System and device programming

Iniziato	mercoledì, 1 giugno 2022, 00:18
Stato	Completato
Terminato	mercoledì, 1 giugno 2022, 00:18
Tempo impiegato	15 secondi
Valutazione	**0,00** su un massimo di 14,50 (**0**%)

Domanda 1

Risposta non data

Punteggio max.: 1,00

Indicate the output that the following program can generate when it is run on a UNIX-like system. Note that more than one response can indeed be correct and that incorrect answers may imply a penalty on the final score.

```
sem_t sa, sbc, sone, sd, me;
int na, nbc, nd;
na = nbc = nd = 0;
static void *TA ();
static void *TB ();
static void *TC ();
static void *TD ();

int main (int argc, char **argv) {
    pthread_t th;
    sem_init (&sa, 0, 0);
    sem_init (&sbc, 0, 0);
    sem_init (&sone, 0, 0);
    sem_init (&sd, 0, 1);
    sem_init (&me, 0, 1);
    setbuf(stdout, 0);
    pthread_create (&th, NULL, TA, NULL);
    pthread_create (&th, NULL, TB, NULL);
    pthread_create (&th, NULL, TC, NULL);
    pthread_create (&th, NULL, TD, NULL);
    pthread_exit(0);
}
static void *TA () {
    pthread_detach (pthread_self ());
    while (1) {
        sem_wait (&sa);
        printf ("A");
        na++;
        if (na<2) {
          sem_post (&sa);
        } else {
          printf ("\n");
          na = 0;
          sem_post (&sd);
        }
    }
    return 0;
}
static void *TB () {
    pthread_detach (pthread_self ());
    while (1) {
        sem_wait (&sbc);
        printf ("B");
        nbc++;
        if (nbc<2) {
            sem_post (&sbc);
            sem_wait (&sone);
        } else {
            nbc = 0;
            sem_post (&sone);
            sem_post (&sa);
        }
    }
    return 0;
}
static void *TC () {
    pthread_detach (pthread_self ());
    while (1) {
        sem_wait (&sbc);
        printf ("C");
        nbc++;
        if (nbc<2) {
            sem_post (&sbc);
```

```
            sem_wait (&sone);
        } else {
            nbc = 0;
            sem_post (&sone);
            sem_post (&sa);
        }
    }
    return 0;
}
static void *TD () {
    pthread_detach (pthread_self ());
    while (1) {
        sem_wait (&sd);
        printf ("D");
        nd++;
        if (nd<2) {
            sem_post (&sd);
        } else {
            nd = 0;
            sem_post (&sbc);
        }
    }
    return 0;
}
```

Scegli una o più alternative:

- [ ] (a) It can print lines with string DDCCBBAA.
- [ ] (b) It can print lines with string DACBDA.
- [ ] (c) It can print lines with string DCBA.
- [ ] (d) It can print lines with string DACBAD.
- [ ] (e) It can print lines with string DDBCAA.
- [ ] (f) It can print lines with string DDCBAA.

Risposta errata.

La risposta corretta è: It can print lines with string DDBCAA.
, It can print lines with string DDCBAA.

Domanda 2

Risposta non data

Punteggio max.: 1,00

The following program is run on a UNIX-like system with a single parameter on the command line equal to 2. Indicate the output the program generates. Note that more than one response can indeed be correct and that incorrect answers may imply a penalty on the final score.

```
int main (int argc, char *argv[]) {
   char str [50];
   int i;
   setbuf (stdout, 0);
   i = atoi (argv[1]);
   if (i<0)
      exit (0);
   if (fork () > 0) {
      if (fork () > 0) {
         sprintf (str, "echo -n S");
         system (str);
      }
      sprintf (str, "%d", i-2);
      printf ("E");
      execlp (argv[0], argv[0], str, NULL);
   }
   exit (0);
}
```

Scegli una o più alternative:

- [ ] (a) SEESEE
- [ ] (b) ESEESEESE
- [ ] (c) ESEESE
- [ ] (d) EESEESEES
- [ ] (e) SEESEESEE
- [ ] (f) EEEEEESSS

Risposta errata.

La risposta corretta è: ESEESEESE, SEESEESEE

Domanda 3

Risposta non data

Punteggio max.: 2,50

Consider inter-process communication in the Unix environment. Compare FIFOs and message queues. Suppose you must transfer between a process PW (the writer) and a process PR (the reader) a sequence of records of type

```
struct record_s {
    int id;
    char s1[N], s2[N];
    float f;
} record_t;
```

where N is a constant value. Report two code snippets, the first one illustrating the strategy with FIFOs and the second one with message queues. Describe the advantages and disadvantages of both strategies.

Please, remind the following system calls:

```
int mkfifo (const char *path, mode_t mode);
int mkfifoat (int fd, const char *path, mode_t mode);
key_t ftok (const char *path, int id);
int msgget (key_t key, int flag);
int msgctl (int msqid, int cmd, struct msqid_ds *buf);
int msgsnd (int msqid, const void *ptr, size_t nbytes, int flag);
ssize_t msgrcv (int msqid, void *ptr, size_t nbytes, long type, int flag);
```

If you do not remember the exact syntax of a specific system call, please write down a version that likely resembles what you remember together with some comments, briefly summarizing what you were willing to use.

