System and device programming

Iniziato	giovedì, 8 settembre 2022, 13:40
Stato	Completato
Terminato	giovedì, 8 settembre 2022, 13:41
Tempo impiegato	13 secondi
Valutazione	**0,00** su un massimo di 15,00 (**0**%)

Domanda 1

Risposta non data

Punteggio max.: 2,00

Cheat Sheet UNIX.

Explain the main differences among blocking, non-blocking, multiplexed, and asynchronous I/O. Report a typical example of use for each of them. Motivate your answer.

For the students who followed the course before the academic year 2021-2022:

Explain the main differences among blocking, non-blocking, and asynchronous I/O. Report a typical example of use for each of them. Motivate your answer.

Domanda 2

Risposta non data

Punteggio max.: 2,50

An unknown number of threads can invoke one of the following two functions:

- **void wait_ch (int x)** that blocks the calling thread until the function **signal_ch (int y)** is called with **y==x.**
- **void signal_ch (int y)** that wakes up all threads that previously called the function **wait_ch (int x)** with **x==y** and are in the waiting state. It has no effect if there are no threads that are waiting and called **wait_ch (int x)** with **x==y** before.

Example of operations:

- Thread A calls **wait_ch (4)** and it is blocked.
- Thread B calls **wait_ch (2)** and it is blocked.
- Thread C calls **signal_ch (8)** without any effect.
- Thread D calls **wait_ch (4)** and it is blocked.
- Thread E calls **signal_ch (4)** and it wakes up threads A and D.
- Thread D calls **signal_ch (2)** and it wakes up thread B.

Implement the functions **wait_ch ()** and **signal_ch ()** using POSIX semaphores, defining the related shared variables. Suppose the values of **x** and **y** are limited to a pre-defined constant value **N**.

The students who followed the course before the academic year 2021-2022 can write the C code adopting the UNIX or the Windows API notation.

Domanda 3

Risposta non data

Punteggio max.: 1,00

Suppose that the following program is executed passing the value 3 on the command line.
Indicate the number of characters 'X' printed by the program on standard output.

```
int main (int argc, char *argv[]) {
  char str[100];
  int n = atoi(argv[1]);

  setbuf(stdout, 0);
  if (n>0) {
     fork();
     if (fork()) {
        printf("X");
        sprintf(str, "%s %d", argv[0], n-1);
        system(str);
     }
  }

  return (0);
}
```

- (a) 12
- (b) 8
- (c) 14
- (d) 10
- (e) 16
- (f) 15
- (g) 9
- (h) 13

Risposta errata.

La risposta corretta è: 14

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.: 2,50

Write a C++ program for simulating the control of an elevator in a building of two floors, according to the following main requirements:

- the elevator can be called by a person from ground floor (= floor 0), from floor 1 or from floor 2: when it is called from a floor different from the one the elevator is currently waiting, it should firstly move to that floor to pick the person up;
- once arrived at the calling floor, the elevator will receive an order to go to a different floor (make it random but consistent with its current status, i.e. if elevator is at floor 2, request should be for 0 or 1).

The following simplifications are allowed:

- elevator's trip time is assumed to be constant (set it to 2 seconds);
- elevator has no memory: it will not queue requests that it cannot serve, which are discarded (it is equivalent to a person taking the stairs if elevator is busy);
- new requests to the elevator arrive between 1 and 3 seconds;
- simulation runs for 30 seconds;

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 6

Risposta non data

Punteggio max.: 3,50

A program runs 4 threads, namely TA, TB, TC, and TD. These threads run forever and cooperate to generate sets of symbols on subsequent lines of the standard output, where each line has the following format:

$A^*\{B|C\}D^3$

This means that for each sequence there are:

- A*: Zero or more A, i.e., there is no limit to the numbers of A printed but no A can appear as well.
- {A|B}: One single B **or** one single C, not both.
- D^3: Exactly 3 symbols D.

Each sequence is terminated by a “new line” character.

The following is a correct example of the execution of such a program:

```
ABDDD
CDDD
BDDD
AACDDD
AAAAAABDDD
AAAACDDD
...
```

The students who followed the course before the academic year 2021-2022 can write the C code adopting the UNIX or the Windows API notation.

Domanda 7

Risposta non data

Punteggio max.: 1,00

The following piece of code shows a possible implementation of the system call **alarm()** by means of other system calls. Suppose that the code must trigger the alarm after N seconds.

```
void myAlarm (int sig) {
  if (sig==SIGUSR1)
     printf (“Alarm on ...\n”);
  return;
}

...
pid_t pid;
signal (SIGUSR1, myAlarm);
...
if (fork()) {
   pause (N);
   kill (getppid(), SIGUSR1);
   exit (0);
}
...
```

Please, indicate which of the following statements are correct. Note that incorrect answers imply a penalty in the final score.

Scegli una o più alternative:

- [ ] (a)
  The process must use the system call **sleep()** and not the **pause()**.
- [ ] (b)
  The system call **fork()** can be replaced by the system call **exec()**.
- [ ] (c)
  The process will never wake up from the system call **pause()**.
- [ ] (d)
  The **kill()** system call is not used correctly because a signal must be sent to the child process, and not the parent process.
- [ ] (e)
  In the previous piece of code a signal is sent from the parent to the child.
- [ ] (f)
  The signal **SIGALRM** has to be used, and not **SIGUSR1**.
- [ ] (g)
  The system call **getppid()** must be substituted by the system call **getpid()**.

Risposta errata.

La risposta corretta è:
The **kill()** system call is not used correctly because a signal must be sent to the child process, and not the parent process.,
The process must use the system call **sleep()** and not the **pause()**.,
The process will never wake up from the system call **pause()**.

Domanda 8

Risposta non data

Punteggio max.: 1,50

Which one(s) – if any – of the five copy control members is(are) executed at line

auto y1= [&]() -> Y { return f1(y0);} ; ? Explain **WHY**.

```
using namespace std;

class Y {
 public: //the five copy control members
   Y()  { } //default constructor dc
   Y(const Y&) {  } //copy constructor cc
   Y(Y &&) noexcept { }; //move constructor mc
   Y &operator=(const Y &) { } //copy assignment ca
   Y&operator=(Y &&) { } //move assignment ma
   ~Y() { } //destructor d
};

Y f1(Y &y) {Y y_copy = y; return y_copy;}
```

```
int main() {
   Y y0;
   auto y1= [&]() -> Y { return f1(y0);} ;
   y1();
   return 0;
}
```

Domanda 9

Risposta non data

Punteggio max.: 1,00

Which value of a is printed ?

```
#include <iostream>
#include <future>
#include <mutex>
using namespace std;
std::mutex m;
void plusHalf(double &i) {
      std::unique_lock<std::mutex> lk(m);
      i+=i/2;
}
void minusDouble(double &i) {
      std::unique_lock<std::mutex> lk(m);
      i-=(2*i);
   }
int main()
{
   double a = 10;
   future<void> a1 = std::async(std::launch::deferred, plusHalf, ref(a));
   future<void> a2 = std::async(std::launch::deferred, minusDouble, ref(a));
   std::cout << a << endl;
}
```

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

- It depends on the execution order of a1 and a2
- 10
- 15
- -15
- -10

Risposta errata.

La risposta corretta è: 10