ISEP – LEEC - Sistemas Computacionais 2017/2018

Ficha 6 – *Semáforos POSIX*

As respostas às questões 2.2 e 2.3 deverão ser entregues em manuscrito e individualmente no início da próxima aula PL.

Extraia o conteúdo do ficheiro `ficha-sinc-ficheiros.zip` para o seu diretório de trabalho. Execute o comando `make` para criar os ficheiros executáveis.

1 - Analise o seguinte extrato de um programa:

```
int main() {
  int *v = mmap(NULL, sizeof(int), PROT_READ | PROT_WRITE,
   MAP_SHARED | MAP_ANONYMOUS, -1, 0);

  sem_t *psem = sem_open("/sem1", O_CREAT | O_RDWR, 0600, 1);

  v[0] = 0;
  int r, n = 0;
  for(int i=0;i<2;++i) {
    sleep(1);
    r = fork();
    ++n;
    if(r == 0) {
      r = fork();
      ++v[0];
      if(r == 0) {
        sem_wait(psem);
        sleep(3);
        sem_post(psem);
        printf("n = %d, v = %d, pid = %d, ppid = %d.\n", n, v[0], getpid(), getppid());
        return(0);
      }
      sleep(1);
      printf("%d a terminar; *v = %d.\n", getpid(), v[0]);
      exit(0);
    }
    waitpid(r, NULL, 0);
    printf("%d terminado; *v = %d, n = %d.\n", r, v[0], n);
  }
}
```

Apresente a sequência de impressões produzidas por este programa. Assuma que não existem interferências de outros processos no sistema, que o identificador do processo inicial é 2000 e que o(s) novo(s) processo(s) toma(m) o(s) valor(es) seguinte(s). Apresente um diagrama temporal representativo da execução do programa e justifique sucintamente. A sequência de impressões deve ser apresentada de forma destacada.

Note que recorrendo à *flag* MAP_ANONYMOUS, a função mmap permite criar um bloco de memória partilhada sem ser necessário recorrer à função `shm_open`.

2 - Analise o programa contido em `ex2.c`. A função `myprint` é a mesma função descrita no exercício 1 da ficha anterior.

2.1 - Utilize o mecanismo de semáforos com nome para garantir que a impressão de cada processo é enviada para o ecrã sem ser interrompida pelas impressões do outro processo.

2.2 – Apresente um diagrama temporal da execução deste programa desde o seu arranque até ao momento em cada processo já executou pelo menos uma vez a função `myprint`.

2.3 – Ao contrário do que se observou no exercício 1 da ficha anterior, basta agora fazer uma chamada à função `malloc` (nomeadamente, `buf = malloc(256)`) para que cada "tarefa" armazene e imprima a sua própria *string*. Porquê?

2.4 (Memória partilhada sem nome) - Altere a linha

```
char *buf = malloc(256);
```

para

```
char *buf = (char *) mmap(NULL, 256, PROT_READ | PROT_WRITE, MAP_SHARED | MAP_ANONYMOUS, -1, 0);
```

e verifique que ambos os processos passam a imprimir a mesma mensagem.

2.5 – Repita a alínea 2.1 usando um semáforo sem nome.