Esercitazione di Venerdì 20 Aprile 2018

1) (parametri1.c) Scrivere un programma in C che, dopo aver controllato di essere invocato con almeno 1 parametro, visualizza su standard output il valore di tutti i parametri (cioè le corrispondenti stringhe) inserendo opportune frasi che facciano capire all'utente che cosa viene visualizzato.
2) (parametri2.c) Scrivere un programma in C che deve essere invocato con esattamente 3 parametri: il primo deve essere considerato il nome di file, il secondo un numero intero N strettamente maggiore di 0 e il terzo deve essere considerato un singolo carattere C. Dopo aver fatto tutti i controlli necessari, si visualizza su standard output il valore di tutti i parametri (secondo il loro significato) inserendo opportune frasi che facciano capire all'utente che cosa viene visualizzato.
3) (contaOccorrenze.c) Scrivere un programma in C che deve essere invocato con esattamente 2 parametri: il primo deve essere considerato il nome di file F, mentre il secondo deve essere considerato un singolo carattere Cx. Dopo aver fatto tutti i controlli necessari, si calcoli quante occorrenze del carattere Cx sono presenti nel file F e si visualizza tale valore su standard output inserendo opportune frasi che facciano capire all'utente che cosa viene visualizzato.
4) (mycat1.c) Scrivere un programma in C che, partendo dal programma mycat.c mostrato a lezione, consideri di poter essere invocato anche con un numero qualunque di nomi di file.
5) (22sett99-1.c) Scrivere un programma in C che, partendo dal programma 22sett99.c mostrato a lezione, consideri un ulteriore parametro carattere che deve essere usato per sostituire il carattere cercato.
6) (append1.c) Scrivere un programma in C che, partendo dal programma append.c mostrato a lezione, ottenga lo stesso comportamento, ma usando la open nella sua forma avanzata.
7) (selezionaMultipli.c e selezionaMultipli1.c) Scrivere un programma in C che deve essere invocato con esattamente 2 parametri: il primo deve essere considerato il nome di file F, mentre il secondo un numero intero n strettamente maggiore di 0. Dopo aver fatto tutti i controlli necessari, si visualizza su standard output tutti i caratteri del file F che si trovano in posizione multipla di n.
8) (myhead1.c, myhead2.c e myhead3.c) Scrivere un programma in 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 modifichi, quindi tale programma in modo che in assenza di parametri venga considerato di filtrare le prime 10 linee, esattamente come di comporta il filtro head. Da ultimo, si modifichi ulteriormente il programma in modo che 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).
9) (selezionaLinea.c) Scrivere un programma in C che deve essere invocato con esattamente 2 parametri: il primo deve essere considerato il nome di file F, mentre il secondo un numero intero n strettamente maggiore di 0. Dopo aver fatto tutti i controlli necessari, si visualizza su standard output la linea n-esima del file F.
10) (selezionaLunghezzaLinea.c) Scrivere un programma in C che deve essere invocato con esattamente 2 parametri: il primo deve essere considerato il nome di file F, mentre il secondo un numero intero n strettamente maggiore di 0. Dopo aver fatto tutti i controlli necessari, si visualizza su standard output tutte le linee del file F la cui lunghezza compreso il terminatore di linea sia esattamente uguale a n.