Diomio

Esercizio 1

```
monitor alvr {
    condition c[MAX]; //MAX massimo numero di processi
    int w[MAX];

    entry void least(int n) {

        int liberabili = 1;

        int maxindex = -1;

        for (int i = 1; i <= MAX; i++) {
            liberabili += w[i];

            if (liberabili >= maxindex)
                maxindex = i;
        }

        for (int i = 1; i <= maxindex; i++)
            c[i].signal();

        if (maxindex < n) {
            w[n]++;
            w[n].wait();
            w[n]--;
            if (w[n] > 0)
                w[n].signal();
        }
    }
}
```

Esercizio c.2

```
void chained_send(T msg, lit_of_pids dests) {
        Coppia a = <T msg, dest.next>
        ssend(a, dest.value);
}

T chained_recv(void) {
    <msg, dests> = srecieve();
    Coppia a = <msg, dests.next>
    ssend(a, dests.value);
    return msg;
```

```
}
```

soluzione discussa con il prof surante il ricevimento del 13 gennaio 2022 :

```
void chained_send (T msg, list_of_pids dests){
  ssend(<msg, dests.tail()>, dests.first());
  //first() restiuisce il primo della lista
  //tail() restituisce la lista tranne il primo elemento
  //(se la lista ha un solo elemento tail è vuoto)
}

T chained_recv(void){
  <msg, dests> = sreceive(ANY);
  if !dests.empty()
    ssend(<msg, dests.tail()>, dests.first())
 return msg
}
```

Eserczio g.3

1,2,3,4,1,2,5,1,2,3,4,5

3 + 1|2|3| -> 4|2|3 -> 4|1|3 -> 4|1|2 -> 5|1|2 -> 5|3|2 -> 5|3|4

4 + 1|2|3|4 -> 5|2|3|4 -> 5|1|3|4 -> 5|1|2|4 -> 5|1|2|3 -> 4|1|2|3 -> 4|5|2|3

parte 1

`1, 2, 3, 4, 5, 1, 2, 3, 6, 1, 2, 3, 4, 5, 6`

3 + 1|2|3|4 -> 5|2|3|4 -> 5|1|3|4 -> 5|1|2|4 -> 5|1|2|3 -> 6|1|2|3 -> 6|4|2|3 -> 6|4|5|3

4 + 1|2|3|4|5 -> 6|2|3|4|5 -> 6|1|3|4|5 -> 6|1|2|4|5 -> 6|1|2|3|5 -> 6|1|2|3|4 -> 5|1|2|3|4 -> 5|6|2|3|4 ### parte 2

```
lista = []
for i to n:
    lista.append(i)

for i to n-2:
    lista.append(i)

lista.append(n+1)

for i to n+1:
    lista.append(i)
```

Spiegazione

- caso n-1:
  - Da 1 a n impiega n page fault
  - in memoria ci sono le pagine da 2 a n
  - faccio altri $n-2$ page fault perchè ad ogni page fault tolgo la pagina a cui farei l'accesso subito dopo
  - un'altro page fault $(2n-1)$
  - in memoria ci sono da 1 a $n-2$ e $n+1$
  - faccio altri 2 page fault (per $n-1$ e n)
  - l'accesso a $n+1$ hitta perchè non è stato sovrascritto
  - totale page fault: $2n+1$
- caso n:
  - Da 1 a n impiega n page fault
  - in memroia ci sono pagine da 1 a n
  - faccio un'altro page fault $n+1$
  - in memoria ci sono da 2 a $n+1$
  - faccio $n+1$ page fault perchè ogni page fault nuovo rimpiazzo quella che cerco subito dopo
  - totale page fault: $2n+2$

Esercizio g.4

a) Lo scheduler sceglie fra i processi ready quale porre in esecuzione. Ma cosa succede quando la coda dei processi ready è vuota?

   Se il processo corrente non è stato interrotto da una chiamata bloccante (ad esempio l'interval timer in un Round Robin) il processo corrente viene messo nella coda ready per poi essere subito scelto come processo da eseguire. Altrimenti se il processo è bloccato e abbiamo uno schedulere con priorità viene caricato il processo IDLE che ha priorità minima e non fa nulla. Il SO sta aspettando che un processo si svegli per riprendere l'esecuzione di quello.

b) Quali vantaggi offre RAID5 rispetto a RAID1?

   RAID5 ha molta meno ridondanza rispetto a RAID1, infatti RAID1 raddoppia la necessità dei dischi (duplicandone il contenuto) mentre RAID5 richiede un solo disco di ridondanza, "sparso fra i vari dischi". Inoltre in RAID1

c) Se per effetto di un guasto o di un bug il numero di hard link di un i-node è errato, quali conseguenze possono esserci?

d) Quali eventi causano la valutazione dell'algoritmo del banchiere? Cosa si fa se lo stato risulta unsafe e cosa invece se è safe?