```
                     CS460 LAB5 pre-work
                   DUE: Thursday 3-12-2020

1. Step1: REQUIREMENT: YOUR own sh script mksdc to create a sdimage

2. Step2: samples/LAB5pre/step2:
   The USER directory contains:
          us.s
          u1.c, which includes string.c, uio.c, ucode.c
          ---------------------------------------------
          mku: assemble, compile, link us.s,u1.c to BINARY executable u1
          -------------------------------------------------------------
   us.s   entry point of Umode program: call main in C
          syscall:    int syscall(int a,b,c,d) to KERNEL by swi #0
          getcpsr:    get CPSR to verify it's in Umode

   u1.c   getpid, getppid, ps, chname, switch are all syscalls VIA
                  ucode.c
          ugetc(), uputc() are syscalls to kgetc|kputc in KERNEL

REQUIREMENTs:

(1). Write your OWN u2.c which print in UPPERcase OR in a foreign language.
(2). Use mku to generate u1, u2 to /bin directory of sdimage in step1.
===========================================================================
3. Step3

   samples/LAB5pre/step3 contains all files for the kernel.
   Run  mk  to test the system first.

   The link step is
arm-none-eabi-ld -T t.ld ts.o t.o uPtable.obj load.obj svc.obj -Ttext=0x10000 -o t.elf
   It uses 3 .obj files: uPtable.obj, load.obj, svc.obj

uPtable.obj: It implements a function
                int uPtable(PROC *p)
   which builds a page table for process p:

(1). Each proc has a 16KB ptable at
          4MB + (pid)*16KB
---------- process ptable in Kmode: all the same ------------------------
(2). The first 128 entries of ptable ID map to 128MB PA; FLAG=0x412
(3). Entres 256-257 map to 2MB I/O area at 256-258 MB    FLAG=0x412
                                                         AP=01 Domanin=0
-------------------------------------------------------------------------
(4). Each proc has a 1MB Umode image at VA=2GB (0x80000000).
     The physical address is at         PA=8MB + (pid-1)*1MB
         i.e. P1 at 8MB, P2 at 9MB, etc.
Thus,
     ptable[2048] map to the 1MB Umode image area;       FLAG=0xC32
                                                        AP=11 Domain=1

REQUIREMENT: Write your own uPtable(PROC *p) function to replace the uPtable.obj
HELP: read mkPtable() in t.c


load.obj: it contains a function
             int load(char *filename, PROC *p)
   which loads /bin/filename to p's Umode image area.

REQUIREMENT: write your own load.c file to replace load.obj

HINT: consult YOUR booter of LAB#1  (there is no ES register in ARM)
      getblock(int blk, char *buf) of sdc.c file

svc.obj: read the svc.c file to write the needed kernel functions
==============================================================================

                     COMBINED TEST:

  With your load.c, uPtable.c, svc.c (include them in t.c)
  Modify mk by deleting the .obj files in ld step

                     mk and run the system
```