Documentazione progetto di S.O. 2022/2023 – Phase 1

Per la fase 1 si è scelto, come suggerito dalle specifiche del progetto, di distinguere il codice in tre moduli differenti. In generali, funzioni ausiliarie e variabili globali sono state marcate come `HIDDEN` / `static` poiché non necessarie all'infuori del modulo di riferimento. Non sono state effettuate modifiche ai file forniti, eccetto l'inclusione delle librerie tramite simboli " " piuttosto che < >, in quanto tutto presente nella stessa directory. Di seguito le principali scelte implementative fatte nello specifico nei singoli moduli:

- `pcb.c / pcb.h`: lo sviluppo del codice per questo modulo è stato molto basilare, ci si è limitati a seguire le indicazioni fornite dal testo del progetto e a cercare di implementare le funzioni in modo che rispettassero le specifiche. La parte più interessante è sicuramente la gestione dell'albero dei processi. Scelta significativa da un punto di vista implementativo degna di nota è stata quella di mantenere l'ordine di inserimento dei figli nella funzione `insertChild().`

- `ash.c / ash.h`: poiché il numero di semafori è fissato a `MAXPROC = 20`, la tabella hash che implementa la *Active Semaphore Hash* viene allocata usando 5 bit, di fatto rendendola di dimensione pari a 32. Questo consente di avere un numero di *bucket* sufficiente per i semafori, e contemporaneamente di minimizzare lo spazio in eccesso. Grazie a questa scelta viene garantito che ogni *bucket* contenga un solo semaforo, anche nel caso in cui siano tutti attivi. Si è implementata la funzione `get(int *semAdd)` per il recupero di semafori dalla tabella hash: data la chiave del semaforo, il metodo indicizza la tabella basandosi sullo stesso criterio della *macro* `hash_add`. Trattandosi di un puntatore a intero, l'*hashing* delle chiavi viene eseguito con la *macro* `hash_ptr`. L'implementazione del resto delle funzioni è triviale e segue banalmente le indicazioni delle specifiche.

- `ns.c / ns.h`: la gestione dei *namespace* è stata implementata attraverso la matrice `nsd_t ns_Table` e i due array di `list_head`, `ns_Free_h` e `ns_Active_h`. La lunghezza degli array e la prima dimensione della matrice combaciano con la *macro* `NS_TYPE_MAX`, dichiarata nel file `pandos_types.h`: per introdurre un nuovo *namespace* sarà sufficiente definirlo incrementando di 1 il valore dell'ultimo *namespace* definito e aggiornare la macro `NS_TYPE_LAST` al *namespace* così definito.