Domanda 4

Risposta non data

Non valutata

If you want to withdraw from this part (Quer/Vetrò) of the exam, please select true/vero/yes. Otherwise, i.e., you want to take the exam, select false/falso/no.

Notice: It is also possible to withdraw once the exam has been completed, sending an e-mail to the instructors.

Scegli una risposta:

- Vero
- Falso

La risposta corretta è 'Falso'.

Domanda 5

Risposta non data

Punteggio max.: 1,00

Consider the syntax of a lambda function:

```
[ A ] ( B ) -> C { D }
```

Where do you specify the function parameters?

Note that incorrect answers may imply a penalty on the final score.

- A
- B
- C
- D

Risposta errata.

La risposta corretta è: B

Domanda 6

Risposta non data

Punteggio max.: 1,50

Modify the body of function ***pay()*** and of function ***top_up_wallet()*** to make sure that thread ***a*** pays only after thread ***b*** has recharged the wallet. You MUST USE a condition variable.

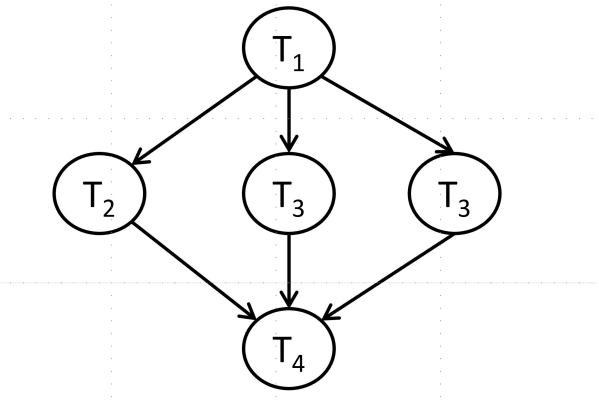
```
#include <thread>
int wallet = 3;

void pay(int pay_amount) {
    wallet-=pay_amount;
 }
void top_up_wallet() {
    wallet+=10;
}
int main() {
    std::thread a(pay,5);
    std::thread b(top_up_wallet);
    a.join();
    b.join();
    return 0;
}
```

Domanda 7

Risposta non data

Punteggio max.: 2,50

Implement in C code, under the Windows operating system, and with the minimum number of semaphores, the precedence graph which is represented in the following picture. The graph includes 4 threads; all threads are **cyclic** and there are two instances of thread number 3; both these instances have to run **once** for each main cycle.



The main program does **not** have to be implemented.
If you do not remember the exact syntax of a specific system call, please write down a version that likely resembles what you remember together with some comments, briefly summarizing what you were willing to use.

Domanda 8

Risposta non data

Punteggio max.: 2,50

Write a small C++ program to simulate the control of a bar to let cars enter and exit a parking lot.

For the sake of simulation, consider that a car arrives every t1 random seconds (1<= t1 <= 3), while a car leaves the parking lot every t2 random seconds (4<= t2 <=7). If a car cannot enter because the parking lot is full, it will just search for parking somewhere else (hence, no queues will form in front of the bar at the entrance).

The following parameters can be initialized within the program:

- Number of places in the parking lot.
- Duration of the simulation (in seconds), computed starting from the first car that enters.

The function to put a thread in the sleep status (e.g., for 1 second) is the following one:

```
std::this_thread::sleep_for (std::chrono::seconds(1))
```

Write the code of the program and manage threads synchronization.

Make sure all threads finish running before the main program terminates.

If you do not remember the exact syntax of C++ synchronization primitives, you can write down a mock version (with same sense...). Correctness is strictly required in the template syntax which is required to be right, as well as in any basic C++ syntax.

Domanda 9

Risposta non data

Punteggio max.: 2,50

A binary file, stored in the Windows system, includes an undefined number of records. Each record includes the following fields:

```
struct record_s {
   DWORD id;
   TCHAR s1[N], s2[N];
   FLOAT f;
} record_t;
```

Show how it is possible to use Windows system calls to:

- Read an integer value N from the standard input.
- Read the record in position N within the file and swap the strings s1 and s2 (s1 becomes s2 and vice-versa) locking the record before manipulating it.

Please, remind the following system calls:

```
HANDLE CreateFile (LPCTSTR lpName, DWORD dwAccess, DWORD dwShareMode, LPSECURITY_ATTRIBUTES lpsa,  DWORD dwCreate, DWORD dwAttrsAndFlags, HANDLE hTemplateFile);
BOOL ReadFile (HANDLE hFile, LPVOID lpBuffer, DWORD nNumberOfBytesToRead, LPDWORD lpNumberOfBytesRead,  LPOVERLAPPED lpOverlapped);
BOOL WriteFile (HANDLE hFile, LPCVOID *lpBuffer, DWORD nNumberOfBytesToWrite, LPDWORD lpNumberOfBytesWritten,  LPOVERLAPPED lpOverlapped);
BOOL SetFilePointerEx (HANDLE hFile, LARGE_INTEGER liDistanceToMove, PLARGE_INTEGER lpNewFilePointer,  DWORD dwMoveMethod);
BOOL LockFileEx (HANDLE hFile, DWORD dwFlags, DWORD dwReserved, DWORD nNumberOfBytesToLockLow, DWORD nNumberOfBytesToLockHigh, LPOVERLAPPED lpOverlapped);
```

If you do not remember the exact syntax of a specific system call, please write down a version that likely resembles what you remember together with some comments, briefly summarizing what you were willing to use.