

Lab 0 – Operating Systems

Name(s) _____

The purpose of this lab is to get you familiar with the Linux and basic C Systems Programming with Linux. This lab can be completed alone or with a partner. The purpose of this lab is to prepare you to be able to do Project 0. This lab is worth 5% of your class score. The lab itself is worth 15 points.

Part A – Transferring, Compiling, and Running a C Program on a Linux Server (5 points)

Please watch the short Videos 1, 2, and 3 on the class Operating Systems “Resources” page and follow along with your own laptops/machines.

Please initial below when you have watched these videos and successfully logged onto the Linux server, transferred a file to the server, compiled a C program, and ran a C program. Let me know if you have any questions on this process before moving on.

Initials _____

Part B – Getting Familiar with C (10 points)

We are going to learn the basics of C for this class. I recognize that some of you are more familiar with C and/or Linux than others. If you are more experienced and finish the lab quickly, you can take some time to further hone your skills by completing some extra credit activities listed on the main course website.

Let’s use a nice interactive website source to learn C basics. You will be able to look at materials and try coding solutions entirely in this webpage. However, any of the code you try in this tutorial will also work if you copy it to the Linux server, compile, and run it there.

Open up a web browser and go to <http://www.learn-c.org/>. Your ads may vary. (They are a free resource, so they need to make money.) Click on the “Hello, World!” link to start that module.



Table of Contents

Learn the Basics

- [Hello, World!](#)
- [Variables and Types](#)
- [Arrays](#)
- [Multidimensional Arrays](#)
- [Conditions](#)

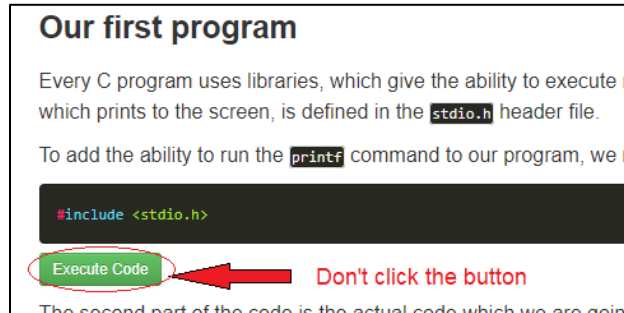
Code ▶ Run Reset Solution Output

```
1 /* Welcome to the Interactive C Tutorial.
2 Start by choosing a chapter and
3 write your code in this window. */
4
5 #include <stdio.h>
```

Once you start each module under “Learn the Basics”, you will be able to read information about that C topic. Then you’ll complete the exercise at the bottom of the page in the black code areas. The “Run” button runs the code. “Reset” works to reset the code, and “Solution” will show you the solution if you are stuck.

Module #1: Hello, World!

Click on the “Hello, World!” exercise link and read through the text. Notice that as you read through the text, you’ll see code examples. Under each code example is a green button “Execute Code”. Depending on the situation, this button might not be very useful, because it tries to run the little snippet of code. If the little snippet of code isn’t a full program, you will just see a syntax error. In the figure below, clicking the “Execute Code” button won’t run a very interesting program, because this line only includes (brings in) a library, but no code actually runs. If it looks like a full program, though, feel free to click the button to see what happens!



Our first program

Every C program uses libraries, which give the ability to execute n... which prints to the screen, is defined in the `stdio.h` header file.

To add the ability to run the `printf` command to our program, we n...

```
#include <stdio.h>
```

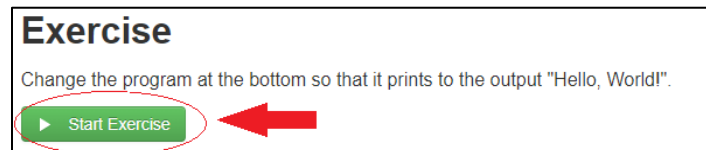
Execute Code **Don't click the button**

The second part of the code is the actual code which we are going to...

Answer the following questions:

1. What is the name of the function used to print things to the screen?
2. When returning the number 0, what does that signify?

After you’ve finished reading through the module text, click on the green “Start Exercise” button and complete the exercise at the end of the page.

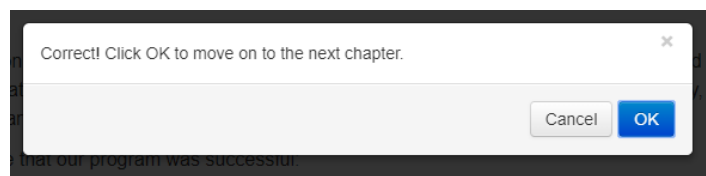


Exercise

Change the program at the bottom so that it prints to the output "Hello, World!".

▶ Start Exercise

Clicking that button will expand the exercise directions. Edit directly in the black code area. Once you think you have the solution, click the “Run” button to check it. (Hint, click the blue “Solution” button if you get stuck.) Otherwise, raise your hands for help. Once you have it correct, this window will pop up:



Correct! Click OK to move on to the next chapter.

Cancel OK

that our program was successful

If you understand the solution, initial here. Then click OK to move to the next chapter.

Initials _____

Module #2: Variables and Types

C is different than Python (but similar to Java) in that you can't just use a variable without *defining* it. That means we have to list the variable name and its type (like int or float) to let our compiler know that we plan to use those variables. This section will give us practice with that.

When we print, we also use placeholders to allow us to print variables. Here is a table of common placeholders used:

Placeholder	Type of variable
%d or %i	Integer
%f	Float
%c	Character (not a string, just a single character)
%s	String

Please read through the Variables and Types instructional material.

Click on the green "Start Exercise" button and complete the exercise. If you understand the solution, initial here. Otherwise, raise your hands for help. Once you have it correct, click OK to move to the next chapter.

