PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

SISTEMAS OPERATIVOS

1ra práctica (tipo a)
(Primer semestre de 2018)

Horario 0781: prof. V. Khlebnikov

Duración: 1 h. 50 min.
Nota: No se puede usar ningún material de consulta.
La presentación, la ortografía y la gramática influirán en la calificación.
Puntaje total: 20 puntos

Pregunta 1 (2 puntos – 10 min.)

(a) (1 punto – 5 min.) ¿Cuál es la memoria más rápida en una computadora?

(b) (1 punto – 5 min.) ¿Cuál es la opción alternativa al método llamado *busy waiting*?

Pregunta 2 (2 puntos – 10 min.) En en el enfoque microkernel:

(a) (0,5 puntos – 2,5 min.) ¿por qué se dice que el sistema operativo es fácil de extender?

(b) (0,5 puntos – 2,5 min.) ¿por qué se dice que es fácil de portar de un hardware, a otro diferente?

(c) (0,5 puntos – 2,5 min.) ¿por qué se dice que es más seguro y fiable?

(d) (0,5 puntos – 2,5 min.) ¿Cómo los programas de usuario y los servicios del sistema interactúan?

Pregunta 3 (8 puntos – 40 min.) Describa el árbol y la "vida" de cada proceso durante la ejecución del siguiente programa. Considere que el PID del proceso, creado por el shell durante la ejecución de este programa, es 24.

```
$ cat -n 2018-1_pr1.c
     1  #include <sys/types.h>
     2  #include <sys/wait.h>
     3  #include <stdlib.h>
     4  #include <unistd.h>
     5
     6  void die(char *s);
     7
     8  int
     9  main(void)
    10  {
    11    int pfd[2], n=2, i=0;
    12    pid_t pid, ppid;
    13
    14    ppid=getppid();
    15    if (pipe(pfd) == -1) die("pipe() error\n");
    16    if ((pid=fork()) == -1) die("fork() error\n");
    17    if (!pid) {
    18      if (dup2(pfd[1],1) == -1) die("dup2(pfd[1],1) error\n");
    19      (void) close(pfd[0]);
    20      sleep(1);
    21      execl("/bin/ps", "ps", "-l", NULL);
    22      die("execl(/bin/ps) error\n");
    23    }
    24    while(i<n && (fork() || !fork())) i++;
    25    sleep(3);
    26    while(waitpid(-1,NULL,0) != -1);
    27    if (ppid != getppid()) exit(0);
    28    if (dup2(pfd[0],0) == -1) die("dup2(pfd[0],0) error\n");
    29    (void) close(pfd[1]);
    30    execl("/usr/bin/less","less",NULL);  /* less is a program similar to more (1), more is a filter */
    31    die("execl(/usr/bin/less) error\n"); /* for paging through text one screenful at a time.       */
    32  }
    33
    34  void die(char *s)
    35  {
    36    if (s != (char *)NULL) {
    37      while (*s) (void) write(2, s++, 1);
    38    }
    39    exit((s == (char *) NULL) ? 0 : 1);
    40  }
```

<u>Pregunta 4</u> (8 puntos – 40 min.)

a) (3 puntos – 15 min.) Explique qué sucede cuando se ejecuta el siguiente programa:

```
1 #include <sys/types.h>
2 #include <sys/stat.h>
3 #include <stdio.h>
4 #include <unistd.h>
5 #include <fcntl.h>
6
7 int main(void) {
8       int fd;
9
10      if ((fd = open("salida",O_CREAT | O_RDWR, 0770))<0) perror("open failed");
11      if (dup2(fd,1)<0) perror("dup2 failed");
12      printf("Hello world\n");
13      close(fd);
14      if(execlp("sleep","sleep","0",(char *)NULL)<0) perror("exec failed");
15      exit(EXIT_SUCCESS);
16 }
```

b) (5 puntos – 25 min.) Analice el siguiente código y responda a las preguntas que están a continuación.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <unistd.h>
4  #include <sys/wait.h>
5  #include <errno.h>
6  #include <string.h>
7
8  #define READ 0
9  #define WRITE 1
10
11 void error(char *cad)
12 {
13   fprintf(stderr,"%s\n",cad);
14   exit(1);
15 }
16
17 void filter (int *fd)
18 {
19      int tmp, num, pid,x;
20      int fd2[2];
21
22      if(read(fd[READ],&tmp, sizeof(int))<0) error("It can not read number");
23
24      /* A -1 indicates shutdown */
25      if (tmp> -1) {
26         fprintf(stderr,"%d\n",tmp);
27
28         /* Release descriptors */
29            for(x=3;x<fd[0];x++) close(x);
30
31         /* Now we allocate stuff for the next process */
32         if(pipe(fd2)<0) error("filter: I can not make pipe");
33
34         if((pid = fork ())< 0) error("filter: I can not spawn process");
35
36         /* Now process the integer stream */
37         if (pid > 0)  /* PARENT */
38            do {
39                if(read(fd[READ], &num, sizeof(int))<0)
40                            error("filter: I can not read numbers from pipe");
41                /* get the integer from the pipe */
42                if (num <0 || (num % tmp) != 0)
43                      if(write(fd2[WRITE], &num, sizeof(int))<0)
44                                  error("filter: I can not write numbers in pipe");
45            } while (num != -1);
46         else
47             filter (fd2);
48       }
49 }
50
51 int main(int narg, char *argv[])
52 {
53      int i, numbers, pid, status;
54      int first[2];
55
```

```
56      /* Get info from the user */
57        if(narg !=2) {
58              fprintf(stderr,"Usage: %s <number>\n",argv[0]);
59              exit(1);
60      }
61
62      numbers = atoi(argv[1]);
63
64      /* Allocate the first filter process */
65      if(pipe(first)<0) error("main: I can not make pipe");
66      if((pid = fork ())< 0) error("main: I can not spawn process");
67      if (pid == 0)  /*CHILD*/
68              filter(first);
69      else {
70              /* Generate the integer strearn and send it out! */
71              printf("... in the range 1 to %d:\n", numbers);
72              fflush(stdout);
73              for (i=2; i<=numbers; i++)
74                      if(write(first[WRITE], &i, sizeof(int))<0)
75                           error("main: I can not write numbers in pipe");
76
77              /* Done! Send a shutdown rnessage! */
78              i= -1;
79              if(write(first[WRITE], &i, sizeof(int))<0)
80                            error("main: I can not write -1 in pipe");
81
82              /* Only stop when the last process has stopped! */
83              wait(&status);
84      }
85 }
```

- (2 puntos – 10 min.) ¿Cuál es la salida del programa cuando se ejecuta con 20, como argumento en la línea de comando? Dibuje el árbol de procesos (considere el número 16730 como *pid* del primer proceso).

- **(1 punto – 5 min.)** La salida del programa se mezcla con el *prompt* del *shell* ¿a qué se debe esto y cómo se puede solucionar?

- **(2 puntos – 10 min.)** Si se desea que el mismo programa imprima el árbol de procesos, dónde y qué líneas se deben agregar.

La práctica ha sido preparada por AB (2,4) y VK (1,3)
con LibreOffice Writer en Linux Mint 18.3 Sylvia.

Profesores del curso: (0781) V. Khlebnikov
(0782) A. Bello R.

Pando, 6 de abril de 2018