



A arte do debug

boss



Quem somos nós?




Clarissa



Carla R.



Agenda

- A arte de saber debugar
 - Classes de problemas/ defeitos que causam bugs
 - Estratégias de debug
 - Backtracking
 - Simplificação do problema
 - Patinho de borracha
 - Como evitar bugs?
 - Tarefa da semana
 - Prática!
- 

A arte de saber debugar

(E o que ninguém te conta)



A arte de debugar

- Todo software está sujeito a bugs (e muitos)
- Debugar te ensina sobre o código
- Saber debugar te dá autonomia para desenvolver em qualquer ambiente!



Antes de começar...

Tente reproduzir o bug sozinha para
observar o comportamento do
software



Classes de problemas/ defeitos



Erros de sintaxe ou de tipo

```
python calculation.py
Traceback (most recent call last):
  File "calculation.py", line 2, in <module>
    result = a/10
TypeError: unsupported operand type(s) for /: 'str' and 'int'
```

```
python calculation.py
File "calculation.py", line 4
    def result(a)
                ^
SyntaxError: invalid syntax
```


Problemas de digitação ou outras falhas simples

```
calculation.py
1  a = 10
2  b = 2
3
4  ✓ def divide(a, b):
5      ... result = a/b
6      ... return result
7
8  result(b, a)
```

Problemas de implementação

(normalmente associados a implementação da lógica estar incorreta)

Implementação de Bhaskara

calculation.py

```
1 import math
2
3 a = 5
4 b = 6
5 c = 5
6
7 primeira_raiz = b**2 - math.sqrt(4 * a * c)/2 * a
8 segunda_raiz = b**2 + math.sqrt(4 * a * c)/2 * a
```

A falta de
parênteses causa
erro no resultado

Problemas de lógica

(a implementação condiz com a lógica pensada, mas a lógica é falha)

Implementação de Bhaskara

```
calculation.py
1  import math
2
3  a = 0
4  b = 6
5  c = 5
6
7  primeira_raiz = (b**2 - math.sqrt(4 * a * c))/2 * a
8  segunda_raiz = (b**2 + math.sqrt(4 * a * c))/2 * a
```

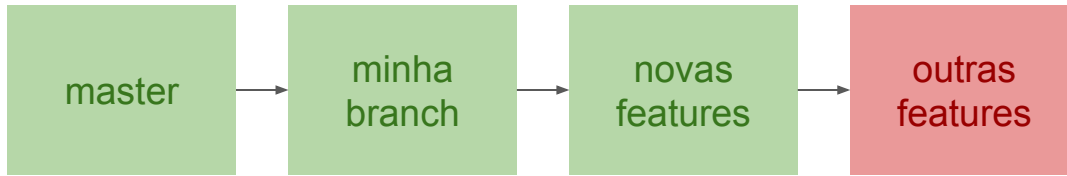
A implementação condiz com a lógica, mas faltou pensar nos "corner cases"



Estratégias de debug



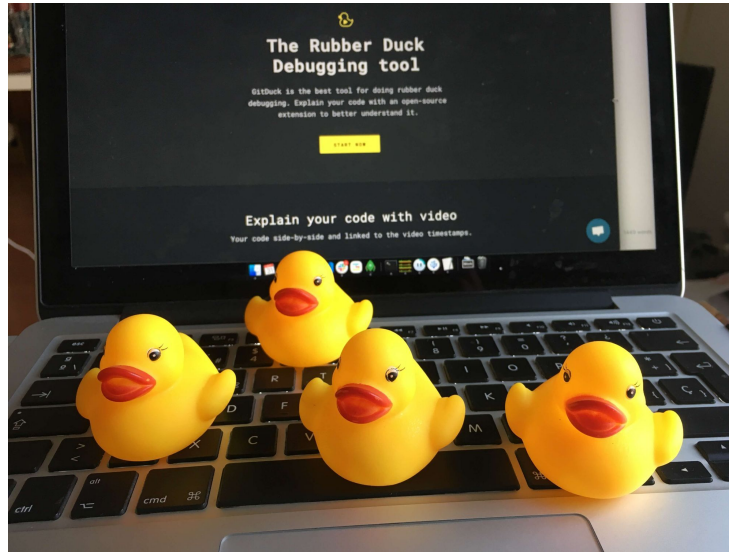
Backtracking



Simplificação do problema




Patinho de borracha






Como prevenir bugs

- Planejar bem o código antes de escrever
 - Testar pequenos trechos de código em vez de escrever tudo de uma vez e testar só no fim
 - Escrever testes automatizados
- 



Dicas importantes

- O bug pode estar em um lugar que você não espera
 - Se pergunte onde o bug **não** está
 - Veja o que tem em cada dado da sua solução
 - Tenha certeza de que seu código está atualizado com a branch, assim como a sua build
 - Tire um tempo para descansar
- 



Tarefa da semana

- Concluir a atividade da semana passada para as duplas que não concluíram ainda
 - Aproveitem o tempo para terminar outras tarefas passadas também
 - Mandem PRs do que vocês fizeram de tarefas passadas para podermos revisar e dar feedbacks =)
- Buscar issues no repositório original do boilerplate para resolver



Licença

Estes slides são concedidos sob uma Licença Creative Commons. Sob as seguintes condições: **Atribuição, Uso Não-Comercial e Compartilhamento pela mesma Licença.**

Mais detalhes sobre essa licença em: creativecommons.org/licenses/by-nc-sa/3.0/



Obrigada!

Prontas para debugar na prática?



boss



<https://github.com/orgs/BOSS-BigOpenSourceSister>