EE 3233 System Programming for Engineers - Summer 2025

Exam 1

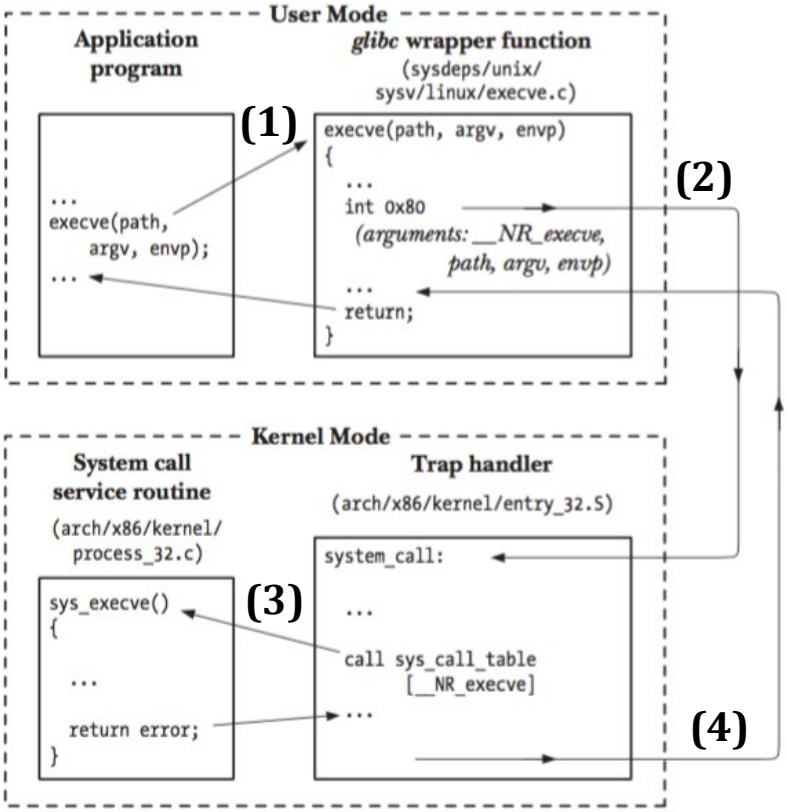
(Monday, June 16)

Name: Arnav Gupta Score: /120

I. Multiple Choice (Each 10 points)

1. Choose one, which is NOT a task performed by the kernel.
   - ☐ a. Memory management
   - ☐ b. Creation and termination of process
   - ☑ c. Compilation of program
   - ☐ d. Provision of a file system

2. Following figure shows the steps in the execution of a system call, *execve( )*. In which step the actual *execve( )* is executed?
   a. (1) ☐ b. (2) ☐ c. (3) ✔ d. (4) ☐



3. Which statement about the clearenv() function is true?
   - ☐ a. It retrieves the value of all environment variables.
   - ☐ b. It adds a new environment variable.
   - ☐ c. It removes a specific environment variable.
   - ☑ d. It erases all environment variables.

4. Choose an INCORRECT statement about the memory layout.

☐ a. 'Text' segment contains machine-language instructions of the program
☐ b. 'Data' segment contains global and static variables
☐ c. 'Stack' segment dynamically grows and shrinks
☑ d. 'Heap' segment is used to allocate memory at compile time

II. Choose [T] for True or [F] for False [F] (Each 5 points)

1. When running in Kernel MODE, a CPU can access memory that is marked as userspace. [T] ✔ [F] ☐
2. A **process** is an instance of an executed program. [T] ✔ [F] ☐
3. Two **processes** typically use the same memory space? [T] ☐ [F] ✔
4. A function contains more than one stack frame? [T] ☐ [F] ✔
5. The advantage of separating the virtual address space from the physical address space is isolating processes from one another to prevent one process from accessing the memory of another process. [T] ✔ [F] ☐
6. ***void free(void *ptr)*** deallocates the block of memory pointed to by ***ptr*** and adds the block of memory to a list of free blocks for re-use. [T] ✔ [F] ☐
7. When ***malloc( )*** allocates the block, it allocates extra bytes to hold the size of the block [T] ✔ [F] ☐
8. The expected output when you run the following **Python** script is [(2,3), 'xy'].

```
>>> t=[3,4.1,(2,3),'xy']
>>> print(t[2:])
```

[T] ✔ [F] ☐

III. Fill in the blank(s) in each statement.

1. Each time a function calls another function, stack frame or activation record is pushed onto the stack. This entry contains (return address) to go back to its caller, and (function parameters) and (local variables in the function) – 10 points

2. On x86_64 the stack grows in a (downward) direction and the heap grows in a (upward) direction – 10 points