ESERCITAZIONE DI VENERDÌ 23/04/2021

NOTA BENE: per ognuno dei programmi C che devono essere scritti per questa esercitazione (e per le prossime) deve essere scritto anche il makefile che produce il file eseguibile controllando tutti i possibili WARNING di compilazione (opzione –Wall) che devono sempre essere eliminati, come chiaramente anche gli errori di compilazione e di linking!

1) Scrivere un programma in C **parametri1.c** che, dopo aver controllato di essere invocato con almeno 1 parametro, visualizzi su standard output il nome dell'eseguibile e il numero dei parametri e, quindi, il valore di tutti i parametri (cioè le corrispondenti stringhe) inserendo, contestualmente, il numero d'ordine dei singoli parametri. Si verifichi il funzionamento sia in assenza di parametri che con un numero variabile di parametri.
2) Scrivere un programma in C **parametri2.c** che deve essere invocato esattamente con 3 parametri: il primo deve essere considerato il nome di un file, il secondo un numero intero N strettamente positivo e il terzo deve essere considerato un singolo carattere C. Dopo aver fatto tutti i controlli necessari, si visualizzi su standard output il nome dell'eseguibile e il numero dei parametri (come nell'esercizio precedente) e, quindi, il valore dei tre parametri (secondo il loro significato) inserendo opportune frasi che facciano capire all'utente che cosa viene visualizzato. Si verifichi il funzionamento sia in assenza di parametri che con il numero giusto di parametri, ma di 'tipo' non corretto e infine il funzionamento corretto.
3) Scrivere un programma in C **contaOccorrenze.c** che deve essere invocato esattamente con 2 parametri: il primo deve essere considerato il nome di un file F, mentre il secondo deve essere considerato un singolo carattere Cx. Dopo aver fatto tutti i controlli necessari, si visualizzi su standard output il nome dell'eseguibile e i parametri e quindi si calcoli quante occorrenze (*in termini di long int*) del carattere Cx sono presenti nel file F e si visualizzi tale valore su standard output inserendo opportune frasi che facciano capire all'utente che cosa viene visualizzato. Si verifichi il funzionamento.
4) Scrivere un programma in C **mycat1.c** che, partendo dal programma mycat.c mostrato a lezione, consideri di poter essere invocato anche con un numero qualunque di nomi di file (anche nessuno!). Si verifichi il funzionamento.
5) Scrivere un programma in C **22sett99-1.c** che, partendo dal programma 22sett99.c mostrato a lezione, consideri un ulteriore parametro che deve essere ancora un singolo carattere (come il secondo parametro) che deve essere usato per sostituire il carattere cercato. Si verifichi il funzionamento, anche nel caso che il carattere passato come terzo parametro sia il carattere spazio/blank!
6) Scrivere un programma in C **append1.c** che, partendo dal programma append.c mostrato a lezione, ottenga lo stesso comportamento, ma usando la primitiva open nella sua forma avanzata per spostare contestualmente l'I/O pointer alla fine del file. Si verifichi il funzionamento.
7) Scrivere un programma in C **selezionaMultipli.c** che deve essere invocato esattamente con 2 parametri: il primo deve essere considerato il nome di un file F, mentre il secondo un numero intero n strettamente positivo. Dopo aver fatto tutti i controlli necessari, si visualizzi su standard output tutti i caratteri del file F che si trovano in posizione multipla di n insieme con l'indicazione del numero di tale multiplo (ad esempio scrivendo "Il carattere multiplo *x*-esimo all'interno del file e' *y*" dove *x* è il numero del multiplo e *y* è il carattere). Si verifichi il funzionamento.
8) Scrivere un programma in C **myhead1.c** che deve essere invocato con esattamente un parametro che deve essere considerata una opzione -*numero*. Dopo aver fatto tutti i controlli necessari, il programma si deve comportare come il filtro head e quindi deve filtrare in uscita le prime *numero* linee dello standard input. Si verifichi il funzionamento.
9) Scrivere un programma in C **myhead2.c** che, partendo dal programma **myhead1.c,** può essere invocato con una opzione -*numero* (come il precedente) o anche senza alcun parametro: in questo secondo caso, deve essere considerato di filtrare le prime 10 linee, esattamente come si comporta il filtro head. Si verifichi il funzionamento.
10) Scrivere un programma in C **myhead3.c** che, partendo dal programma **myhead2.c,** si comporti sia come filtro che come comando (filtrando le linee del file specificato invece che quelle dello standard input) e quindi potendo avere fino a due parametri (il primo deve essere considerato l'opzione –*numero* e il secondo il nome del file). Si verifichi il funzionamento.
11) Scrivere un programma in C **selezionaLinea.c** che deve essere invocato esattamente con 2 parametri: il primo deve essere considerato il nome di un file F, mentre il secondo un numero intero n strettamente maggiore di 0. Dopo aver fatto tutti i controlli necessari, si visualizzi su standard output la linea n-esima del file F oppure si riporti che tale linea non esiste. Si verifichi il funzionamento.
12) Scrivere un programma in C **selezionaLunghezzaLinea.c** che deve essere invocato con esattamente 2 parametri: il primo deve essere considerato il nome di un file F, mentre il secondo un numero intero n strettamente positivo. Dopo aver fatto tutti i controlli necessari, si visualizzi su standard output tutte le linee del file F la cui lunghezza, compreso il terminatore di linea, sia esattamente uguale a n oppure si riporti che non esiste alcuna linea con lunghezza uguale a n. Si verifichi il funzionamento.