```c
/*******************************************************************************
**
 * Lab 7 - Resources - cs452 - GVSU
 *
 * A playground for system controls to analyze the resources of the current system.
 *
 * Displays:
 *      - Page size
 *      - Pages in system
 *      - Max number of processes (2 different ways)
 *      - Max file size
 *      - Max open files (hard / soft)
 *      - Clock resolution
 *
 * @author Ron Rounsifer
 *******************************************************************************
**/
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <sys/resource.h> // for rlimit
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/time.h>
#define SIZE 4096
//#define SIZE 18446744073692774399

int main()
{
        int shmID;
        long int *shmPtr;
        //struct shmid_ds data;
        //struct shm_info info;

        // create shared memory
        if ((shmID = shmget(IPC_PRIVATE, SIZE, IPC_CREAT | S_IRUSR | S_IWUSR)) < 0)
        {
                perror("main: semget");
                exit(1);
        }


        // attach to shared memory
        if ((shmPtr = shmat(shmID, 0, 0)) == (void *) -1)
        {
                perror("cannot attach to memory");
                exit(1);
        }

        // determine page size (bytes)
        long page_size = 0;
        page_size = sysconf(_SC_PAGESIZE);
        printf("Page size: %ld\n", page_size);

        // determine physical pages in system
        long pages_in_system = 0;
        pages_in_system = sysconf(_SC_PHYS_PAGES);
        printf("Pages in system: %ld\n", pages_in_system);

        // Max number of child processes per user
        // Note: two different ways to do this shown
        long max_num_processes = 0;
        max_num_processes = sysconf(_SC_CHILD_MAX);
        printf("Max num process: %ld\n", max_num_processes);

        struct rlimit r;
        if (getrlimit(RLIMIT_NPROC, &r) == 0)
                printf("Max num process: %ld\n", r.rlim_max);

        // max filesize (bytes)
        struct rlimit rlim;
        if (getrlimit(RLIMIT_FSIZE, &rlim) == 0)
```

```
                printf("Max file size: %ld\n", rlim.rlim_max);

        // max num of open files
        if (getrlimit(RLIMIT_NOFILE, &r) == 0)
                printf("Max open files (hard): %ld\n", r.rlim_max);
        if (getrlimit(RLIMIT_NOFILE, &r) == 0)
                printf("Max open files (soft): %ld\n", r.rlim_cur);

        // clock resolution
        long clk_resolution = 0;
        clk_resolution = sysconf(_SC_CLK_TCK);
        printf("Clock resolution: %ld\n", clk_resolution);


        // detach from shared memory
        if (shmdt(shmPtr) < 0)
        {
                perror("cannot detach from memory");
                exit(1);
        }

        // remove shared memory
        if (shmctl(shmID, IPC_RMID, 0) < 0)
        {
                perror("cannot deallocate memory");
                exit(1);
        }
        return 0;
}
```