New York University

CSCI-UA.0202-003: Operating Systems (Undergrad): Spring 2025

Quiz 1

- Write your full name on both:
  - the bubble sheet in the “Name” field
  - the quiz booklet
- Write your NYU NetID on the quiz booklet and the bubble sheet in the “ID” field
- Use a #2 pencil to fill in your answers on the bubble sheet
- This quiz contains 6 questions only. Each question has choices from A to D
- Fill the bubbles completely by darkening the entire circle, as shown in the example
- Only mark answers for questions 1-6. Do not mark any bubbles beyond question 6
- Choose only one answer per question
- Submit your bubble sheet together with your exam booklet

Name:

NetId:

1. In the context of x86-64 architecture, which statement about registers is **FALSE**?
   (a) `%rax` is used for storing function return values
   (b) `%rip` points to the next instruction being executed
   (c) `%rsp` points to the base of the current stack frame
   (d) `%rbp` should always contain a value that is higher than `%rsp`

1. After a `fork` system call, which statement is **TRUE**?
   (a) The child process starts execution from the beginning of the program
   (b) Only the parent process continues execution
   (c) The parent process waits until the child process completes
   (d) Both parent and child processes continue execution from the same point

2. Which memory segment contains dynamically allocated memory?
   (a) Stack
   (b) Text
   (c) Heap
   (d) Data

3. What is the role of the `syscall` instruction?
   (a) To allocate memory for system calls
   (b) To switch between user and kernel mode
   (c) To create new processes
   (d) To handle hardware interrupts

4. In the shell command `prog1 | prog2`, what happens to the file descriptors when the shell creates the pipe?
   (a) prog1's stdout is connected to prog2's stdin
   (b) prog1 and prog2 share all file descriptors
   (c) prog1's stderr is connected to prog2's stdin
   (d) prog1's stdin is connected to prog2's stdout

5. Consider the following C code:

```
int* foo() {
    int x = 42;
    return &x;
}
```

What is the **primary reason** that makes the code problematic?
(a) The function doesn't allocate enough memory for the integer
(b) The pointer arithmetic is incorrect
(c) The function returns a pointer to a local variable
(d) The variable x should be declared as a pointer