

# Tolerância a falhas via Supervisor trees

Charlotte Lorelei Oliveira

# Charlotte Lorelei Oliveira



2013 ~ 2014 - PHP  
2015 ~ hoje - Elixir

Twitter: [@umamaistempo](https://twitter.com/umamaistempo)  
Linkedin: [/in/umamaistempo](https://www.linkedin.com/in/umamaistempo)



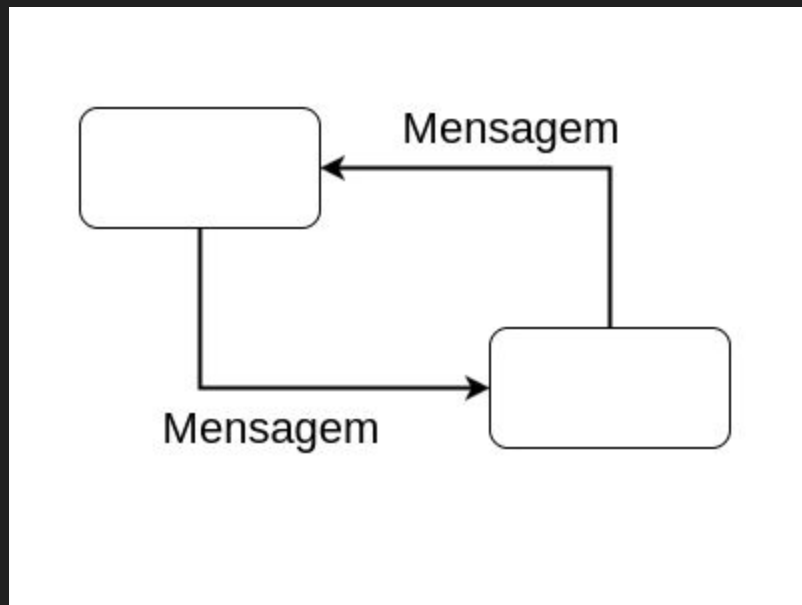
# Hacker Experience 2

[www.hackerexperience.com](http://www.hackerexperience.com)

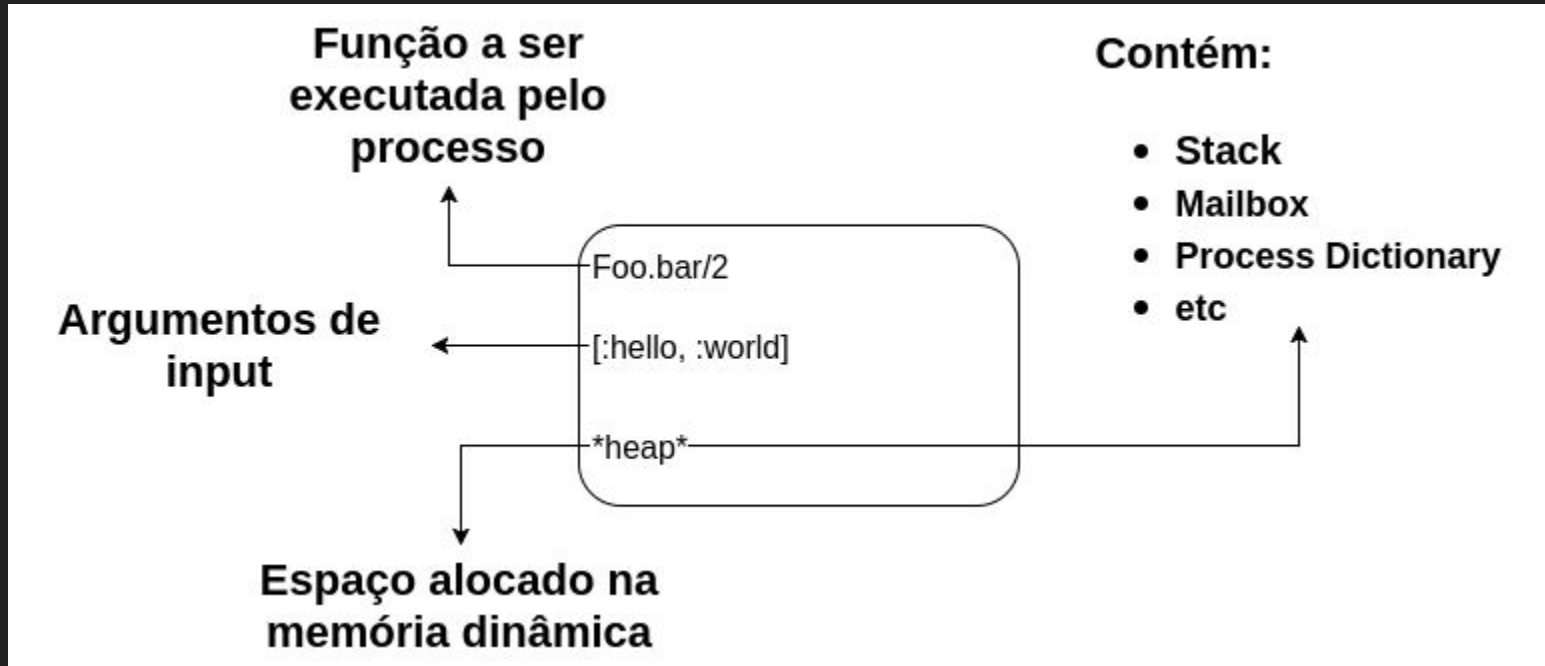
# Tópicos

1. O que é um processo
2. Como funciona o scheduler da BEAM
3. Filosofia Erlang para erros
4. O que é um Supervisor
5. Estratégias com Supervisors

