  fprintf

Back to the lecture material. What is the purpose of fprintf?

What type of value can be inserted / what formatting occurs when using the following substitution sequences?

- `%d ______________________ %5d ______________________`
- `%i ______________________ %-6i ______________________`
- `%f ______________________ %8.2f ______________________ %.2f ______________________`
- `%s ______________________ %20s ______________________`

Write a script that prompts the user for three integers using the built-in `input` function and then prints

- The sum
- The average
- The maximum (the largest of the three)

If any number should be a double, round to three decimal places. Make sure the numbers are properly aligned, as seen below:

Enter a number: 123
Enter a second number: 234
Enter a third number: 345

Sum: 702
Average: 234.000
Maximum: 345

Use format specifiers for both the string label and the number. For example, if I wanted to print the difference of subtracting the second and third numbers from the first one (i.e. num1 - num2 - num3), the code should be:

```matlab
fprintf('%-12s %13d
', 'Difference:', diff)
```

Notice that the string is left justified with a width of 12 characters, and the integer is right justified with a width of 13 characters.

Hint: For the average formatting, consider how many extra characters you need to print in order to get the character spacing correct.

3  MATLAB fun . . . functions

Type each of the following expressions in the MATLAB Command Window (and press Enter) to see what it does. Write the screen output (and answer the question) on each blank.

```matlab
a = mod(7,2)  %______ Function rem is similar; to see how they differ type the following
help mod  % quick function reference
help rem
doc rem  % detailed function documentation

b = (4*2) ...
+ 1  % What does ... do? ______________________ It’s called the ellipsis
```
4 Different ways to create vectors

\[
a= \text{zeros}(1,3) \quad \%_____________
\]

What do the arguments specify?

\[
b= \text{ones}(3,1) \quad \%_____________
\]

\[
c= \text{rand}(1,4) \quad \%_____________
\]

\[
d= 10:2:17 \quad \%_____________
\]

\[
f= 10:-1:7 \quad \%_____________
\]

\[
g= [10 \ 20 \ 40] \quad \%_____________
\]

What does the space separator do?

\[
h= [10,20,40] \quad \%_____________
\]

What does the comma separator do?

\[
k= [10;20;40] \quad \%_____________
\]

What does the semi-colon separator do?

\[
m= [a \ g] \quad \%_____________
\]

\[
n= [b; k] \quad \%_____________
\]

\[
p= [a \ k] \quad \%ERROR--mismatched dimensions! (Attempt to concatenate a column to a row)
\]

\[
q= b' \quad \%_____________
\]

This operation is called "transpose"

\[
r= [a \ b'] \quad \%_____________
\]

\[
s= \text{sum}(r) \quad \%_____________
\]

What does function sum do?

5 Multiples of \( k \)

Complete the program so that it reads a positive integer \( k \) and outputs all the multiples of \( k \) up to 1000.

\[
k = \text{input}('Enter a positive integer smaller than 1000: '); 
\]

\[
\text{for } j = \______________ \\
\quad \text{fprintf}('%d ', j); \\
\text{end} \\
\text{fprintf}('
');
\]

6 Examining a subarray

Write a function \texttt{vectorQuery(v,n,r)} to determine whether the number \( r \) appears in the first \( n \) components of vector \( v \). The function returns 1 if \( r \) is in the first \( n \) components of \( v \) and 0 otherwise. Your function assumes that \( v \) is a vector of numbers, \( n \) is a positive integer, and \( r \) is a number. Use a loop to do the search and make sure that the loop index doesn’t go “out of bounds” (if \( n \) is greater than the length of vector \( v \)).

7 A cow in a sheep field

Given an array where 1’s represent sheep and 0 represents a cow, create a function \texttt{findCow(animalArray)} to find a cow in the sheep field. Return the index into the array where the first cow is found. Make sure to use the \texttt{length} function so that you can have fields with various numbers of sheep (i.e. the length of the array can change).
8 Creating arrays of unknown length

Write a function `sequence(m)` that generates a sequence of random *integer* numbers between 1 and \( m \), inclusive, stopping when a value is repeated for the first time. The function returns an array containing all the numbers generated (in the order in which they were generated) except for the last value that is a repeated occurrence.

Example: If the generated sequence is 3 1 9 5 7 2 5, the array to be returned should be 3 1 9 5 7 2.

*Hints:* 1) Use the function `vectorQuery` that you have developed already. 2) When you don’t know how long a vector needs to be, you can build it one component at a time. Here is an example to store only the even integer values that a user enters:

```matlab
% Prompt user to enter positive numbers and store the even integers in a vector v
k = 0; % vector length so far
num = input('Enter a positive number: ');
while num > 0
    if rem(num, 2) == 0
        k = k + 1;
        v(k) = num;
    end
    num = input('Enter a positive number (negative to stop): ');
end
```

In a computer lab? Delete your files from the computer before you leave.