Dept. Computer Science **Operating Systems**
Vrije Universiteit **26.03.2012**

1a Describe the two main functions of an operating system. *5pt*

1b Give two different ways an operating system can be structured. Describe pros and cons for each of these operating system layouts. *5pt*

2a The code sample below is an implementation of the reader-writers problem. Is this implementation safe? If not, explain in detail what can happen and how this implementation can be corrected. Also explain how the this implementation works, after eventually fixing it. Are there other problems beside safety? *10pt*

```
typedef int semaphore;

semaphore mutex   = 1;
semaphore db      = 0;

int reader_count = 0;
```

```
void reader(void)
{
 while (1) {
     down(&mutex);
     reader_count++;
     up(&mutex);
     if(reader_count==1) down(&db);
     read_database();
     down(&mutex);
     rc--;
     up(&mutex);
     if (rc == 0) up(&db);
     process_data();
 }
}
```

```
void writer(void)
{
 while (1) {
     generate_data();
     down(&db);
     write_database();
     up(&db);
 }
}
```

2b What is the difference between processes and threads? Further, threads can be implemented at kernel-level or at user-level: What are the pros and cons of both mechanisms (4 in total). *5pt*

2c Describe the steps that take place from turning on the computer until the Minix login screen appears. *10pt*

3a What are the devices `/dev/null`, `/dev/zero`, `/dev/ram`, `/dev/boot` `/dev/mem` and `/dev/kmem` used for? Describe similarities and differences in Minix' implementation of those devices. *10pt*

3b What is a deadlock? Name the four necessary conditions for a deadlock to occur and describe how to attack each of these four conditions. *5pt*

4a Describe how you would calculate the physical size of a single level page table? *5pt*

4c What is an inverted page table? When is it used? How does the page lookup work using an inverted page table and how can you accelerate it?. *5pt*

4d What is the purpose of a translation look-aside-buffer (TLB)? Describe two different ways the TLB can be managed? *5pt*

5a What is an i-node? Name five types of information that can be found in an i-node of the Minix file system? *5pt*

5b Consider a file system mounted to the root file system at the location /usr/ast/. Describe how the file look up for the text file /usr/ast/mbox would take place. (Both are Minix file systems). *5pt*

5c What is the difference between a write-back and a write-through buffer cache. What are their pros and cons? *5pt*

6a The following code sample shows the `check_sig()` function from the MINIX process_manager. Explain in detail what determines whether a signal is delivered and how many processes are signaled. Also interpret the possible return values of this function. What might the signal number 0 be used for? *10pt*

```
/*===========================================================================*
 *                              check_sig                                    *
 *===========================================================================*/
PUBLIC int check_sig(proc_id, signo)
pid_t proc_id;                  /* pid of proc to sig, or 0 or -1, or -pgrp */
int signo;                      /* signal to send to process (0 to _NSIG) */
{
  register struct mproc *rmp;
  int count;
  int error_code;

  if (signo < 0 || signo > _NSIG) return(EINVAL);

  if (proc_id == INIT_PID && signo == SIGKILL) return(EINVAL);

  count = 0;
  error_code = ESRCH;
  for (rmp = &mproc[0]; rmp < &mproc[NR_PROCS]; rmp++) {
        if (!(rmp->mp_flags & IN_USE)) continue;
        if ((rmp->mp_flags & ZOMBIE) && signo != 0) continue;

        if (proc_id > 0 && proc_id != rmp->mp_pid) continue;
        if (proc_id == 0 && mp->mp_procgrp != rmp->mp_procgrp) continue;
        if (proc_id == -1 && rmp->mp_pid <= INIT_PID) continue;
        if (proc_id < -1 && rmp->mp_procgrp != -proc_id) continue;

        if (mp->mp_effuid != SUPER_USER
            && mp->mp_realuid != rmp->mp_realuid
            && mp->mp_effuid != rmp->mp_realuid
            && mp->mp_realuid != rmp->mp_effuid
            && mp->mp_effuid != rmp->mp_effuid) {
                error_code = EPERM;
                continue;
        }

        count++;
        if (signo == 0) continue;

        sig_proc(rmp, signo);

        if (proc_id > 0) break;
  }

  if ((mp->mp_flags & (IN_USE | ZOMBIE)) != IN_USE) return(SUSPEND);
  return(count > 0 ? OK :  error_code);
}
```

Grading: *The final grade is calculated by adding the scores per question (maximum: 90 points), and adding 10 bonus points. The maximum total is therefore 100 points. The curve will be adjusted if the exam is too difficult.*