

ESERCITAZIONE 0

Lettura e scrittura File in C e Java

Codice C: Produttore

```
int main(int argc, char* argv[]) {
    int fd, written;
    char *file_out, riga[MAX_DIM];

    file_out = argv[1];

    fd = open(file_out, O_WRONLY|O_CREAT|O_TRUNC, 00640);

    printf("Inserisci il testo o premi EOF per terminare.\n");

    while (fgets(riga, MAX_DIM, stdin)) {
        written = write(fd, riga, strlen(riga)*sizeof(char));
    }

    close(fd);
    return EXIT_SUCCESS;
}
```

Si legge il testo in input usando la funzione fgets all'interno di un ciclo while. Con la funzione fgets si legge una linea da stdin di dimensione massima MAX_DIM (o fino a quando non si incontra \n o EOF) e la si memorizza nel buffer riga.

Codice C: Consumatore

```
int main(int argc, char* argv[]) {
    int nread, fd, trovato;
    char read_char, charToRemove[MAX_DIM], carSeq;

    strcpy(charToRemove, argv[1]);

    if (argc == 3) {
        fd = open(argv[2], O_RDONLY);
    } else if (argc == 2) {
        fd = STDIN_FILENO;
    }

    while (nread = read(fd, &read_char, sizeof(char)) {
        trovato = 0;
        for (int i = 0; !trovato && i < strlen(charToRemove); i++) {
            carSeq = charToRemove[i];
            if (read_char == carSeq) {
                trovato = 1;
            }
        }
        if (!trovato) {
            putchar(read_char);
        }
    }
    close(fd);
}
```

Per semplificare e abbreviare il codice si fa all'inizio una distinzione tra il caso in cui si passano entrambi i parametri come argomenti e il caso in cui si usa la ridirezione; a seconda del caso si mette nella variabile fd il file descriptor del file passato come argomento o dello standard input.

Avendo gestito l'input prima, ora si può scrivere un codice unico con la lettura da file usando la variabile fd. L'algoritmo per eliminare i caratteri prevede di leggere un carattere alla volta e confrontarlo con tutti quelli che sono stati passati come argomento. Se il carattere letto è tra quelli passati come argomento, si cambia il valore della variabile trovato da 0 a 1 (e si esce dal ciclo for). Se invece trovato vale 0 alla fine del ciclo for, si stampa il carattere letto.

Codice Java: Produttore

```
public class Produttore {  
    public static void main(String[] args) {  
        BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
        FileWriter fout;  
  
        try {  
            String line;  
            fout = new FileWriter(args[0]);  
  
            while ((line = in.readLine()) != null) {  
                fout.write(line, 0, line.length());  
                fout.write('\n');  
            }  
            fout.close();  
        }  
        catch (IOException e) {  
            e.printStackTrace();  
            System.exit(1);  
        }  
    }  
}
```

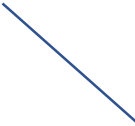
Codice Java: Consumatore (1)

```
public class Consumatore {  
    public static void main(String[] args) {  
        BufferedReader bufferedReader = null;  
        boolean charIsPresent = false;  
        String characters_to_remove;  
        int nread;  
        char read_char;  
  
        characters_to_remove = args[0];  
  
        if (args.length == 1) {  
            bufferedReader = new BufferedReader(new InputStreamReader(System.in));  
        }  
        else if (args.length == 2) {  
            try {  
                bufferedReader = new BufferedReader(new FileReader(args[1]));  
            }  
            catch (FileNotFoundException e) {  
                System.out.println("Il file non è stato trovato.");  
                System.exit(1);  
            }  
        }  
    }  
}
```

Anche in questo caso si fa inizialmente una distinzione tra i due casi di input (con e senza ridirezione) e si usa in entrambi i casi un `BufferedReader`, in modo tale da rendere la lettura da file/input unica.

Codice Java: Consumatore (2)

```
try {  
    while ((nread = bufferedReader.read()) >= 0) {  
        read_char = (char) nread;  
        for (int i = 0; i < characters_to_remove.length() && !charIsPresent; i++) {  
            if (read_char == characters_to_remove.charAt(i)) {  
                charIsPresent = true;  
            }  
        }  
        if (!charIsPresent) {  
            System.out.print(read_char);  
        }  
        charIsPresent = false;  
    }  
    System.out.print('\n');  
    bufferedReader.close();  
}  
catch (IOException e) {  
    System.out.println("Errore di input.");  
    System.exit(2);  
}  
}
```



L'algoritmo per leggere il testo e stampare i caratteri non presenti nella stringa di caratteri passata come argomento è identico a quello in C.