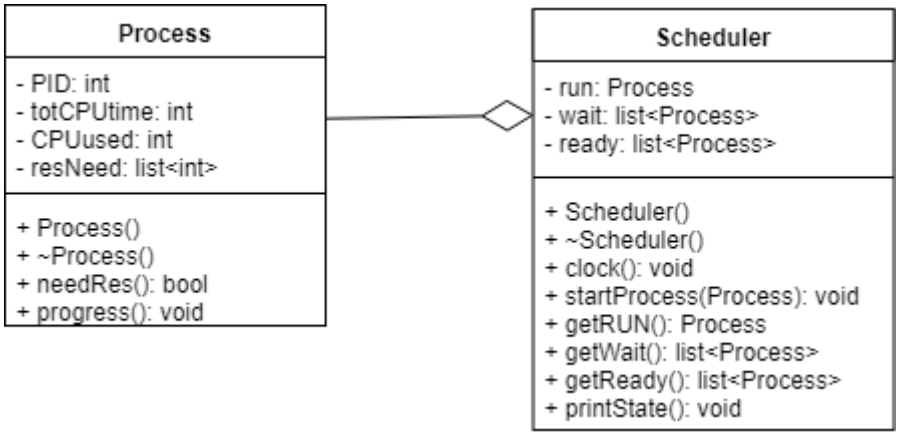
2018-05-16 – Simulazione - ***Short time scheduler - pasw01101***

Si vuole realizzare un simulatore semplificato di uno scheduler per l'assegnazione dei processi in un sistema monoprocessore.



Lo schema rappresenta una bozza di diagramma delle classi in cui sono rappresentati i processi e lo scheduler di gestione.

L'avanzamento dell'intero sistema avviene par passi ed è simulato inviando allo scheduler un messaggio di "clock" generato dall'input dell'applicazione.

La classe Process ha 4 attributi:

- PID rappresenta il Process IDentifier
- totCPUtime rappresenta il numero di passi di esecuzione per portare a compimento il processo
- CPUused rappresenta il numero di passi già eseguiti
- resNeed è una lista di interi che rappresenta i momenti in cui il processo necessita di una risorsa
    - esempio la sequenza di valori 8 5 6 rappresenta il fatto che dopo 8 passi il processo necessita di una risorsa poi necessita nuovamente di una risorsa dopo 5 passi e ancora dopo 6 passi
- il metodo progress viene attivato ad ogni passo di esecuzione del processo
- il metodo needRes restituisce il valore true se il processo necessita di una risorsa

La classe Scheduler ha 3 attributi

- run è un riferimento al processo attualmente in esecuzione
- wait è una lista di riferimenti a processi in stato di attesa per una risorsa
- ready è una lista di riferimenti a processi in stato di attesa
- i metodi getRun, getWait e getReady restituiscono il valore dei relativi attributi
- printState visualizza lo stato attuale degli attributi
- startProcess riceve un riferimento a un processo che viene posto in stato di ready
- clock rappresenta l'avanzamento di un passo che comporta la gestione del processo in stato di run che può avanzare, terminare in quanto completato, o essere posto in stato di wait o di ready
    - nel caso il processo in esca dallo stato di run deve essere prelevato un processo dalla coda di ready gestita con logica FIFO
- ogni n clock viene simulata l'assegnazione di una risorsa a un processo che viene selezionato casualmente, tolto dalla lista di wait e inserito nella coda di ready
- il quanto di time-sharing è equivalente a m cicli di clock

Aggiungendo i necessari attributi e metodi alle classi scrivere una applicazione che permetta di “creare” nuovi processi e di simulare l’evoluzione dell’intero sistema.

Impostare il “quanto di tempo di CPU” e la logica di “risveglio” di un processo in stato di wait utilizzando valori costanti in modo da poter configurare il sistema in modo da bilanciare il numero di processi nei vari stati di esecuzione.

È consentito l’utilizzo di strutture date e/o algoritmi presenti nella STL.

Un esempio del ciclo di evoluzione dell’applicazione potrebbe essere il seguente:

```
<create process p1>
<create process p2>
…
foreach <process>
   scheduler.<process execute>
do
{
        scheduler.clock()
        if ( m times clock)
                scheduler. awakeOneProcess()
        scheduler.printState()
        sleep (or wait for input)
} while true
```