



California State University, Sacramento
College of Engineering and Computer Science

Computer Science 35: Introduction to Computer Architecture

Fall 2021 – Lab 1 – *Hello World*

Overview

It is finally time for your first lab. Woo hoo!

Anyway, it is a long-standing tradition in computer science that your first program displays “Hello World” to the screen. This dates back to the first message sent over ARPANET – the predecessor to the Internet.

This week, you will basically get your feet wet with assembly programming. On the next page of this handout, there is a very basic “Hello World” program. Essentially, your only task for this lab is to print the traditional “Hello World!” to the screen followed by some other information.



Part 1. Connecting to the Server

Step 1 – Connecting to the VPN

To get access to the server, which we will use for our labs, you must connect to the GlobalProtect VPN. There are instructions on our website.

Step 2 – For Windows

If you are using Windows, you need to download and install a copy of MobaXterm or PuTTY. Once you have installed it, open the application, and connect to the following address using SSH (Secure Shell).

`ecs-pa-coding1.ecs.csus.edu`

Step 2 – For Macintosh

Open the Terminal program. This is the same UNIX prompt that you get when you connect to Athena. Mac-OS X is a version of UNIX. Neat! Once at the prompt, type the following where **username** is your ECS username. You might have to manually type “yes”.

`ssh username@ecs-pa-coding1.ecs.csus.edu`

Part 2. Getting the Course Library (csc35.o)

This course uses a library object file. The library contains a large number of utility functions that will allow you to easily print integers, strings, and other useful tasks. But, to use the library, you first need to obtain it.

You need to use a UNIX command called "curl" (Copy from URL). Please type the following at the command-line and press the Enter Key. It will download the library from my website and save it to your account.

```
curl devincook.com/csc/35/csc35.o > csc35.o
```

UNIX will display information during the download. You can ignore this. It means it worked correctly.

```
[dcook@ecs-pa-coding1 ~]$ curl devincook.com/csc/35/csc35.o > csc35.o
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           %             %         Dload  Upload   Total   Spent    Left   Speed
100  6624  100  6624    0     0   5434      0  0:00:01  0:00:01 --:--:--  5433
```

You can check that the file downloaded by typing **ls** and pressing the Enter Key. You should see the library file listed.

Part 3. Creating / Editing a File

Now that you have the library, it's time to write your first assembly program. First, you need to create your source code. In UNIX, there are a number of excellent text editors. We are going to use an application called *Nano* – which is easy-to-use and quite user-friendly.

To create a new file (or edit an existing file), type "nano" followed by the filename and press the Enter Key.

```
nano lab1.asm
```

Once it opens, notice that all the options are listed on the bottom of the screen. In UNIX, the carrot symbol **^** represents pressing *and holding* the **Control** key. So, for example, to exit your program you will press **Control** and **X**.

The following is the solution for the classic "Hello, World" program. This solution makes use of the CSC 35 library. Please type it verbatim into the text editor. Pay very close attention where there are spaces.

You don't have to type the comments, but it might be good idea. You need to understand what each line does.

```

# lab1.s
# YOUR NAME HERE
#
# 1. Assemble : as -o lab1.o lab1.asm
# 2. Link      : ld -o a.out lab1.o csc35.o
# 3. Execute   : ./a.out

.intel_syntax noprefix           #Use Intel formatting
.data                           #Start the data section
Greeting:                       #Message is an address
    .ascii "Hello, world!\n\0"  #Create a buffer of ASCII

.text                           #Start the text section
.global _start                  #Make the _start label public

_start:                         #UNIX starts here
    lea rax, Greeting           #Put address into rax
    call PrintCString           #Execute the csc35.o subroutine

    call EndProgram             #Execute the csc35.o subroutine

```

Once you have typed out the program, press **Control** and **X** to exit Nano. Remember to select "yes" to save the buffer (that's the memory which is storing the text of your program).

Part 4. Assemble the Program

If you type **ls** and press Enter, you should be able to see your new file listed. Now that it is created, let's assemble it into an object file. Type the following and, then, press the Enter Key. Don't forget the **-o**. It specifies that the filename listed – after it – is the output.

Warning: If you list your **.asm** file after the **-o**, it will be destroyed.

```
as -o lab1.o lab1.asm
```

If you have any typos in your program, the assembler will list the errors. If that happens, open your existing program (type **nano lab1.asm**) and fix them.

Part 5. Link the Objects

If you didn't receive any error messages from Step 3, then the object file was created. Now let's link your object file to the CSC 35 library and create a program. Let's use a default program name of "a.out".

Type the following. Don't forget to type **a.out** after **-o**.

```
ld -o a.out lab1.o csc35.o
```

If you have any mistakes with your labels, you will receive an error. Often, this is the result of a simple typo.

Part 6. Execute

You can execute your new program by simply typing its name and pressing the Enter Key.

```
./a.out
```

Requirements

Now work on each of the requirements below one at a time. You will turn in the final program, but incremental design is best for labs. You must think of a solution on your own. The requirements are as follows:

1. Put your name and section # in a comment at the very top of your program. We are using class accounts, and **I need to be able to identify you.** (5 points)
2. Print "hello world" to the screen. You can make it "howdy" or any type of greeting you like. This was pretty much done for you already in the sample code. (5 points)
3. Print the text "My name is" and your full name to the screen. (5 points)
Do **not** use the same string as in #1. Create a new label and ASCII text.
4. Print a quote from someone. It can be funny or inspirational. (5 points)
Do **not** use the same string as in #1 or #2. Create a new label and ASCII text.
5. Print off a historical year that is interesting to you. For example: "In the year 1947 Sacramento State was founded!" The sentence must have text before and after the year. (10 points)

Here's the tricky part: the year must be printed using the **PrintInt** subroutine. So, put the immediate value (the year) into **rax**. Make sure to use **mov** rather than **lea**.

There is no concatenation in assembly. For the final sentence, you will need three prints – two PrintCString and one PrintInt.

Submitting Your Lab



This activity may only be submitted in Intel Format.

Using AT&T format will result in a zero. Any work from a prior semester will receive a zero.

Afterwards, run Alpine by typing the following and, then, enter your username and password.

```
alpine
```

Please send an e-mail to yourself (on your Outlook, Google account) to check if Alpine is working. To submit your lab, send the assembly file (not a.out or the object file) to:

```
dcook@csus.edu
```

UNIX Commands

Editing

Action	Command	Notes
Edit File	nano <i>filename</i>	"Nano" is an easy to use text editor.
E-Mail	alpine	"Alpine" is text-based e-mail application. You will e-mail your assignments it.
Assemble File	as -o <i>object source</i>	Don't mix up the <i>objectfile</i> and <i>asmfile</i> fields. It will destroy your program!
Link File	ld -o <i>exe object(s)</i>	Link and create an executable file from one (or more) object files

Folder Navigation

Action	Command	Description
Change current folder	cd <i>foldername</i>	"Changes Directory"
Go to parent folder	cd ..	Think of it as the "back button".
Show current folder	pwd	Gives the current a file path
List files	ls	Lists the files in current directory.

File Organization

Action	Command	Description
Create folder	mkdir <i>foldername</i>	Folders are called directories in UNIX.
Copy file	cp <i>oldfile newfile</i>	Make a copy of an existing file
Move file	mv <i>filename foldername</i>	Moves a file to a destination folder
Rename file	mv <i>oldname newname</i>	Note: same command as "move".
Delete file	rm <i>filename</i>	Remove (delete) a file. There is no undo.