Week 7 – Functions

0 Mid-Semester Survey
We value your feedback and we read every comment you write (they are anonymous). If you have not yet done it, please do the mid-semester survey. See last week’s lab, Exercise 1.

1 More Q’s on Functions and Basic Logic

1.1 Factorial
Write a function that takes an integer parameter (you may assume the given value is always an integer) and returns the factorial of that number. The factorial of a number \( n \) is written as:

\[ n! = n \times (n-1) \times (n-2) \times \ldots \times 2 \times 1 \]

For example, the factorial of 4 is:

\[ 4! = 4 \times 3 \times 2 \times 1 = 24 \]

Note that \( 0! = 1 \)

It only makes sense to calculate the factorial of an integer that is non-negative, so you should use the `error()` function when a negative value is passed as a parameter.

**Hint:** This exercise is similar to Lab 6, Exercise 2.2. You really should be able to do this Q without the tutor’s help.

1.2 Using the mod function
Copy the sample solution of the function `isPrime` from last week’s lab (Lab 6), Exercise 4.3, the question where you had to write a function to check if an integer is a prime number or not.

Inside this function, it uses another function `isDivisible`, which checks to see if one integer is divisible by another integer.

Now, rewrite the `isPrime` function so that you do not use the `isDivisible` function (therefore you should not copy the `isDivisible` function for this exercise), but instead, use the `mod` function which we learnt in the lectures this week.

Recall from the lectures that to use a function, you only need to know what it does, you don’t need to know how it does it. So how many lines of code should you have to change in the `isPrime` function? Do you ever need to see the code in the `mod` function?
2 More Q’s on Arrays and Matrices

2.1 Understand the difference between index and element at that index
This question was recently used in Lab Exam 2

At the start of your program, write code that creates a 1-D array containing integers.

Then, write code that loops through the array and only prints those elements in the array that are at an index (position) in the array that is the same as the element value. Print these numbers on top of each other, on separate lines.

For example, if your array contained the integers 3, 2, 4, 9, 5, 0 -7, 9 (in this order), then your program should print 2 and 5, since the element 2 appears at position 2 in the array, and the element 5 appears at position 5 in the array.

If your program does not print any numbers, then you should print an appropriate message.

Your program should work, regardless of what integers (if any) were put into the array at the start of the program.

Note: This Q uses material that has been learned in weeks 4, 5 and 6. Remember, you must know the material in weeks 4-7 to pass this course, so you should be able to do this Q if you want to pass. You will face Q’s like this in lab exam 2.

2.2 Swapping elements in arrays
This question was recently used in the Final Exam

Write a function called swapPairs which has one parameter: an array containing integers (you may assume that this array does not contain any other data types besides integers).

Your function should swap each value in the array that is in an odd-numbered index with the value in the even-numbered index immediately following (if it exists), and return this array with the new ordering of elements.

For example, if the given array contains the numbers 5, 6, 3, 9, 8 and 6 (in this order), then your function should return an array containing the numbers 6, 5, 9, 3, 6 and 8 (in this order), because the numbers at indices 1 and 2 were swapped (same with 3 and 4, and also 5 and 6).

If a value at an odd-numbered index has no next element, then it should remain at its position.

Note: This Q uses material that has been learned in weeks 4, 5 and 6. Remember, you must know the material in weeks 4-7 to pass this course, so you should be able to do this Q if you want to pass. You will face Q’s like this in the final exam.
2.3 Summing the diagonals of a matrix

Write a function that takes a matrix as a parameter, and returns the sum of all the values on the long diagonals of the matrix. Note that if there is a central number that appears on both diagonals, this number should not be added twice.

For example, if your function is given the following matrix:

```
1 2 3
4 5 6
7 8 9
```

your function should return 25, since $1 + 5 + 9 + 3 + 7$ is equal to 25.

If the given matrix is not a square matrix, then your function should return 0.

3 Printing Patterns

3.1 Printing a cross

Write a function that takes an integer parameter and prints out a cross of *’s, where the width and height of the cross is the given integer parameter.

For example, if your function was given the value 5, then your function should print:

```
*  
* **
*****
** *
*  
```

However, your function should print nothing if the given parameter value is an even number.

Hints:

- An even number is a number that is divisible by 2. That means, when you divide an even number by 2, the remainder is 0. Use lecture 7-1, slide 24-25 to help you.
- This exercise is really just an extension of Lab 5, Exercises 5.4 and 5.5.

