Zagazig University
Faculty of Engineering
Computer and Systems Dept.

Spring 2020
Third Year
CSE325: Operating Systems

Discussion #3

Covers: Chapter 3

Questions:

1. Using the program shown below, explain what the output will be at LINE A.

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
int value = 5;
int main()
{
   pid t pid;
   pid = fork();
   if (pid == 0) { /* child process */
        value += 15;
        return 0;
   }
   else if (pid > 0) { /* parent process */
        wait(NULL);
        printf("PARENT: value = %d", value); /* LINE A */
        return 0;
   }
}
```

2. Including the initial parent process, how many processes are created by the program shown below

```
#include <stdio.h>
#include <unistd.h>
int main()
{
   /* fork a child process */
   fork();
   /* fork another child process */
   fork();
   /* and fork another */
   fork();
   return 0;
}
```

3. The Sun UltraSPARC processor has multiple register sets. Describe what happens when a context switch occurs if the new context is already loaded into one of the register sets. What happens if the new context is in memory rather than in a register set and all the register sets are in use?

4. When a process creates a new process using the fork() operation, which of the following states is shared between the parent process and the child process?
    a. Stack
    b. Heap
    c. Shared memory segments
5. Describe the differences among short-term, medium-term, and long-term scheduling.
6. Describe the actions taken by a kernel to context-switch between processes.
7. Including the initial parent process, how many processes are created by the program shown below

```c
#include <stdio.h>
#include <unistd.h>
int main()
{
   int i;
   for (i = 0; i < 4; i++)
        fork();
   return 0;
}
```

8. Explain the circumstances under which the line of code marked printf("LINE J") in Figure 3.33 will be reached.

```c
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
int main()
{
   pid t pid;
   /* fork a child process */
   pid = fork();
   if (pid < 0) { /* error occurred */
        fprintf(stderr, "Fork Failed");
        return 1;
   }
   else if (pid == 0) { /* child process */
        execlp("/bin/ls", "ls", NULL);
        printf("LINE J");
   }
   else { /* parent process */
           /* parent will wait for the child to complete */
        wait(NULL);
        printf("Child Complete");
   }
   return 0;
}
```

9. Using the program shown below, identify the values of pid at lines A, B, C, and D. (Assume that the actual pids of the parent and child are 2600   and 2603, respectively.)

```c
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
int main()
{
   pid t pid, pid1;
   /* fork a child process */
   pid = fork();
   if (pid < 0) { /* error occurred */
        fprintf(stderr, "Fork Failed");
        return 1;
   }
   else if (pid == 0) { /* child process */
        pid1 = getpid();
        printf("child: pid = %d", pid); /* A */
        printf("child: pid1 = %d", pid1); /* B */
   }
   else { /* parent process */
        pid1 = getpid();
        printf("parent: pid = %d", pid); /* C */
        printf("parent: pid1 = %d", pid1); /* D */
        wait(NULL);
   }
   return 0;
}
```

10. Using the program shown below, explain what the output will be at lines X and Y.

```c
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
#define SIZE 5
int nums[SIZE] = { 0,1,2,3,4 };
int main()
{
   int i;
   pid t pid;
   pid = fork();
   if (pid == 0) {
        for (i = 0; i < SIZE; i++) {
             nums[i] *= -i;
             printf("CHILD: %d ", nums[i]); /* LINE X */
        }
   }
   else if (pid > 0) {
        wait(NULL);
        for (i = 0; i < SIZE; i++)
             printf("PARENT: %d ", nums[i]); /* LINE Y */
   }
   return 0;
}
```