Initials _____

Module #3: Arrays

Arrays are similar to lists in Python in that they hold multiple elements, but unlike Python, each element has to be of the same type. Therefore, if you want an array (list) of integers, you would need to define the array before you start to use it. Unlike Python, you also have to know the size of the array ahead of time when you define it. (In systems programming, it is very important to know exactly how much memory we are going to use before we use it.) However, just like Python, we use indexes to access different locations in the array.

Please read through the Arrays instructional material.

Click on the green "Start Exercise" button and complete the exercise. (The blue Solution button does not seem to work for this lab, unfortunately.) If you understand the solution, initial here. Otherwise, raise your hands for help. Once you have it correct, click OK.

Initials _____

[STOP!] Note: The text takes you to multidimensional arrays here, but I would rather you not do that quite yet! Go back to the main webpage: www.learn-c.org. Click on the Conditions link. We'll come back to multidimensional arrays, don't worry.

Module #4: Conditions

Conditional statements in C are similar to those in Python, but instead of using **tabs** to define the condition suite, we use curly braces { and }.

Please read through the Conditions instructional material and answer the following questions:

3. Instead of using the words "and" and "or" for compound Boolean statements, what symbols do we use in C instead?
4. Are parenthesis necessary around the Boolean conditional statement? Yes or no?

Click on the green "Start Exercise" button and complete the exercise. If you understand the solution, initial here. Otherwise, raise your hands for help. Once you have it correct, click OK. (Hint: Don't forget about \n for the newlines! C doesn't add those automatically like Python.)

Initials _____

Module #5: Strings

Strings are a little bit more complicated in C than they are in Python. We cannot compare two strings using ==, and we cannot concatenate two strings together using +. The reason is because of pointers. However, we can use other specialized **string functions** to do comparisons or string concatenation.

Please read through the Strings instructional material and answer the following questions:

5. What function do we use to compare two strings (to see if they are equal)?
6. What function do we use to get the length of a string?
7. What function do we use to concatenate two strings (e.g. put two strings together)?

Click on the green “Start Exercise” button and complete the exercise. If you understand the solution, initial here. Otherwise, raise your hands for help. Once you have it correct, click OK.

Initials _____

Module #6: For Loops

The for loop notation looks very unlike Python, but it is pretty similar to Java. Just like with the if statements, we need to put curly braces {} around what goes in the loop.

Please read through the Strings instructional material and answer the following questions:

Click on the green “Start Exercise” button and complete the exercise. If you understand the solution, initial here. Otherwise, raise your hands for help. Once you have it correct, click OK.

Initials _____

Module #7: While Loops

While loops in C are pretty similar to Python. We can also use the **continue** directive to continue on with the next iteration of the loop, and we can use the **break** directive to stop and “break out of” the loop to continue onto the rest of the code.

Please read through the Strings instructional material and answer the following questions:

Click on the green “Start Exercise” button and complete the exercise. (Hint: Don’t forget to increment the variable i in the loop!) If you understand the solution, initial here. Otherwise, raise your hands for help. Once you have it correct, click OK.

Initials _____

Module #8: Functions

Please read through the Strings instructional material and answer the following questions:

8. What does it mean to create a function with the void keyword?

Click on the green “Start Exercise” button and complete the exercise.

(Hint: The directions for this exercise could use some improvement. Don’t print exactly “x is big”. Print the variable “number” that is passed into the function instead of x.)

If you understand the solution, initial here. Otherwise, raise your hands for help. Once you have it correct, click OK.

Initials _____

Module #9: Static

Please read through the Static instructional material. (Clicking the Execute Code buttons in this reading is pretty useful to see what's going on.)

Click on the green "Start Exercise" button and complete the exercise. If you understand the solution, initial here. Otherwise, raise your hands for help. Once you have it correct, click OK.

Initials _____

Module #10: Multidimensional Arrays

Ok, now let's go back to the home page and go back to multidimensional arrays. Now that we've seen for loops in C, the exercise will make sense.

In Python, we've seen multidimensional arrays when we talk about lists of lists. Indexing into a C multidimensional array works just the same—index going left to right.

An multidimensional array defined like this in C is just 3 arrays of 4 elements each:

```
int a[3][4] = {
    {0, 1, 2, 3} , /* initializers for row indexed by 0 */
    {4, 5, 6, 7} , /* initializers for row indexed by 1 */
    {8, 9, 10, 11} /* initializers for row indexed by 2 */
};
```

It looks like this in Python:

```
a = [[ 0, 1, 2, 3], [4, 5, 6, 7], [8, 9, 10, 11]]
```

If we used some line breaks in Python, it looks a little more like the C example:

```
a = [[ 0, 1, 2, 3],
     [4, 5, 6, 7],
     [8, 9, 10, 11]]
```

Please read through the Multidimensional Arrays instructional material.

Click on the green “Start Exercise” button and complete the exercise. If you understand the solution, initial here. Otherwise, raise your hands for help.

Initials _____

Module #11: Pointers

This is the only module I need you to complete in the “Advanced” section. We won’t need to do a lot with pointers in this class, but they do show up periodically. All they are is just a memory address number (pointer) to where your string, array, or object (struct) is stored. Pointers are usually represented as an integer or a long integer in the system.

To understand the basics about pointers, we need to understand and use the * and & symbols.

Please go back to the home page, click on the Pointers link, and read through the Pointers instructional material and answer the following questions:

Click on the green “Start Exercise” button and complete the exercise. (Don’t feel bad if you need to check out the blue “Solutions” button here.) If you understand the solution, initial here. Otherwise, raise your hands for help.

Initials _____

Congratulations! You are done with the lab! Please remember to turn this lab in to Dr. Diesburg to get your lab points. Now you can get started on Project 0, which must be completed individually.