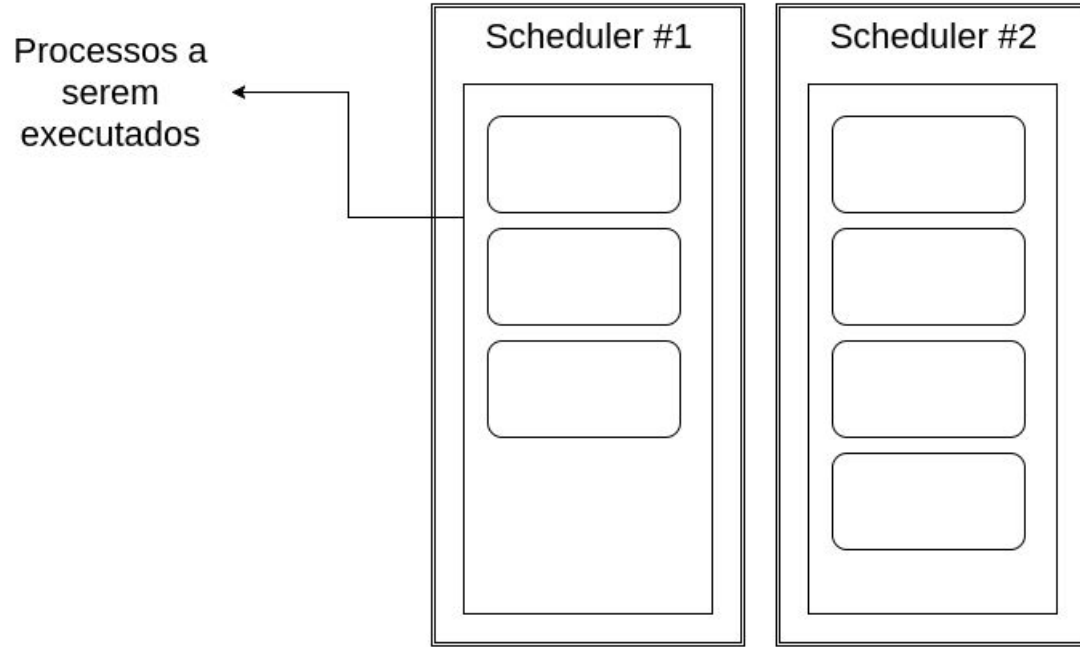
# O que é um processo



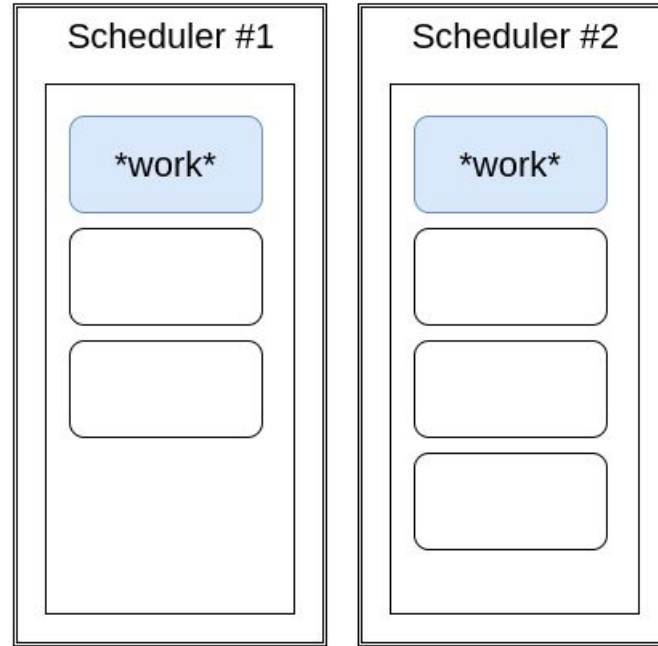
# A anatomia de um processo



# Escalonamento de processos

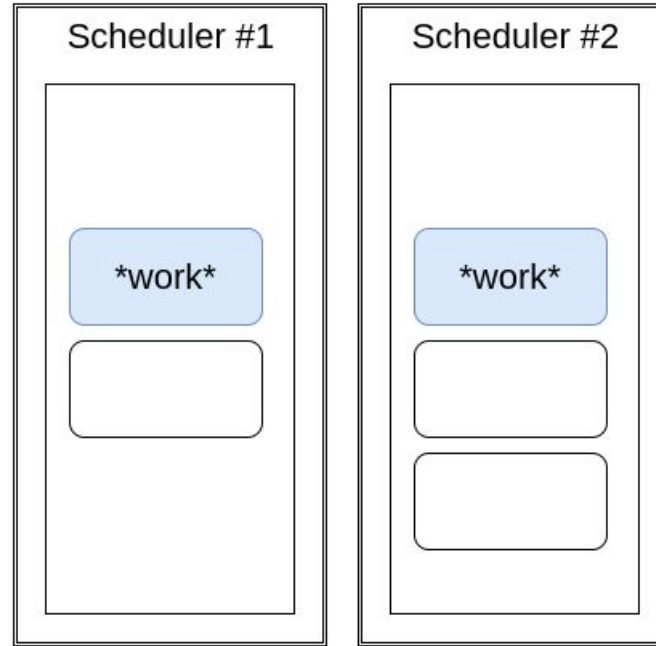


# Escalonamento de processos

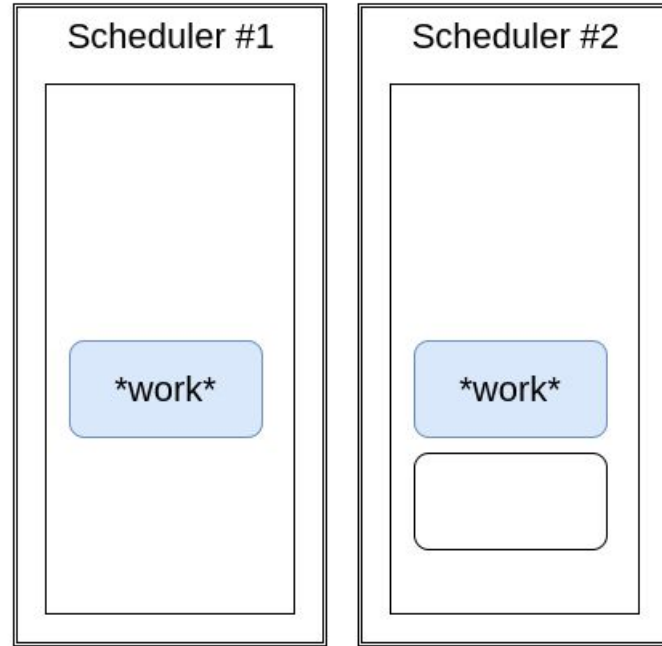




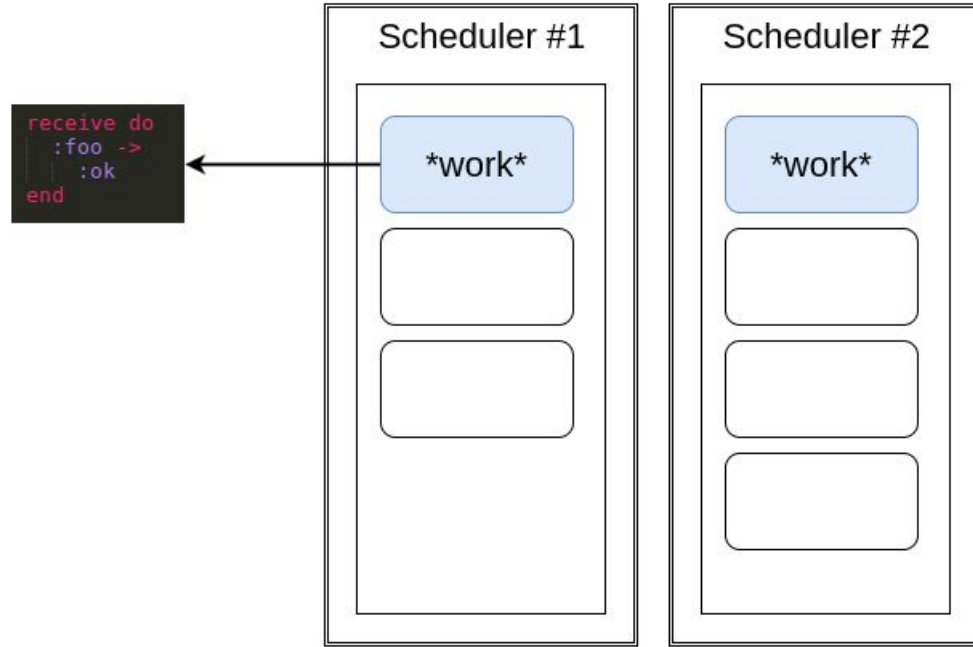
