[Esercizio4 (P&C).pdf](https://drive.google.com/open?id=1qqlFpG1mwTHkJsod17TVo_diJ5FlBWMQ)

Si implementi il classico problema produttori/consumatori con le seguenti modifiche:

* esiste la classe dei consumatori "speciali" che consumano due elementi anziché uno.
* se un consumatore speciale va in waiting allora da questo momento in poi al massimo 5 consumatori normali potranno consumare prima che almeno un consumatore speciale venga risvegliato.

Appunti iniziali:

* Prod, Cons, SpecialCons
* Se wait(SpecialCons), al max 5 Cons possono lavorare prima di signal(SpecialCons)

## Variabili:

full=0

free=N

i=0

j=0

k=0

h=0

waitP=0

waitC=0

waitCS=0

bonus=5 // Per implementare la priorità ai consumatori speciali

## Semafori:

semP=0

semC=0

semCS=0

mutexP=1

Si assume l’array inizializzato con numeri negativi

# Produttore

while(true) {

…

item=...;

wait(mutex);

if(free==0) { // Se l’array e’ pieno

waitP++;

wait(semP);

signal(mutex);

}

else { // Se l’array non e’ pieno, produce

while(buffer[i]<0)

i=(i+1)%N;

buffer[i]=item;

i=(i+1)%N;

free--;

full++;

if(waitC>0 && bonus>0) {

waitC--;

signal(semC);

}

}

}

## Variabili:

waitP // condivisa da tutti (mutex)

waitC // condivisa da tutti (mutex)

waitCS // condivisa da tutti (mutex)

i // condivisa tra produttori (mutexP)

j // condivisa tra consumatori (mutexC)

k // condivisa tra consumatori speciali (mutexCS)

full=0 // condivisa da tutti (mutex)

empty=N // condivisa da tutti (mutex)

## Semafori:

mutex=1

semP=1

semC=1

semCS=1

# Produttore

while(true) {

…

item=...;

wait(mutex);

if(empty==0) { // Se e’ pieno

waitP++;

signal(mutex);

wait(semP);

}

else { // Se non e’ pieno, produco

buffer[i]=item;

i=(i+1)%N;

signal(mutex);

}

}

# Consumatore

while(true) {

}

# ConsumatoreSpeciale

while(true) {

}

Con Depa:

## Variabili:

* waitProd=0
* waitCons=0
* waitConsSpec=0
* bonus=5 // Num max di consumatori che potranno consumare prima che almeno un ConsSpec venga risvegliato
* empty=N // Lunghezza buffer
* full=0

## Semafori:

* semProd=0
* semCons=0
* semConsSpec=0
* mutex=1

# Produttore

while(true) {

…

wait(mutex);

if(empty==0) { // Se array pieno

waitProd++;

signal(mutex);

wait(semProd);

wait(mutex);

}

// Produzione

empty--;

full++;

if(waitCons>0) { // Se c'è almeno un consumatore in attesa

waitCons--;

signal(mutex);

signal(semCons);

}

else

signal(mutex);

…

}

# Consumatore

while(true) {

…

wait(mutex);

if(full==0) { // Non c’e nessun elemento nell’array

waitCons++;

signal(mutex);

wait(semCons);

wait(mutex);

}

// Consumazione

full--;

empty++;

if(waitConsSpec>0) { // Se ci sono ConsSpec in attesa

bonus--;

if(bonus==0) { // Se hanno lavorato 5 Cons da quando ConsSpec e’ entrato in attesa

waitConsSpec--;

signal(mutex);

signal(semConsSpec);

}

else

signal(mutex)

}

else

signal(mutex);

…

}

# ConsumatoreSpeciale

while(true) {

}