

Pesquisa em Depuração de Software



O que é depuração?

The Google logo is displayed in its standard multi-colored font.

Search bar containing the text "depurador". To the right of the text are icons for clearing the search (X), keyboard input, and voice search (microphone).

Pesquisa Google

Estou com sorte

depurador



Depuração

- Tornar puro, retirando as impurezas; limpar: depurar o azeite, depurar os costumes
- Sinônimo de limpar, purgar, purificar, aprimorar, aperfeiçoar

A Secretaria de Informação e Comunicação (Sicom) da Presidência do Paraguai anunciou nesta sexta-feira a criação na terceira semana deste mês do Conselho de Radiodifusão, que buscará a **depuração** do sistema de emissoras do país para "eliminar" as que funcionam de maneira ilegal.

Folha de S.Paulo, 08/01/2010

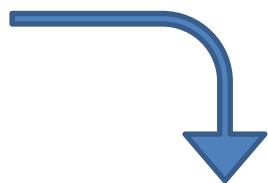
Segundo o presidente da Câmara, Sarney tem que tomar medidas que visem "à **depuração** dos costumes" no Senado.

Folha de S.Paulo, 03/07/2009

Depuração de Software: conceitos básicos

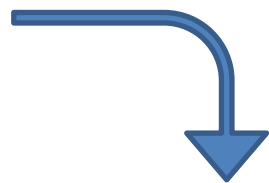


Engano



```
01001100 01101111 01110010
01110000 01110011 01110101
01101100 01101111 01110010
00100000 0110001 01101101
01100011 01101111 01101110
```

Defeito



```
sum(1,2) = 1 10010
               10101
01101100 01101111 01110010
00100000 0110001 01101101
01100011 01101111 01101110
```

Erro

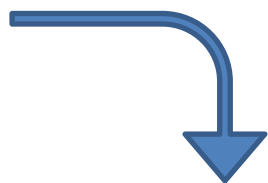


Falha

Depuração de Software: conceitos básicos

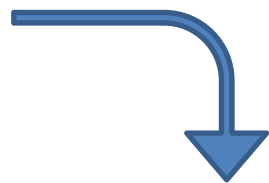


Engano



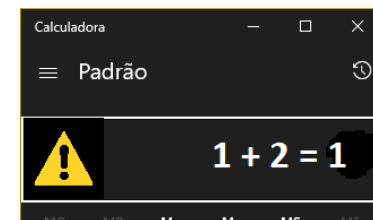
```
01001100 01101111 01110010
01110000 01110011 01110101
01101100 01101111 01110010
00100000 0110001 01101101
01100011 01101111 01101110
```

Defeito



```
sum(1,2) = 1 10010
               10101
01101100 01101111 01110010
00100000 0110001 01101101
01100011 01101111 01101110
```

Erro



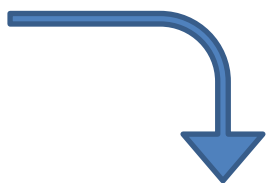
Falha

QUEM É O
“BUG”?

Depuração de Software: conceitos básicos

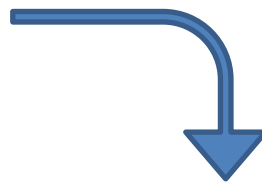


Engano



```
01001100 01101111 01110010
01110000 01110011 01110101
01101100 01101111 01110010
00100000 01100011 01110101
01100011 01101111 01110110
```

Defeito



```
sum(1,2) = 1 10010
               10101
               0110010
00100000 01100011 01101101
01100011 01101111 01101110
```

Erro



Falha

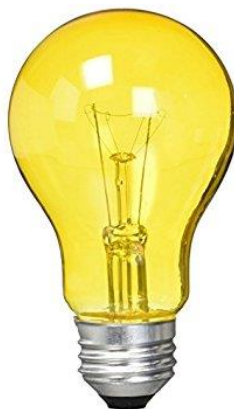
QUEM É O
“BUG”?

Depuração de Software

- Atividade que tem como objetivo corrigir defeitos de software



Localizar



Entender



Consertar

Técnica: Logging

```

1  def my_function(a):
2      for i in range(1, 10):
3          print('value of a inside my_function: '
4              +str(a))
5          a += a
6
7  def another_function(a):
8      for i in range(1, 5):
9          print('value of a inside another_function: '
10             +str(a))
11         a = i

```

(a)

```

value of a inside my_function: 1
value of a inside my_function: 2
value of a inside my_function: 4
value of a inside my_function: 8
value of a inside my_function: 16
value of a inside my_function: 32
value of a inside my_function: 64
value of a inside my_function: 128
value of a inside my_function: 256
value of a inside another_function: 1
value of a inside another_function: 1
value of a inside another_function: 2
value of a inside another_function: 3

