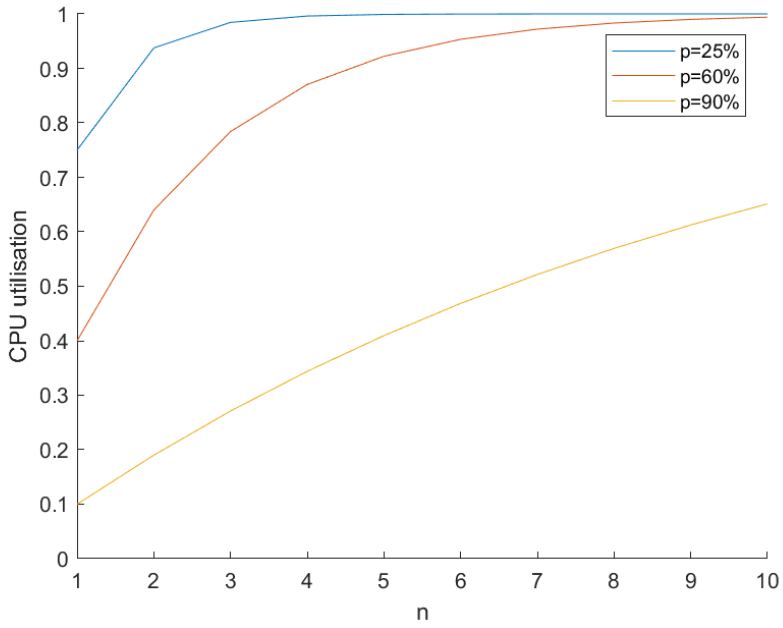
VE482 Homework 2

Liu Yihao 515370910207

Ex. 1 — Multiprogramming

1. The probability for n processes to be waiting at the same time is p^n.
   The CPU utilisation is $1 - p^n$.

2. 

3. a)
$$\lfloor (256 - 96) \div 48 \rfloor = 3$$
So three processes can be store simultaneously in memory.

   b)
$$1 - 0.9^3 = 27.1\%$$
So the CPU utilisation is 27.1%.

   c) When 256 MB is added, $\lfloor (512 - 96) \div 48 \rfloor = 8$ processes can be store simultaneously in memory, the CPU utilisation is $1 - 0.9^8 \approx 56.95\%$. It has a improvement of 29.85% per 256 MB.

When 512 MB is added, $\lfloor (768 - 96) \div 48 \rfloor = 14$ processes can be store simultaneously in memory, the CPU utilisation is $1 - 0.9^8 \approx 77.12\%$. It has a improvement of 10.09% per 256 MB.

When 1024 MB is added, $\lfloor (1280 - 96) \div 48 \rfloor = 24$ processes can be store simultaneously in memory, the CPU utilisation is $1 - 0.9^8 \approx 98.02\%$. It has a improvement of 5.23% per 256 MB.

In conclusion, we can find that adding the first 256 MB is the most beneficial and be worth the investment.

Ex. 2 — Keymap in Minix 3

There are three files to be modified.

The first file is `minix/servers/is/dmp.c`

```
struct hook_entry {
    int key;
    void (*function)(void);
    char *name;
} hooks[] = {
    { F1,   proctab_dmp, "Kernel process table" },
    { F3,   image_dmp, "System image" },
    { F4,   privileges_dmp, "Process privileges" },
    { F5,   monparams_dmp, "Boot monitor parameters" },
    { F6,   irqtab_dmp, "IRQ hooks and policies" },
    { F7,   kmessages_dmp, "Kernel messages" },
    { F8,   vm_dmp, "VM status and process maps" },
    { F10,  kenv_dmp, "Kernel parameters" },
    { SF1,  mproc_dmp, "Process manager process table" },
    { SF2,  sigaction_dmp, "Signals" },
    { SF3,  fproc_dmp, "Filesystem process table" },
    { SF4,  dtab_dmp, "Device/Driver mapping" },
    { SF5,  mapping_dmp, "Print key mappings" },
    { SF6,  rproc_dmp, "Reincarnation server process table" },
    { SF7,  proc_num_dmp, "Display the number of currently running processes" },
    { SF8,  data_store_dmp, "Data store contents" },
    { SF9,  procstack_dmp, "Processes with stack traces" },
};
```

SF7 is added in order to map Shift + F7.

The second file is `minix/servers/is/proto.h`

```
/* dmp_kernel.c */
void proc_num_dmp(void); // Added by myself
void proctab_dmp(void);
void procstack_dmp(void);
void privileges_dmp(void);
void image_dmp(void);
void irqtab_dmp(void);
```

```
void kmessages_dmp(void);
void monparams_dmp(void);
void kenv_dmp(void);
```

I added the definition of function `void proc_num_dmp(void)` here.

The third file is `minix/servers/is/dmp_kernel.c`

```
// Added by myself
/*===========================================================================*
 *                                proc_num_dmp                               *
 *===========================================================================*/
void proc_num_dmp(void)
{
    register struct proc *rp;
    int r;

    /* First obtain a fresh copy of the current process table. */
    if ((r = sys_getproctab(proc)) != OK) {
        printf("IS: warning: couldn't get copy of process table: %d\n", r);
        return;
    }

    int num = 0;
    for (rp = BEG_PROC_ADDR; rp < END_PROC_ADDR; rp++) {
        if (isemptyp(rp)) continue;
        num++;
    }
    printf("The number of currently running process is %d\n", num);
}
```

I added the implementation of function `void proc_num_dmp(void)` here.

Then I build the kernal and test it.

```
./releasetools/x86_hdiimage.sh
cd ../obj.i386/destdir.i386/boot/minix/.temp && qemu-system-i386 -serial stdio
↪  -kernel kernel -append "rootdevname=c0d0p1" -initrd
↪  "mod01_ds,mod02_rs,mod03_pm,mod04_sched,mod05_vfs,mod06_memory,mod07_tty,mod
↪  08_mfs,mod09_vm,mod10_pfs,mod11_init" -hda
↪  ~/minix/minix-3.3.0/minix-3.3.0/minix_x86.img --enable-kvm
```