# Escalonamento de processos



# Escalonamento de processos

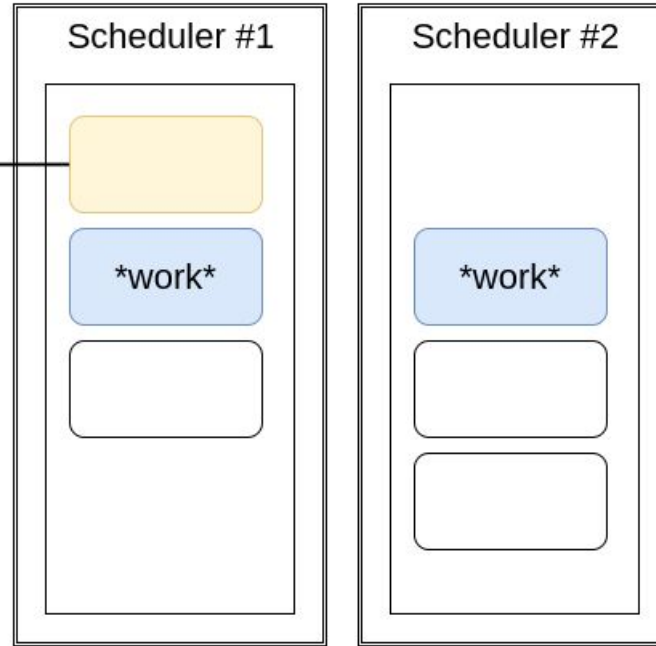


# Escalonamento de processos

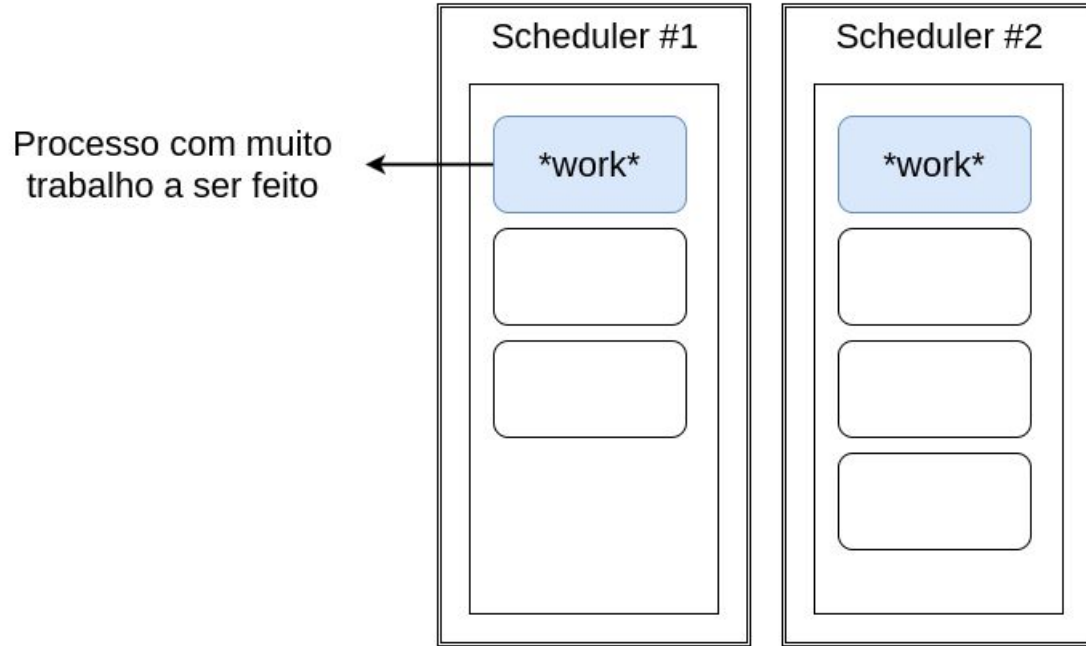


# Escalonamento de processos

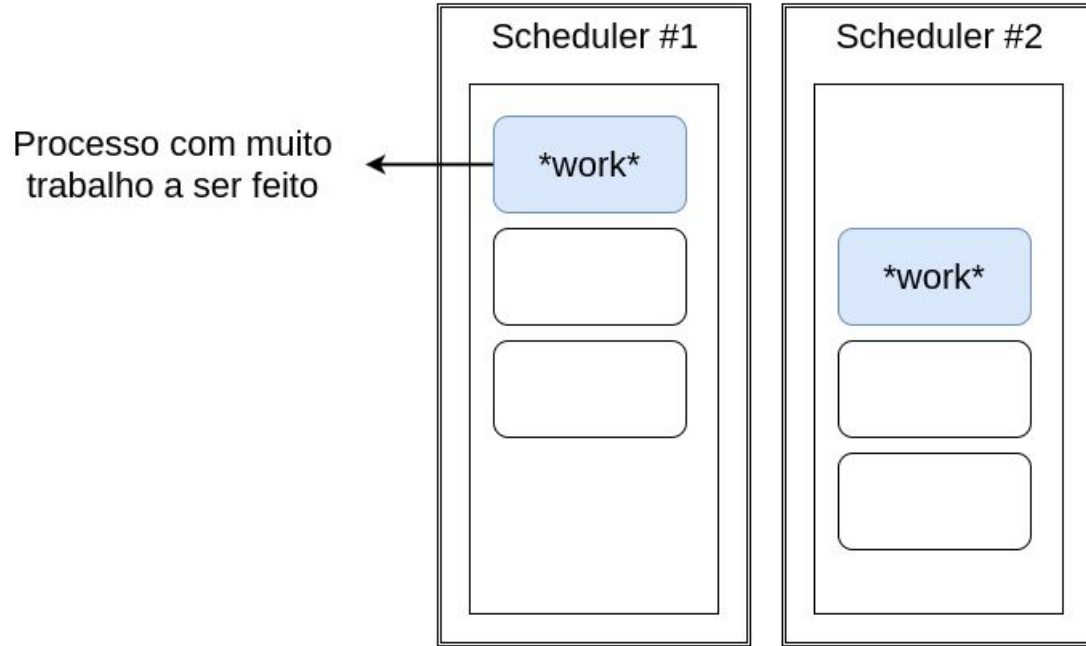
Processo a espera de mensagem na mailbox.  
Não é processado até que uma mensagem seja adicionada à sua mailbox