```

(b)

Ferramenta: Breakpoint

The screenshot displays the Eclipse IDE interface during a debug session. The main editor shows the source code of `HelloWorld.java` with a breakpoint set at line 56. The `Thread` window on the left shows the execution flow, highlighting the current thread `[qtp1121384694-58]` and its position at line 56. The `Variables` window on the right shows the current state of variables: `this` (HelloWorld (id=83)), `request` (Request (id=84)), `response` (Response (id=88)), and `out` (HttpOutput (id=92)). The `Outline` window on the right shows the class structure, including `com.acme` and `HelloWorld`, with methods `doGet`, `doPost`, and `init` listed. The `Console` window at the bottom shows the output of the program, including the `MonjaDB Log` and the `doGet` method execution.

Debug - org.eclipse.jetty:test-jetty-webapp (9)/src/main/java/com/acme/HelloWorld.java - Eclipse - /Users/jesse/Documents/Eclipse/jetty

Quick Access

Java Debug CVS Repository Exploring MonjaDB

Debug [qtp1121384694-58] (Suspended (breakpoint at line 56 in HelloWorld))

- HelloWorld.doGet(HttpServletRequest, HttpServletResponse) line: 56
- HelloWorld(HttpServlet).service(HttpServletRequest, HttpServletResponse) line: 848
- ServletHolder.handle(Request, ServletRequest, ServletResponse) line: 681
- ServletHandler\$CachedChain.doFilter(ServletRequest, ServletResponse) line: 14
- TestFilter.doFilter(ServletRequest, ServletResponse, FilterChain) line: 114
- ServletHandler\$CachedChain.doFilter(ServletRequest, ServletResponse) line: 14
- QoSFilter.doFilter(ServletRequest, ServletResponse, FilterChain) line: 213
- ServletHandler\$CachedChain.doFilter(ServletRequest, ServletResponse) line: 14

Variables

Name	Value
this	HelloWorld (id=83)
request	Request (id=84)
response	Response (id=88)
out	HttpOutput (id=92)

Outline

- com.acme
 - HelloWorld
 - doGet(HttpServletRequest, HttpServletResponse)
 - doPost(HttpServletRequest, HttpServletResponse)
 - init(ServletConfig) : void

Console

MonjaDB Log

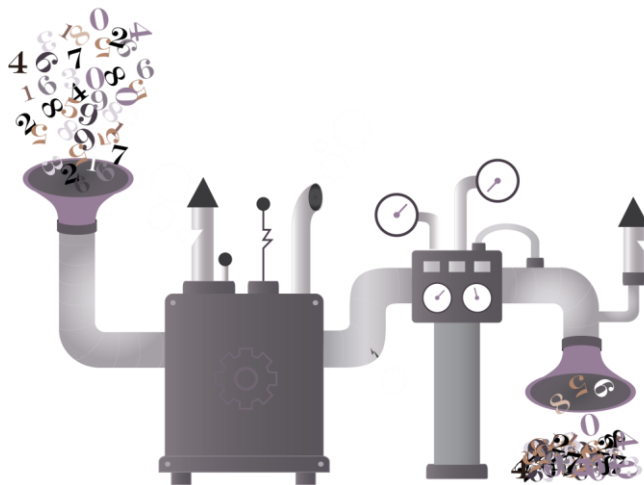
```

Thu Jun 13 10:28:14 CDT 2013:(MActionManager.java:59)::DEBUG:::ab.sessions.find()
Thu Jun 13 10:28:14 CDT 2013:(MActionManager.java:59)::DEBUG:::mj edit 51b9e1cf1116f463396acff9
Thu Jun 13 10:28:14 CDT 2013:(MEditAction.java:45)::DEBUG:::51b9e1cf1116f463396acff9
Thu Jun 13 10:28:14 CDT 2013:(MAuthManager.java:28)::DEBUG:::Thread[Thread-16,6,main]:event_remove_end
Thu Jun 13 10:28:14 CDT 2013:(MAuthManager.java:28)::DEBUG:::Thread[Thread-16,6,main]:event_remove_start
Thu Jun 13 10:28:14 CDT 2013:(MAuthManager.java:28)::DEBUG:::Thread[Thread-16,6,main]:event_remove_end
Thu Jun 13 10:28:14 CDT 2013:(MAuthManager.java:28)::DEBUG:::Thread[Thread-16,6,main]:event_find_start
  
```

Writable Smart Insert 56 : 1

O que existe em comum entre as duas abordagens? (logging e breakpoint)

- O programador precisa escolher variáveis para exibir no log, ou linhas para inserir pontos de interrupção
- Efeito “garbage-in, garbage-out”



“Estado da Prática”

Abordagens amplamente utilizadas

- Logging
- Breakpoint

“Estado da Arte”

Abordagens desenvolvidas em pesquisas

- Depuração reversa
- Depuração onisciente
- Depuração algorítmica
- Depuração estatística
- Outras...

