CSCI423/501 Writing Assignment 2

Problem 1: Using the following program, 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. getpid() function returns the pid of the current process.)

```
#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;
}
```

Problem 2: Using the program shown below, explain what the output will be at lines X and Y, and whether such outputs are because the parent and child processes share their data or each of them has their own copies of data.

```
#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;
}
```

Problem 3: Multicore systems present certain challenges for multithreaded programming. Briefly describe these challenges.

Problem 4: Using Amdahl’s Law, calculate the speedup gain of an application that has a 40 percent parallel component for two processing cores.