You are strongly recommended to look at the sample solution for this Q, which will also be demonstrated by your tutor. Although you may have written the code to work 100% correctly, it may not be as simple as the sample solution, which will be easy to change for whatever pattern you may have to print in the lab exam. Remember that comments and coding style (which includes simplicity and flexibility) will be worth 20% of the marks in Lab Exams 2-3.
3.2 Printing a tricky triangle

Write a function that takes a parameter and prints a triangle on the screen whose width and height is the given parameter. The numbers printed on each row count from the number of digits on that row down to 1, and the triangle is right-aligned (there is a straight vertical edge on the right side of the triangle).

For example, if your function was given the parameter value 4, then you would print the following triangle on the screen:

```
4 3 2 1
3 2 1
2 1
1
```

Your function should print nothing if the given parameter is greater than or equal to 10.

Hints

- This exercise is similar to Lab 5, Exercise 5.5.
- In lecture 7-1, slide 41, it is shown that the `error` function is used in the same way as the `fprintf` function (introduced in Lab 5, Exercise 5.4). Lecture 7-1, slides 42-43 show how you can print out the value of a variable using the `error` function, so you can do the same thing with the `fprintf` function.

4 A deeper understanding

4.1 What's wrong with this solution?

This question is based on a Q that was recently used in the Final Exam

Note: In the final exam, you will be asked to explain some things. It is only possible to get these Q’s correct by having a deep understanding of the material in the lectures and labs. This exercise gives you a relatively simple example of such a Q asking you to explain something.

Consider again the question from last week’s lab: Lab 6, Exercise 6.1.

Using the material from this week’s lecture, you should now be able to give the third reason.

Write this reason down as an additional comment into the code that you used last week.
5 Extra Q’s on matrices

5.1 Checking outer edges
Write a function that is given a matrix as a parameter (you may assume no other data types will be given), and returns a logical value.

The function should return true if any of the elements on the outer sides (top, bottom, left or right side) of the matrix is a negative number. Otherwise, the function returns false.

If the matrix given in the parameter is empty, then your function should return false.

Your solution must use nested loops (one loop inside another loop) and visit every element in the matrix.

5.2 Shifting columns
This question was recently used in the Final Exam

Write a function that is given a matrix of integers as a parameter, and returns a different matrix of integers.

The returned matrix of integers should contain the same elements as the matrix given in the parameter, except that all elements have been shifted 2 columns left. The first and second columns in the matrix given in the parameter should be the second-last and last columns in the returned matrix respectively.

For example, if your function is given the following matrix:

```
1  2  3  4  5
6  7  8  9 10
11 12 13 14 15
```

then your function should return this matrix:

```
3  4  5  1  2
8  9 10  6  7
13 14 15 11 12
```

If the matrix in the given parameter has less than 3 columns, then your function should return an empty matrix.

Your solution must use nested loops (one loop inside another loop) and visit every element in the matrix.
5.3 Checking for adjacent elements

This question was recently used in Lab Exam 2

Write a function that is given a matrix of integers as a parameter (you may assume no other data types will be given), and returns a logical value.

The function should return true if, to the immediate right of every zero in the matrix (i.e. next to the zero, on the right side), there is either a positive number, or there is nothing there at all (because the zero appears on the right edge of the matrix). Otherwise, the function returns false.

If the matrix is empty, or the matrix has no zeroes, then your function should return true.

Your solution must use nested loops (one loop inside another loop) and visit every element in the matrix.

6 Use the Additional Resources

To prepare for the lab exam (and also the final exam), make sure to do the following:

- Do all the questions from labs 4-7. If you struggled, consider doing some of the questions again on your own, if you already did it with your tutor. You learn a lot, even when you are doing the same question again. As you do it, do not look at the sample solutions; you should only look at sample solutions of labs previous to that lab that you are doing.
- Download the sample files that come with the lectures, and play around with them. Go through the same exercises that were covered in the lectures. These exercises are different to the ones in the labs, and cover different material that may be found in Lab Exam 2.
- Download the sample solutions for the lab exercises, and compare them with your solutions. Make sure you save them in a different folder to where you saved your work, so that you do not replace your work. The sample solutions are written as simply as possible, with clear comments and neat layout (including good line spacing and indentation), which is what you will need to do to score marks in the lab exam.
- If anything is unclear, ask your tutor for help.
- You can also watch the lecture recordings for further clarification.

Please open all your exercises before calling the tutor to get marked off and leave early.