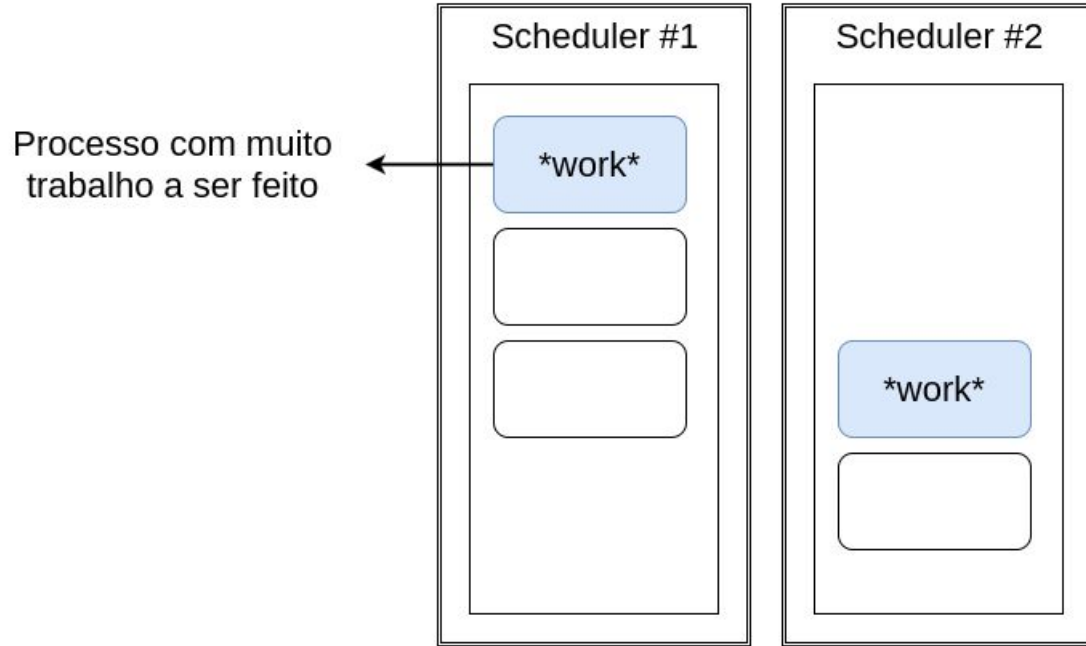
# Escalonamento de processos



# Escalonamento de processos

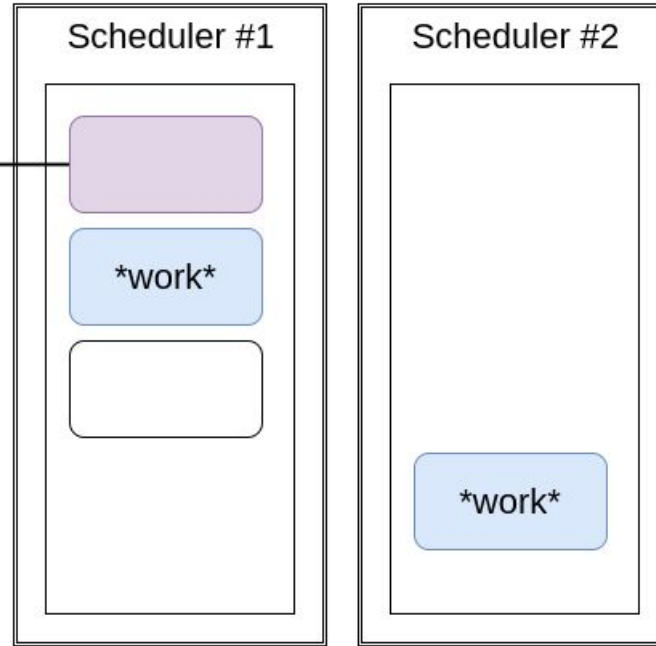


# Escalonamento de processos



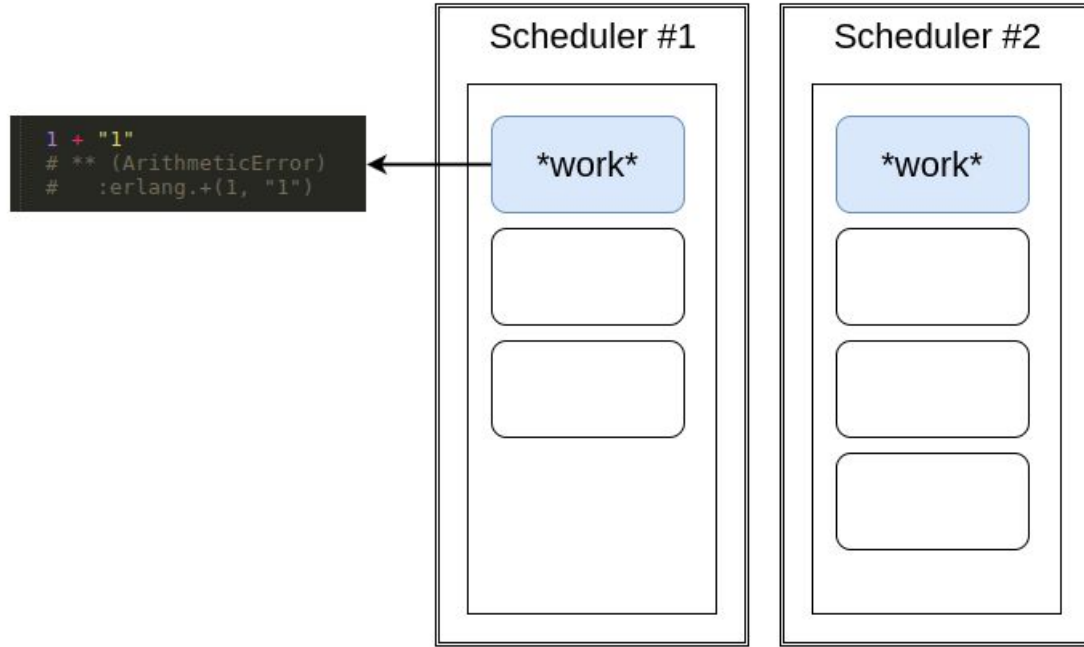
# Escalonamento de processos

Processo é posto em espera permitindo o escalonador trabalhar nos outros processos enfileirados

