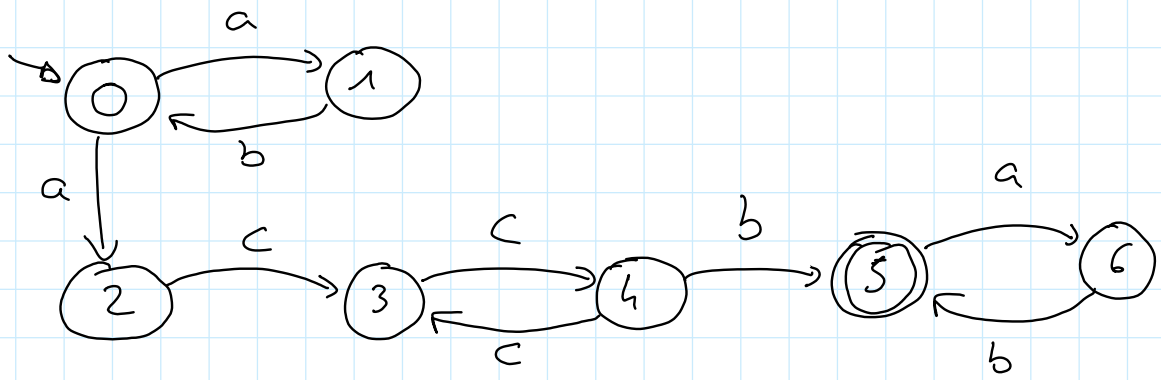
SOLUZIONE DELL'ESERCITAZIONE DEL 27/10/17

1) Si dimostri se è regolare o meno il linguaggio definito dalla seguente grammatica:

$$S \to abS \mid Sab \mid aXb$$
$$X \to ccX \mid cc$$

$$L = \{(ab)^n a (cc)^k b (ab)^2 \mid n, z \geq 0,\ k > 0\}$$



REGOLARE

2) $L = \{ a^m b^k c^t \mid m, k, t > 0 \wedge m \leq k+t \}$

PUMPING LEMMA

Dato $m \in \mathbb{N}$ scelgo

$w = a^{m+1} b^m c$ (potevo fare scelte diverse)

Tutte le suddivisioni possibili sono: $|xy| < m$, $y \neq \varepsilon$

$x = a^t$ $t \geq 0$ $t < m+1$

$y = a^s$ $s > 0$ $s \leq m+1$

$z = a^k b^m c$ $k = (m+1) - (s+t)$

$m+1 = k+s+t$

xy^iz $i = 0$ $xz = a^t a^k b^{m-1} c \in L$ ✗

$i = 2$ $xyyz = a^t a^s a^s a^k b^{m-1} c$

$t+s+s+k = m+1+s > m+1$

$s > 0$

$xy^2z \notin L$

QUINDI IL LINGUAGGIO NON È REGOLARE

$S \to aSc \mid Sc \mid aXc$

$X \to aXb \mid Xb \mid ab \mid b$

3) $\exists i \in [0, dim) \cdot a[i] = \#\{ j \mid j \in [0, dim) \wedge a[j] > 0 \}$

$[3, 7, -1, 2]$ V

```
int contaPositivi (int a[], int dim) {
    int cont=0;
    for (int i=0; i<dim; i++) {
        if (a[i]>0) cont++;
    return cont;
}

int check (int a[], int dim) {
    int i=0;
    int Trovato=0;
    while (i<dim && !Trovato) {
        if (a[i]== contaPositivi (a, dim))
            Trovato=1;
    }
    return Trovato;
}
```

OK !! VEDIAMO UN'ALTRA SOLUZIONE (PIU' EFFICIENTE)

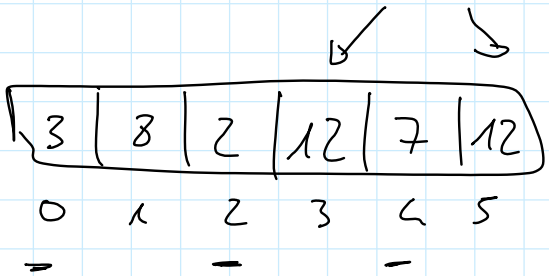
```
int check (int a[], int dim) {
    int i=0;
    int Trovato=0
    int positivi= contapositivi (a, dim);
    while (i<dim && !Trovato) {
        if (a[i]== positivi) Trovato=1;
        i++;
    }
    return Trovato;
}
```

LA CHIAMO UNA VOLTA SOLA FUORI DAL CICLO

UN'ALTRA SOLUZIONE ANCORA :

```
int check (int a[], int dim) {
    int cont=0;
    int i=0;
    int Trovato=0;
    for (int j=0; j<dim; j++)
        if (a[j]>0) cont++;
    while (i<dim && !Trovato) {
        if (a[i]==cont) Trovato=1;
        i++;
    }
    return Trovato;
}
```

4) $\#\{ i \mid i \in (0, dim) \wedge a[i] = \sum_{\substack{j \in (0,dim) \\ j\%2=0}} a[j] \}$



PARTO DEFINENDO UNA FUNZIONE PER $\sum_{\substack{j \in (0,dim) \\ j\%2=0}} a[j]$

```
int sommapospari (int a[], int dim) {
    int somma=0;
    for (int i=0; i<dim; i++)
        if (i%2==0) somma += a[i];
    return somma;
}
```

ALTRO MODO (PIU' EFFICIENTE):

```
int sommapospari (int a[], int dim) {
    int somma=0;
    for (int=0; i<dim; i+=2)
        somma += a[i];
    return somma;
```

}

ORA DEFINISCO UNA FUNZIONE PER

$$\#\{i \mid i \in [0, dim) \wedge a[i] == Sommapospari(a, dim)\}$$

COME NELL'ESERCIZIO PRECEDENTE EVITO DI CHIAMARE Sommapospari DENTRO AL CICLO (PERDEREI TEMPO A CALCOLARE SEMPRE LO STESSO VALORE)

```
int conta (int a[], int dim) {
    int c=0;
    int somma = sommapospari (a, dim);
    for (int i=0; i<dim; i++)
        if (a[i]==somma) c++;
    return c;
}
```

SOLUZIONE ALTERNATIVA :

```
int conta (int a[], int dim) {
    int somma = 0;
    int c = 0;
    for (int i=0; i<dim; i+=2)
        somma += a[i];
    for (int i=0; i<dim; i++)
        if (a[i] == somma) c++;
    return c;
}
```