

# Prova scritta di Sistemi di Elaborazione Ingegneria delle Telecomunicazioni

Ing. G. Lettieri, Ing. A. Vecchio

14 gennaio 2010

1. Supponiamo di avere il seguente programma scritto in parte in Assembler e in parte in C++:

<pre>.text .global f2 f2:     pushl %ebp     movl %esp, %ebp     pushal      movl 8(%ebp), %ebx     movl 12(%ebp), %eax     movl \$1, %esi l1:     cmpl %esi, %eax     jle fine      movl (%ebx, %esi, 4), %edx</pre>	<pre>    cmpl (%ebx) , %edx     jge av      movl (%ebx), %ecx     movl %edx, (%ebx)     movl %ecx, (%ebx, %esi, 4)  av:     incl %esi     jmp l1 fine:      popal     leave     ret</pre>
<pre>#include &lt;stdio.h&gt;  const int MAXN = 100;  void f2(int v[], int n);  void f1(int v[], int n) {     int i;     for(i=0; i&lt;n; i++) {         f2(&amp;v[i], n-i);         printf("%d\n", v[i]);     } }</pre>	<pre>} }  int main(int argc, char* argv[]) {     int v[MAXN];     int i;     for(i=1; i&lt;argc; i++)         sscanf(argv[i], "%d", &amp;v[i-1]);     f1(v, argc-1);     return 0; }</pre>

- Dire cosa viene calcolato dal programma complessivo.
- Tradurre la funzione `f1` in Assembler.

2. Scrivere i seguenti programmi in C++, utilizzando le primitive di Unix e la libreria standard del C.
- (a) Un programma **proteggi** con argomenti da riga di comando *file* e *car*, dove *file* è un nome di file e *car* è un carattere diverso da “@”. Il programma deve leggere il contenuto del file *file* un carattere alla volta e inviarlo sull’uscita standard. Se il carattere letto è *car* oppure “@”, deve prima inviare il carattere “@” e poi il carattere letto (quindi se legge “@” deve inviare due caratteri “@”).
- Un programma **rimuovi** con un argomento *car* da riga di comando, dove *car* è un carattere diverso da “@”. Il programma deve leggere il suo ingresso standard e contare quante volte *car* appare preceduto da una sequenza di un numero pari (zero incluso) di caratteri “@”. Quando ha letto tutto il suo ingresso deve stampare il numero così ottenuto e terminare.
- (b) Un programma **controlla** con argomenti *modo*, *file* e *car*. In ogni caso il programma deve creare un processo che esegue “**rimuovi**” con argomento *car*. L’argomento *modo* deve valere “p” oppure “n”. Se vale “p” il programma deve creare un ulteriore processo che esegue “**proteggi**” con argomenti *file* e *car*, collettato tramite pipe con il processo che esegue “**rimuovi**”. Se *modo* vale “n” il programma deve fare in modo che il processo che esegue “**rimuovi**” legga il file *file*. Alla fine il programma deve aspettare che i processi creati terminino e quindi deve terminare esso stesso.