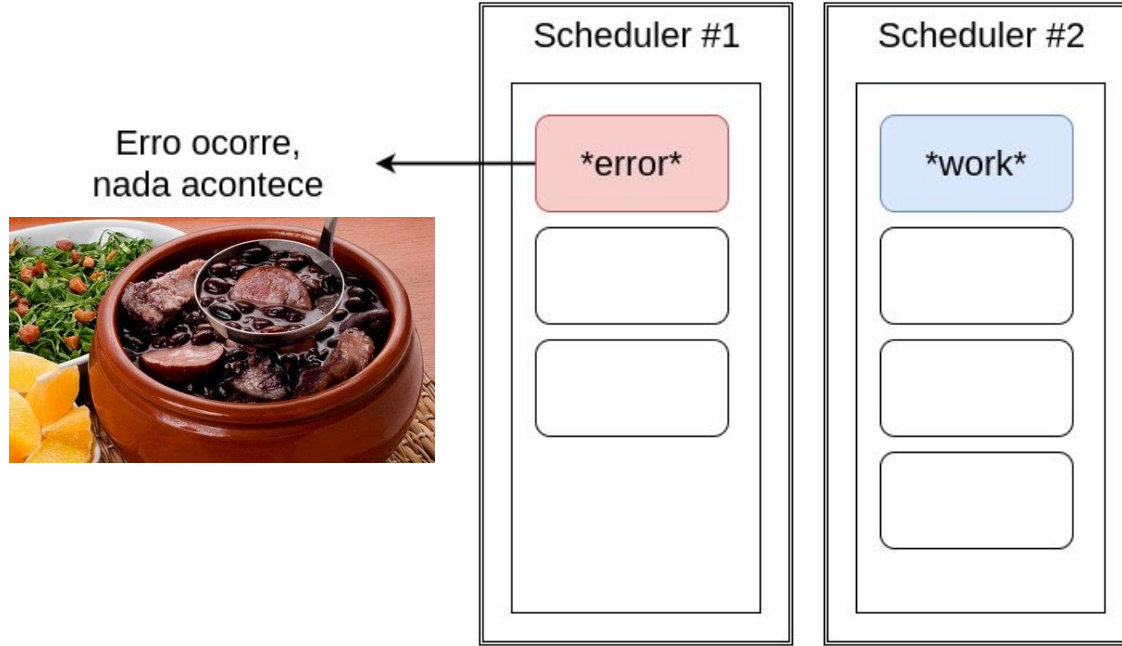




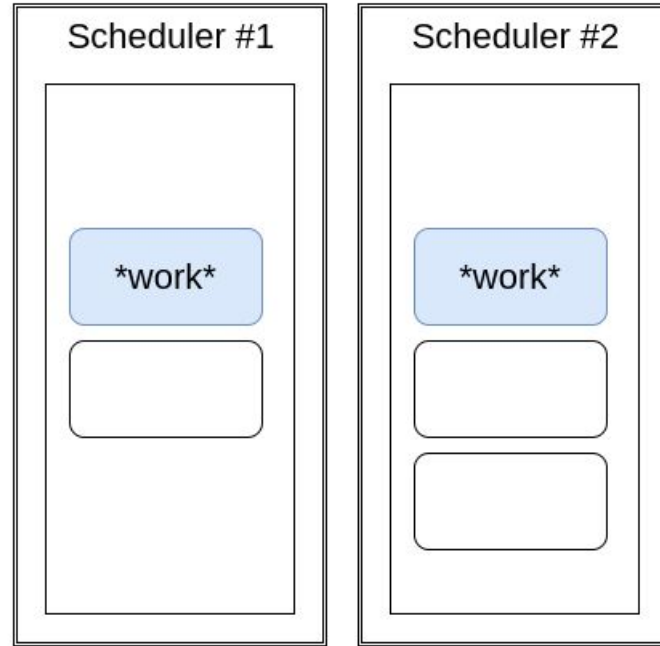
# Escalonamento de processos



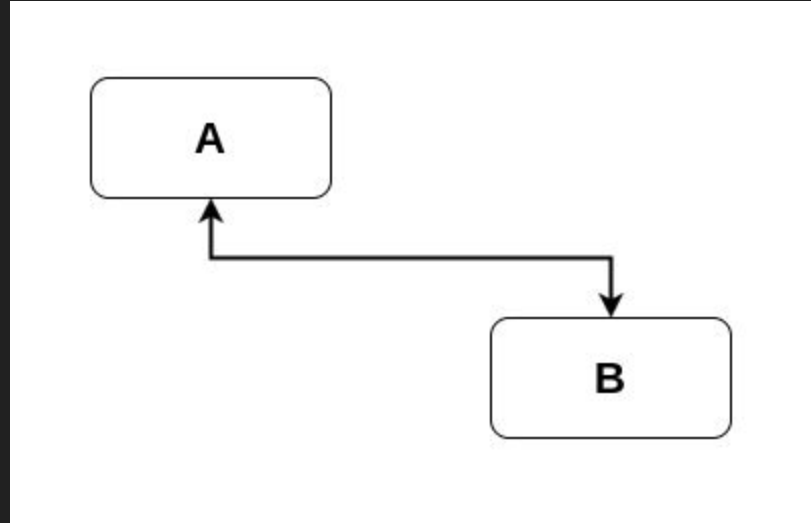
# Escalonamento de processos



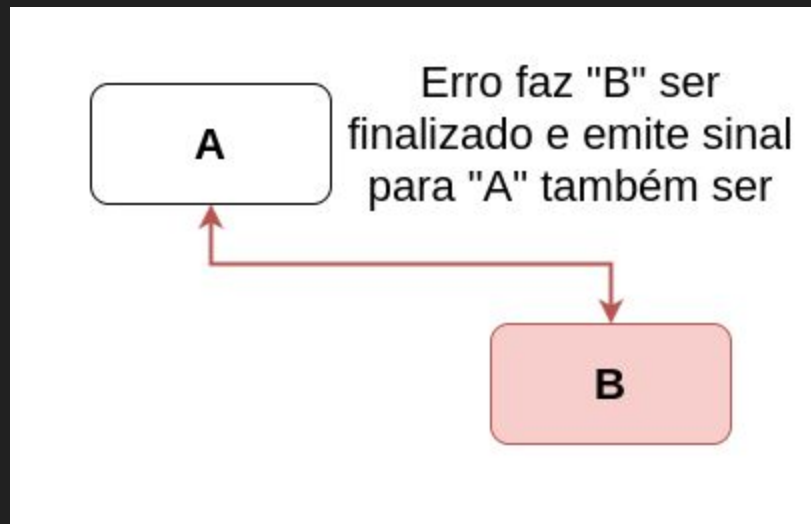
# Escalonamento de processos



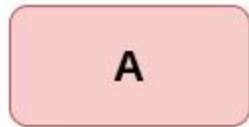
# Process.link



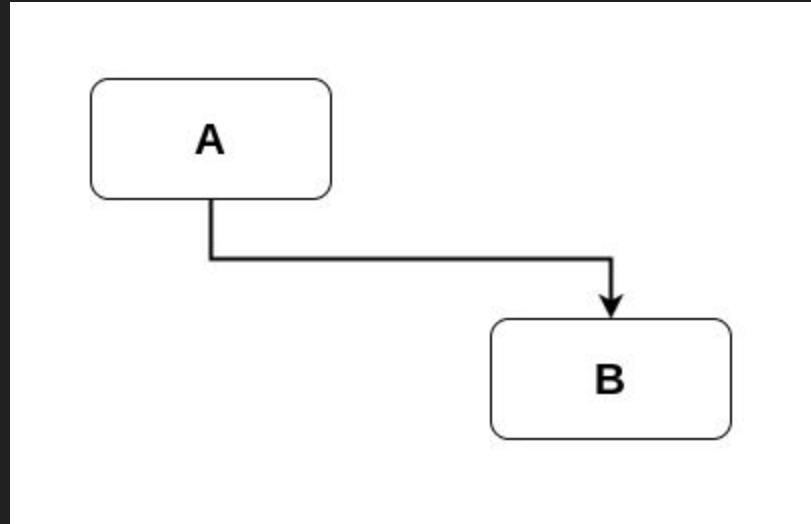
# Process.link



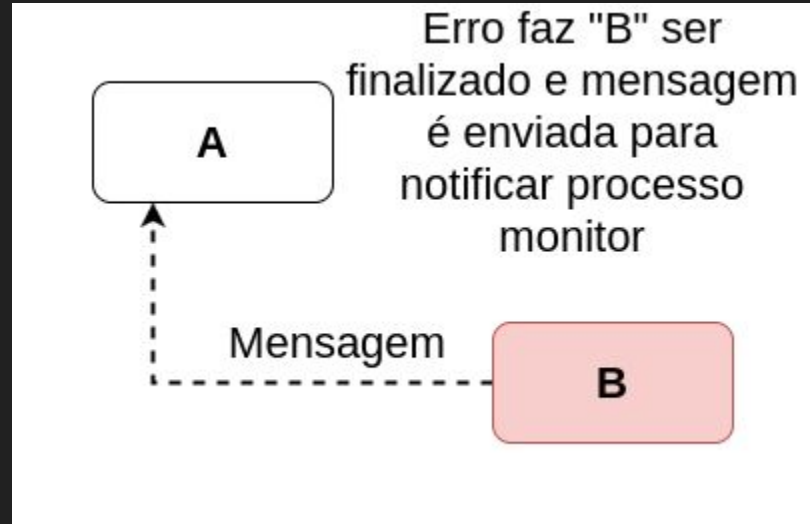
# Process.link



# Process.monitor



# Process.monitor





# A filosofia Erlang em error handling

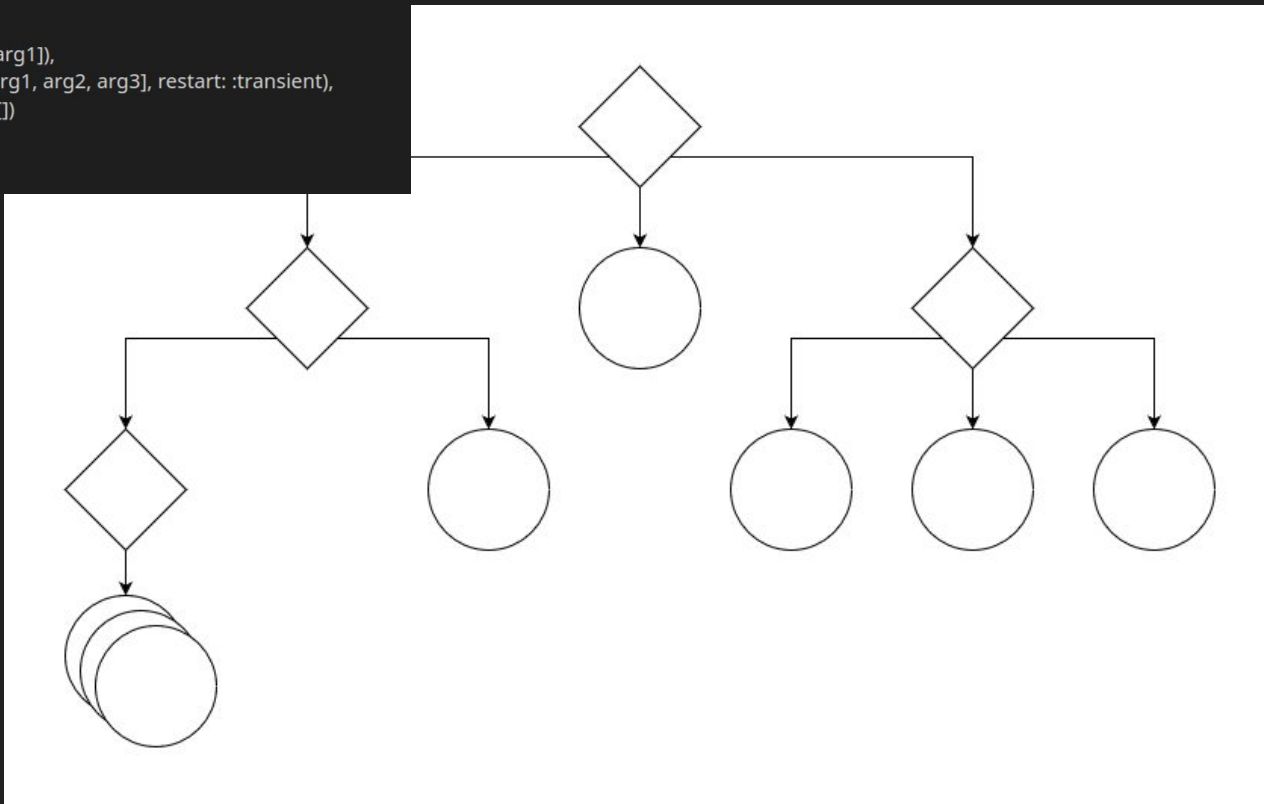
