# Homework 2 for CSC246

Homework 2 consists of 3 questions. Respond to each question and submit your work electronically using Moodle. Please submit your non-program answers in pdf, doc(x), rtf, or plain text. For the short programming questions, just submit the source file. The only comments that are required are comments to explain the compile command you used and the comments specified in the question. You must write the programs in one of the languages specified.

Make sure that your programs work on the school's eos or linux servers. **Include compile commands as comments within your program. Specify what environment you ran it on (e.g. VCL or Linux lab).** Contact the TA if you need help accessing the school's eos or linux environment remotely. The reason for this requirement is that it is important to make sure a program works for the environment for which it is intended. For purposes of this assignment, the intended environment is the N.C.State's linux environment.

This requirement is especially important if you are writing source code in Java. If you are using the latest version of Java in Eclipse, for example, there may be options that are not supported by a Java environment that may be loaded with the school's servers.   
If you are using your eos account from a remote location to do java compilations, please use the Virtual Computing Lab. More information about the VCL can be found at [http://vcl.ncsu.edu](http://vcl.ncsu.edu/).

If you need help, please contact the TA at once and arrange to get help. The TA's email address is on the syllabus.  
  
This homework is due on the date mentioned on the main website **by midnight.** This homework contains 35 points and is 3.5% of your total course grade.

*10 points*

## Question 1: Process Creation

In your own words, explain the work that goes on when a Unix process executes a **fork()** system call. (4 pts)

The fork() system call created a new process called child process that inherits the address space of the parent process. The child process duplicates the parent process that is currently running and processes a different pid. The return value of fork() system call is the child process’ pid and will be store in the parent’s variable. 0 will return to the child process. If no child process was created the system call return negative value.

In your own words, explain the work done when a Unix process invokes the **exec** system call. (4 pts)

The exec() system call will replace the original process with a new program and runs it.

In your own words, explain the work that goes on when a Windows XP application calls the **CreateProcess** routine. (2 pts)

The CreateProcess() in Windows XP application also creates a new child process. However, the new child process does not inherit the space address from the parent process. It requires to load a program to the child process’ address space. It also requires many parameters when calling.

*10 points*

## Question 2 What is your name?

Write a program using the c language and Unix/Linux to demonstrate your understanding of the Unix system calls for creating and synchronizing processes.  
  
The program must display your last name and then your first name.   
  
Your first name must be displayed by a parent process and your last name must be displayed by the child process.

Synchronize the 2 processes using the exit/wait system calls so that the output is guaranteed to be formatted as*:*

*Last, First*   
  
  
Explain (via comments) what is happening at each step in your code.   
  
  
Call this program **name.c**. If you do not have access to a Unix/Linux box, use the Virtual Computing Lab (http://vcl.ncsu.edu) to test your program.

*15 points*

## Question 3 What Goes Where?

Trace through the following code and (1) give the output that it generates, and (2) **designate where the data resides** that is being displayed. Use the following designations.  
  
**A** (child process's stack)  
**B** (child process's data region)  
**C** (parent process's stack)  
**D** (parent process's data region)

The printf statement displays the text provided in the quoted string with the integer value of the 2nd argument inserted for the %d format specification. Assume that the printf will print on a new line each time it is encountered. Indicate where the data resides (A, B, C, or D) by each output statement.

#include <stdio.h>   
  
int global = 5;   
  
void foo();   
  
main() {   
  
    int id = -2;   
    printf("main(): global = %d\n" , global) ;   
    foo();   
    foo();   
  
    id = fork();   
  
    if (id != 0) {   
       global++;   
       wait(NULL);   
    }   
  
   global++;   
   printf( "main(): global = %d\n" , global) ;   
      
}   
  
void foo() {   
  
    static int staticInt = 3;   
    int localInt = 1;   
    printf("foo(): staticInt = %d \n" , staticInt) ;   
    printf("foo(): localInt = %d \n" , localInt) ;   
    staticInt++;   
    localInt++;   
  
}

Output :

main(): global = 5 - D

foo(): staticInt = 3 - D

foo(): localInt = 1 - C

foo(): staticInt = 4 - D

foo(): localInt = 1 - C

main(): global = 6 - B

main(): global = 7 - D

To check your work, and especially if you are unfamiliar with C, you may store the source code in a file named **homework2.c**.

Compile it using this command:   
**gcc homework2.c**

Execute the program using this command:   
**./a.out**

Output will appear on the display.