Esercitazione 0 - Lettura e Scrittura File in Java e C

Corso di Reti di Calcolatori T

Bernardini Claudio, Corsetti Luca, Giardina Gianluca, Straccali Leonardo

Anno accademico 2021/2022

Scopo dell'esercitazione

Lo scopo di questa esercitazione è sviluppare un'architettura che prevede la creazione di due entità, Produttore e Consumatore:

- Il Produttore è un'entità che produce un file di testo contenente caratteri
- Il Consumatore è un filtro che, preso in ingresso (con varie modalità) una sequenza di caratteri definita dall'utente, elimina questi caratteri da un file dato.

Il Produttore

E' un processo che richiede all'utente di inserire riga per riga stringhe che verranno salvate in un file. Il file in questione viene passato come parametro. La lettura dei parametri inseriti dall'utente continua fino a quando l'utente stesso non invia EOF.

Il Consumatore

E' un processo che funge da **filtro a carattere** il cui scopo è eliminare da un file dato i caratteri contenuti in una sequenza, anch'essa data. La sequenza può essere fornita tramite *passaggio per argomento* o *ottenuta dalla prima riga del file passato*. La procedura di eliminazione è eseguita in modo sequenziale dall'inizio del file fino alla fine con il riconoscimento del carattere EOF.

Codice Java

Produttore

```
public class Produttore {
    public static void main(String[] args) {
        BufferedReader in = null;
        int res = 0;

        if (args.length != 1) {
            System.out.println("Utilizzo: produttore <inputFilename>");
            System.exit(0);
        }

        in = new BufferedReader(new InputStreamReader(System.in));

        FileWriter fout = null;
        String inputl = null;
```

```
        try {
            fout = new FileWriter(args[0]);
            while ((inputl = in.readLine()) != null) {
                inputl += System.lineSeparator();
                fout.write(inputl, 0, inputl.length());
            }
            fout.close();
        } catch (NumberFormatException nfe) {
            nfe.printStackTrace();
            System.exit(1); // uscita con errore, intero positivo a livello di
sistema Unix
        } catch (IOException e) {
            e.printStackTrace();
            System.exit(2); // uscita con errore, intero positivo a livello di
sistema Unix
        }
    }
}
```

Consumatore

```
public class Consumatore {
    public static void main(String[] args) {
        String filter = null;
        FileReader r = null;
        int x;
        char ch;

        if (args.length > 2) {
            System.out.println("Utilizzo: consumatore <filterprefix>
<inputFilename> oppure");
            System.out.println("Utilizzo: consumatore <filterprefix> '<'
<inputFilename>");
            System.exit(0);
        }

        filter = args[0];

        try {
            // se args.length == 2 allora il filtro è passato come parametro
            // altrimenti è da leggere da un file
            if (args.length == 2) {
                r = new FileReader(args[1]);
            } else {
                r = new FileReader(FileDescriptor.in);
            }
        } catch (FileNotFoundException e) {
            System.out.println("File non trovato");
            System.exit(1);
        }
```

```
        try {
            while ((x = r.read()) >= 0) {
                ch = (char) x;
                // se il carattere non è tra quelli da filtrare, printa
                if (filter.indexOf(ch) == -1)
                    System.out.print(ch);
            }
            r.close();
        } catch (IOException ex) {
            System.out.println("Errore di input");
            System.exit(2);
        }
    }
}
```

Codice C

Produttore

```
#define MAX_STRING_LENGTH 256

// produttore.c NON e' un filtro
int main(int argc, char* argv[]){
    int fd, readValues, bytes_to_write, written, righe, i;
    char *file_out;
    char riga[MAX_STRING_LENGTH], buf[MAX_STRING_LENGTH];

    //controllo numero argomenti
    if (argc != 2){
        perror(" numero di argomenti sbagliato"); exit(1);
    }

    file_out = argv[1];

    fd = open(file_out, O_WRONLY|O_CREAT|O_TRUNC, 00640);
    if (fd < 0){
        perror("P0: Impossibile creare/aprire il file");
        exit(2);
    }

    printf("Inserisci le righe del file: [EOF per terminare l'inserimento] \n");
    while (gets(riga) != NULL) {
        /* la gets legge tutta la riga, separatori inclusi, e trasforma il fine
           linea in fine stringa */
        // aggiungo il fine linea
        riga[strlen(riga)+1]='\0';
        riga[strlen(riga)]='\n';
        written = write(fd, riga, strlen(riga)); // uso della primitiva
        if (written < 0){
            perror("P0: errore nella scrittura sul file");
            exit(3);
```

```
        }
    }
    close(fd);
}
```

Consumatore

```
#define MAX_STRING_LENGTH 256

// consumatore.c e' un filtro
int main(int argc, char *argv[])
{

    char *file_in, read_char, buf[MAX_STRING_LENGTH], *delete_chars;
    int nread, fd;

    //controllo numero argomenti
    if (argc < 2 || argc > 3)
    {
        perror(" numero di argomenti sbagliato");
        exit(1);
    }

    delete_chars = argv[1];

    if (argc == 3)
    {
        file_in = argv[2];
        fd = open(file_in, O_RDONLY);
        if (fd < 0)
        {
            perror("P0: Impossibile aprire il file.");
            exit(2);
        }
    }
    else
    {
        fd = 0; //File descriptor stdin
    }
    while ((nread = read(fd, &read_char, sizeof(char)))) /* Fino ad EOF*/
    {
        if (nread >= 0)
        {
            if ((strchr(delete_chars, read_char)) == NULL)
                putchar(read_char);
        }
        else
        {
            printf("(PID %d) impossibile leggere dal file %s", getpid(), file_in);
            perror("Errore!");
            close(fd);
```

```
                exit(3);
            }
        }
        close(fd);
    }
```