## **4.3 Error handling philosophy**

Error handling in Erlang is radically different to error handling in most other programming languages. The Erlang philosophy for handling errors can be expressed in a number of slogans:

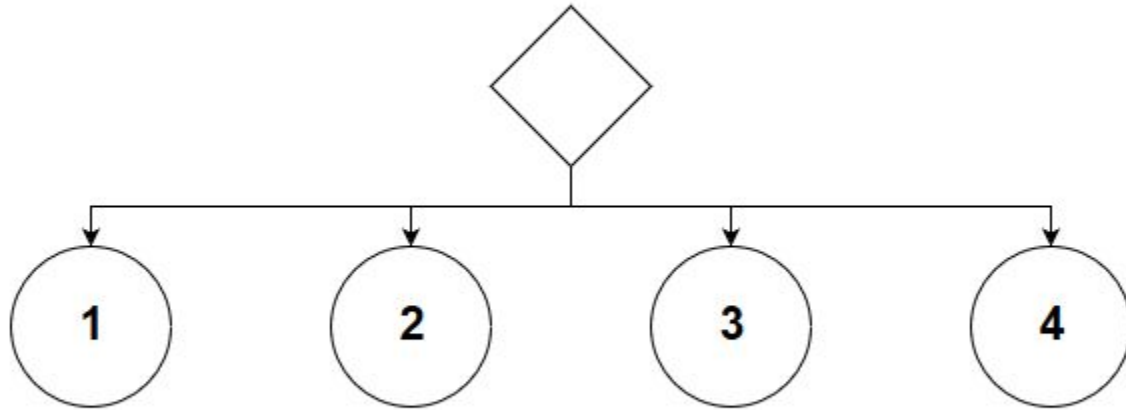
- Let some other process do the error recovery.
- If you can't do what you want to do, die.
- Let it crash.
- Do not program defensively.

# O que é um Supervisor

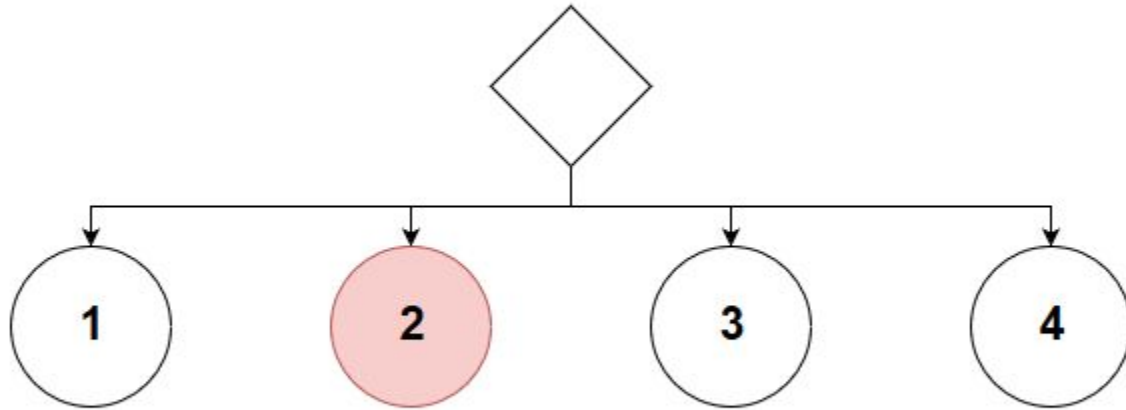
```
children = [  
  supervisor(First, [arg1]),  
  worker(Second, [arg1, arg2, arg3], restart: :transient),  
  supervisor(Third, [])  
]
```



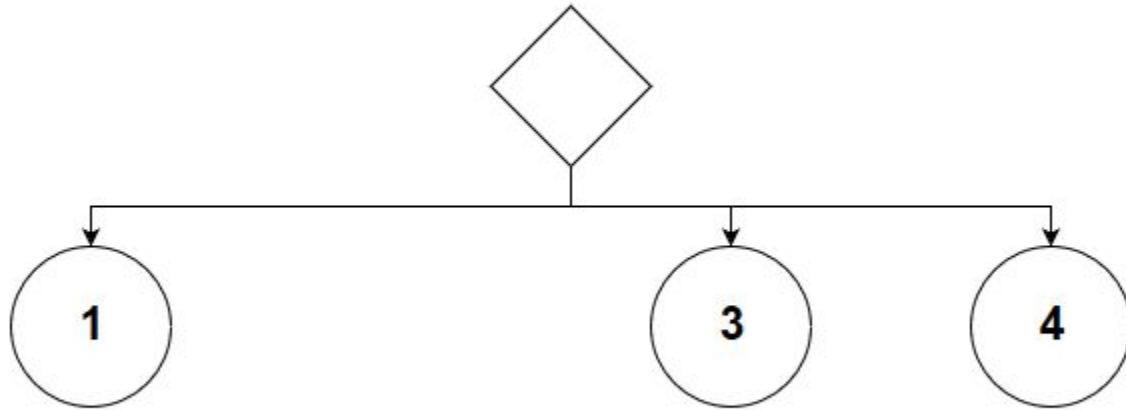
# Estratégia “one for one”



