Università di Verona

Corso di Laurea in Bioinformatica
Laboratorio di Elementi di Architetture e Sistemi Operativi
24 Luglio 2015

Esercizio 1 (11 punti)
Scrivere una funzione **assembler LC3** che preso come input l'indirizzo di una stringa di dimensione <**10** caratteri nel registro **R1** e un carattere (codice ASCII) nel registro **R2**, riporta in **R3** il numero di occorrenze del carattere nella stringa.
Si scriva inoltre il **main**, che legga da tastiera la stringa (NON é necessario fare controlli sulla lunghezza, si assume che sia comunque e sempre al massimo lunga 9 caratteri) e successivamente legga un carattere, chiami la funzione da te realizzata, e al termine stampi a video il valore restituito dalla funzione.
I parametri devono essere caricati esclusivamente nei registri indicati e la funzione deve usare esclusivamente i registri indicati per passare i parametri.
É **indispensabile** commentare il codice indicando almeno nelle parti salienti cosa si intende fare. La mancanza di commenti puó comportare la nullitá dell'esame.
Si presti molta attenzione alle conversioni tra caratteri e numeri, lo zero in ASCII vale 48d.

Esercizio 2 (11 punti)
Scrivere uno script **bash** per concatenare un insieme di file aventi la stessa estensione e contenuti nella stessa directory.
Lo script deve ricevere 3 argomenti nella linea di comando: l'estensione, la directory che contiene i file e il nome del file di output in cui deve essere salvata la concatenazione finale.
I dati concatenati devono essere **ordinati alfabeticamente** prima di essere memorizzati nel file di output.
Lo script deve controllare che il numero di argomenti della linea di comando sia giusto ed in caso contrario deve visualizzare un messaggio indicante il corretto utilizzo dello script. Infine lo script deve visualizzare un messaggio di errore anche nel caso in cui la directory fornita come argomento alla linea di comando non esista.

Esercizio 3 (11 punti)
Scrivere un programma C in cui un processo P crea un figlio F. P deve eseguire il comando `ls -a` e restituire l'output del comando a F utilizzando una pipe. F deve visualizzare nello stdout i dati ricevuti da P.

Eventuali System Call che possono essere utili:

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>
#include <sys/types.h>
int pipe(int filedes[2]);
pid_t fork(void);
int execl(const char *path, const char *arg, ...);
int execlp(const char *file, const char *arg, ...);
int execle(const char *path, const char *arg , ..., char * const envp[]);
int execv(char *path, char *const argv[]);
int execvp(const char *file, char *const argv[]);
int dup(int oldfd);
int dup2(int oldfd, int newfd);
ssize_t read(int fd, void *buf, size_t count);
ssize_t write(int fd, const void *buf, size_t count);
int close(int fd);
int open(const char *pathname, int flags, mode_t mode);
int creat(const char *pathname, mode_t mode);
```

Soluzione 1

```
.ORIG X3000
        LEA     R0, STRINGA     ; PREDISPONGO LA DOMANDA
        PUTS

        LEA     R3, STRING
        LD      R2, LF   ; Load 10

AGAIN:

        GETC                    ; Request keyboard
        OUT                     ; Stampo il carattere a video
        ADD     R1,R2,R0        ; Test for terminating
        BRz     EXIT            ; character
        STR     R0, R3, #0
        ADD     R3, R3, #1

        BRnzp AGAIN     ; ... Continuo all'infinito


EXIT:   LD R0, ZERO
        STR R0, R3, #0          ; Termino la stringa letta

        ; Leggo il carattere
        LEA     R0, CARATTERE     ; PREDISPONGO LA DOMANDA
        PUTS
        GETC                    ; Request keyboard
        OUT                     ; Stampo il carattere a video

        ADD     R2,R0, #0       ; Metto il carattere in R2 come richiesto
        LEA     R1, STRING      ; Metto indirizzo in R1

        JSR     CONTA

        LD        R0, ASCII
        ADD       R0, R0, R3
        OUT

        HALT

CONTA:  ;R1 <- Indirizzo ;R2 <- Carattere
        NOT R2, R2
        ADD R2, R2, 1  ; Trovo l'opposto del carattere

        AND R3, R3, 0

inizioW:
        LDR R4, R1, #0    ; R4 <- STRING[i]
        BRnz fineW
        ADD R1, R1, #1   ; INCREMENTO IL PUNTATORE
        ADD R4, R4, R2
        BRnp inizioW     ; Se il carattere non e' quello avanzo

        ADD R3, R3, 1    ; Se e' quello incremento e poi avanzo
        BRnzp inizioW
fineW:  RET


; Definizione di variabili
```

```
STRINGA       .STRINGZ    "Digita una frase massimo 9 caratteri (Invio per terminare):
; Stringa gi  terminata con \0
STRING        .BLKW 20
CARATTERE     .STRINGZ    "Digita il caratere da cercare: "
LF            .FILL      xFFF6
ASCII         .FILL      48
.END
```