MEMO PER IL PROGRAMMATORE PER EVITARE GLI ERRORI PIU' COMUNI

1. Controllare sempre il return code di una funzione o system call
2. Liberare le aree di memoria allocate dinamicamente con *alloc() quando non si utilizzano più.
3. Non allocare memoria dinamica per aree di memoria o vettori di dimensione nota a priori.
4. Non ripetere interi blocchi di codice in modo identico, ma racchiuderli in una funzione.
5. Inserire commenti concisi, ma chiari, e premettere il codice di ogni funzione con un breve header di spiegazione.
6. Fare i test di funzionalità sia nelle condizioni corrette sia in quelle che devono generare errore.
7. Ricordarsi che in C esiste il terminatore di stringa '\0' e quindi tenerne conto nel dimensionamento dei buffer che devono contenere stringhe.
8. In caso di errore o addirittura terminazione anticipata di un programma, informare sempre l'utente con la stampa su stderr di un messaggio di errore comprensibile.
9. Eliminare prima della consegna le stampe utili solo per la fase di debug, lasciando solo quelle minimali per permettere all'utente di interagire correttamente col programma.
10. Eliminare prima della consegna tutti i file non necessari, quali copie di backup, file superflui e tutti ciò che viene generato dal makefile (.o, eseguibili, librerie).
11. Un processo padre deve recuperare tramite wait/waitpid il termination status del figlio per evitare che diventi un processo zombie.
12. Prevedere la possibilità che l'utente digiti semplicemente Enter (=linea vuota) al prompt di smmMon senza che il programma abortisca!
13. Inserire nel makefile la dipendenza dei moduli oggetto .o da eventuali include file .h, oltre che dai sorgenti .c.
14. Un makefile non deve eseguire nuovamente le azioni se ha già generato quanto previsto.
15. Verificare che quanto affermato nella relazione sia sempre allineato con quanto implementato.
16. Scrivere nella relazione tutto ciò che non è già definito dalle specifiche e che rende unica la propria implementazione (messaggi, strutture dati, codici di errore, meccanismi di sincronizzazione tra processi, algoritmo per il compattamento, metodo adottato per la protezione della memoria, ...).