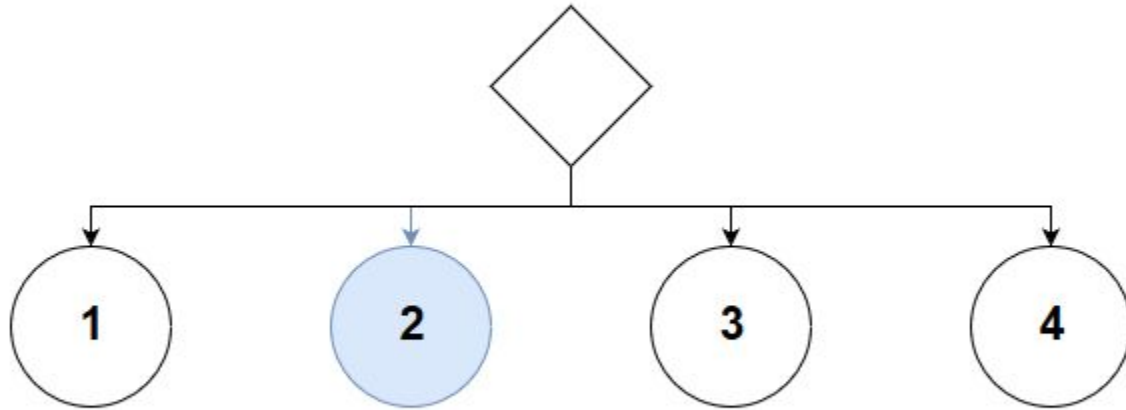
# Estratégia “one for one”



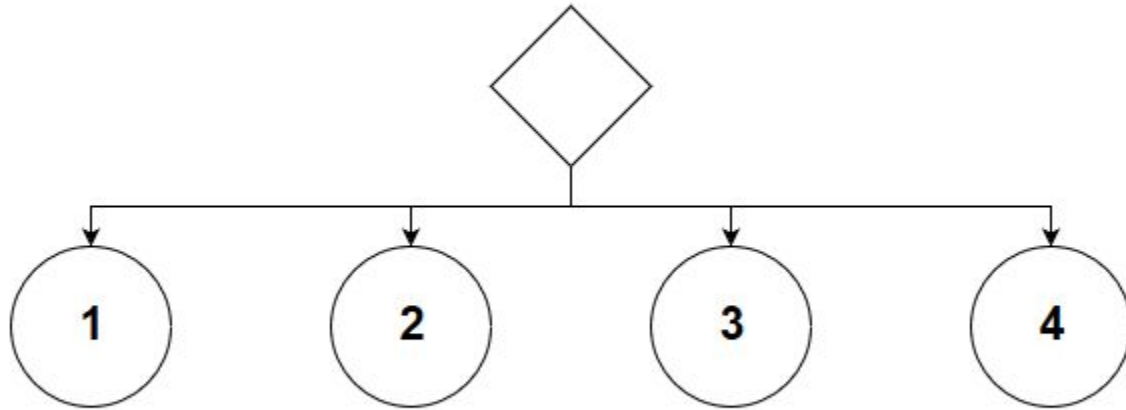
# Estratégia “one for one”



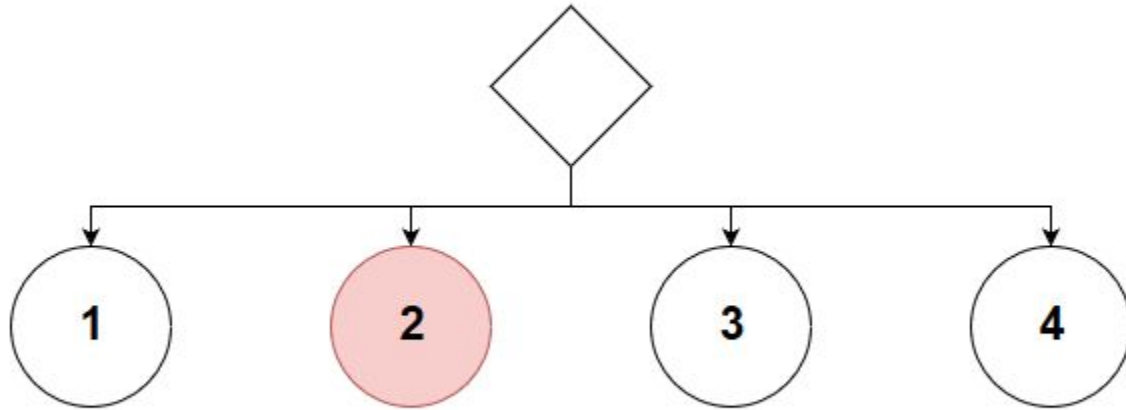
# Estratégia “one for one”



# Estratégia “rest for one”

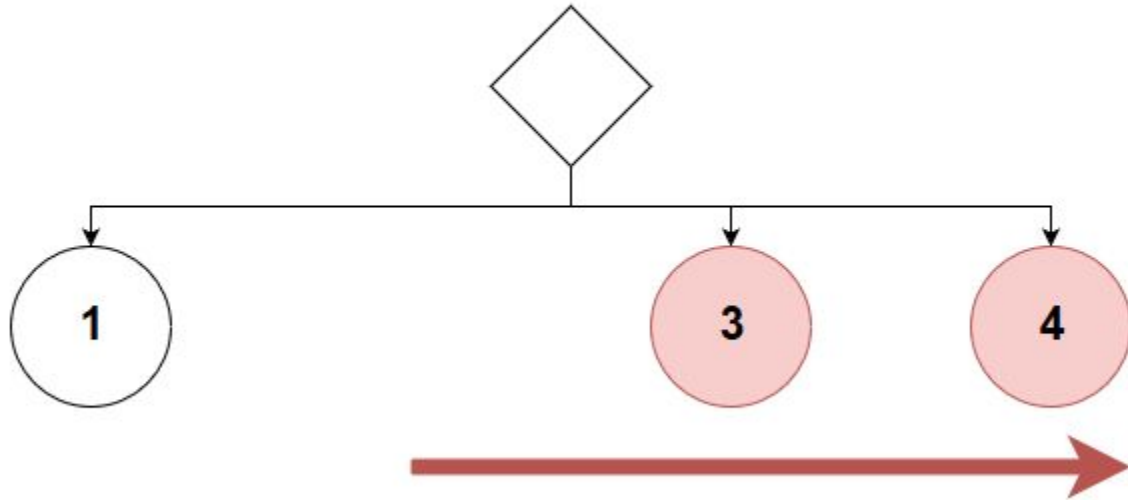


# Estratégia “rest for one”

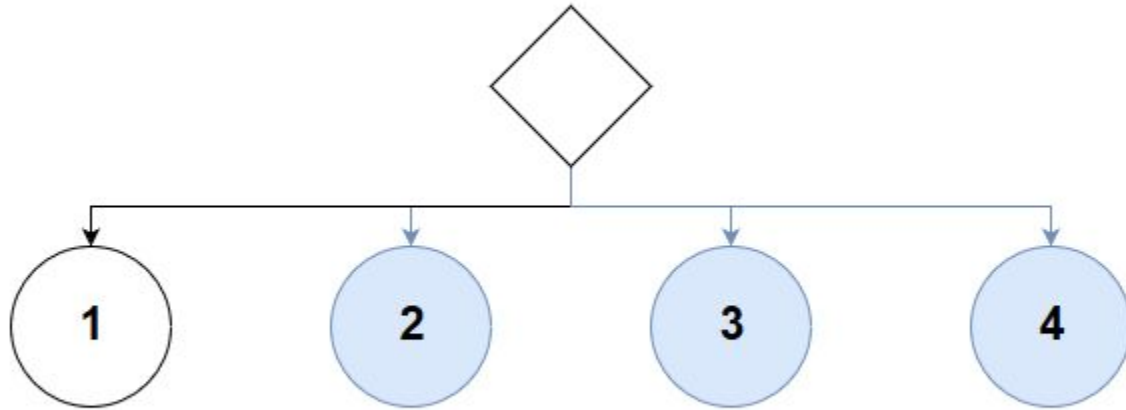




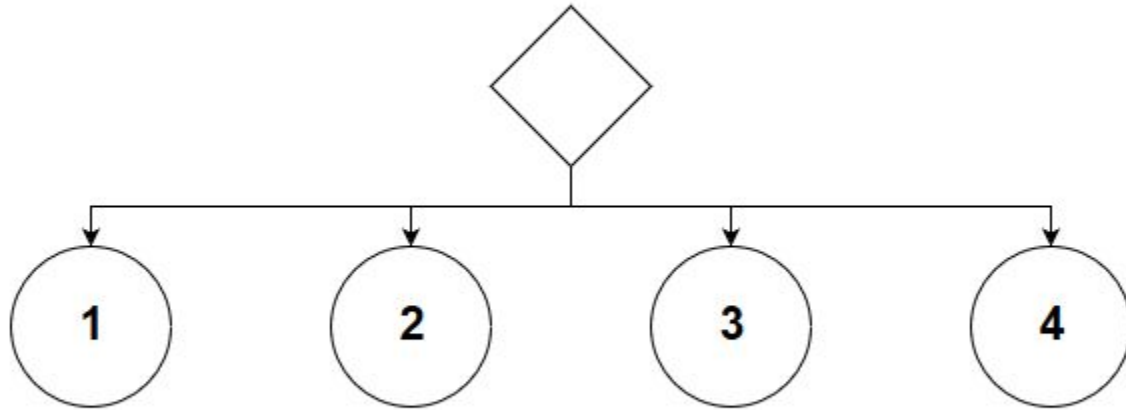
# Estratégia “rest for one”



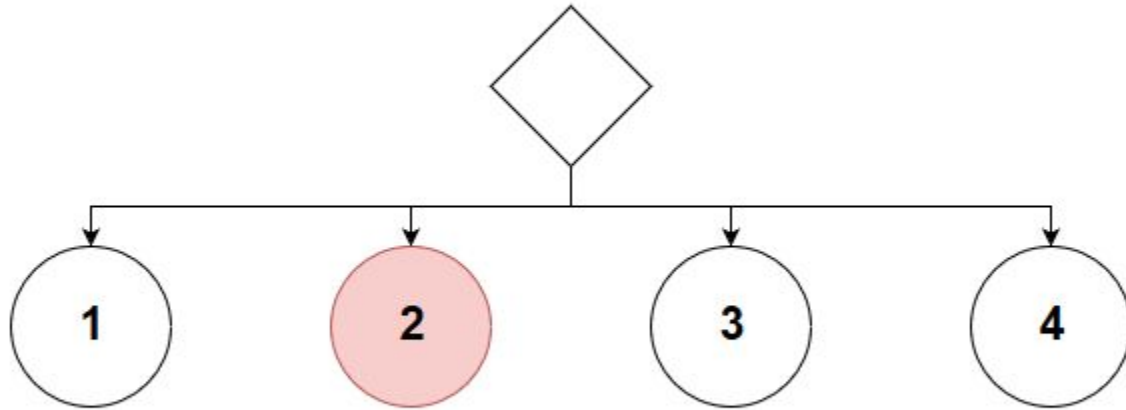
# Estratégia “rest for one”



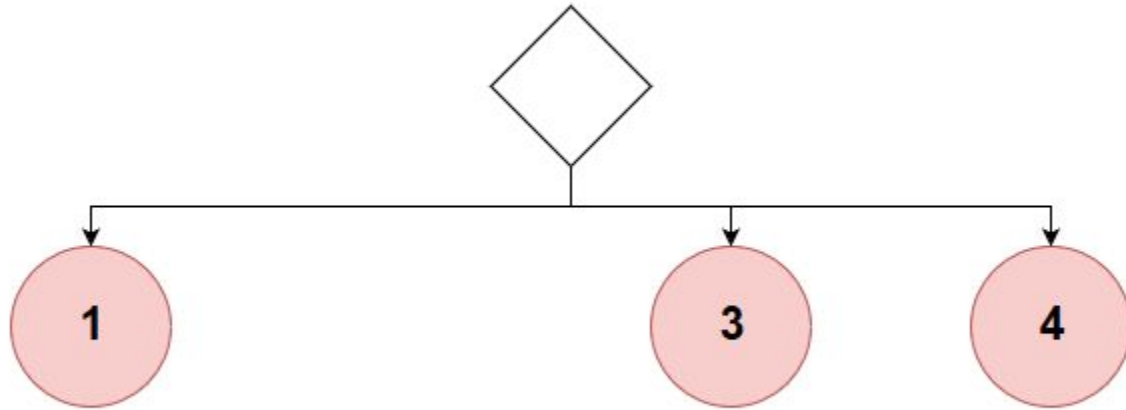
# Estratégia “all for one”



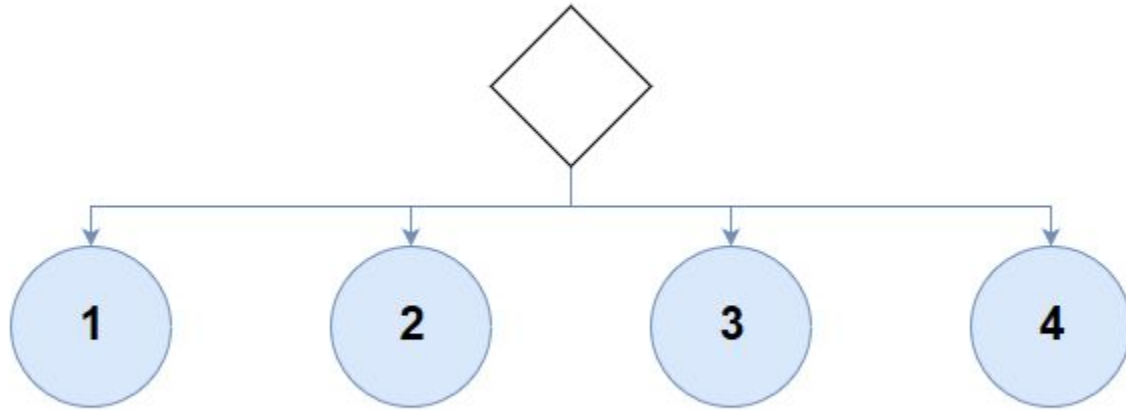
# Estratégia “all for one”



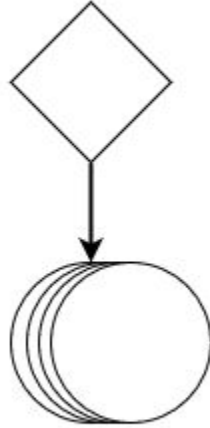
# Estratégia “all for one”



# Estratégia “all for one”



# Estratégia “simple one for one”



# Fim

Para qualquer pergunta, assuma  
que a resposta seja “sim”



**Hacker Experience 2**

MMO cyberpunk de hacking  
[www.hackerexperience.com](http://www.hackerexperience.com)

Twitter: [@umamaistempo](https://twitter.com/umamaistempo)  
Linkedin: [/in/umamaistempo](https://in.linkedin.com/company/umamaistempo)