Depuração Reversa

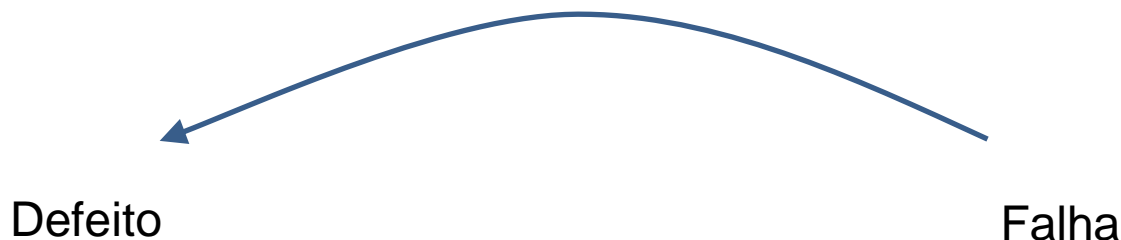


Depuração Reversa



Depuração Reversa

- Permite que o programador inspecione as instruções executadas antes de uma falha

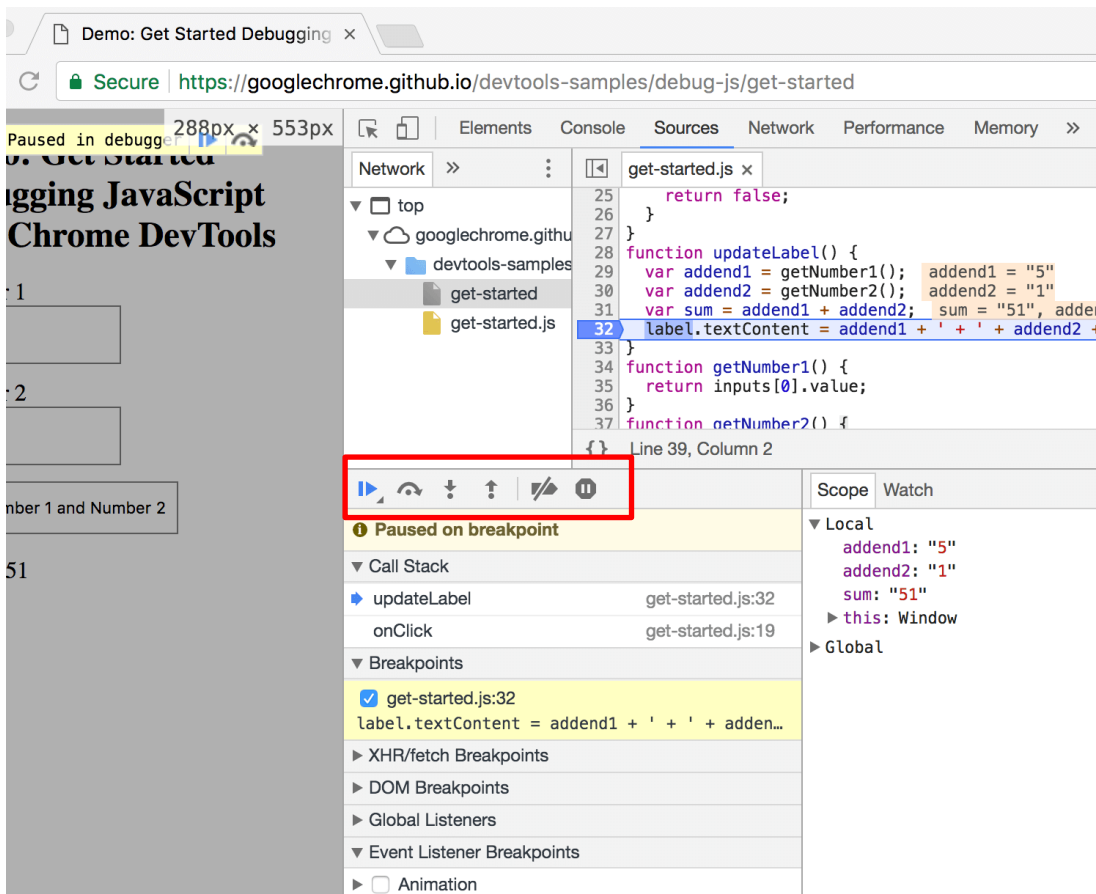


- Navegação da falha para o defeito

*REPT: Reverse Debugging of Failures
in Deployed Software*



Depuração Onisciente

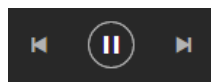


- Pause / resume
- Step over
- Step into
- Step out
- Deactivate breakpoints
- Pause on exceptions

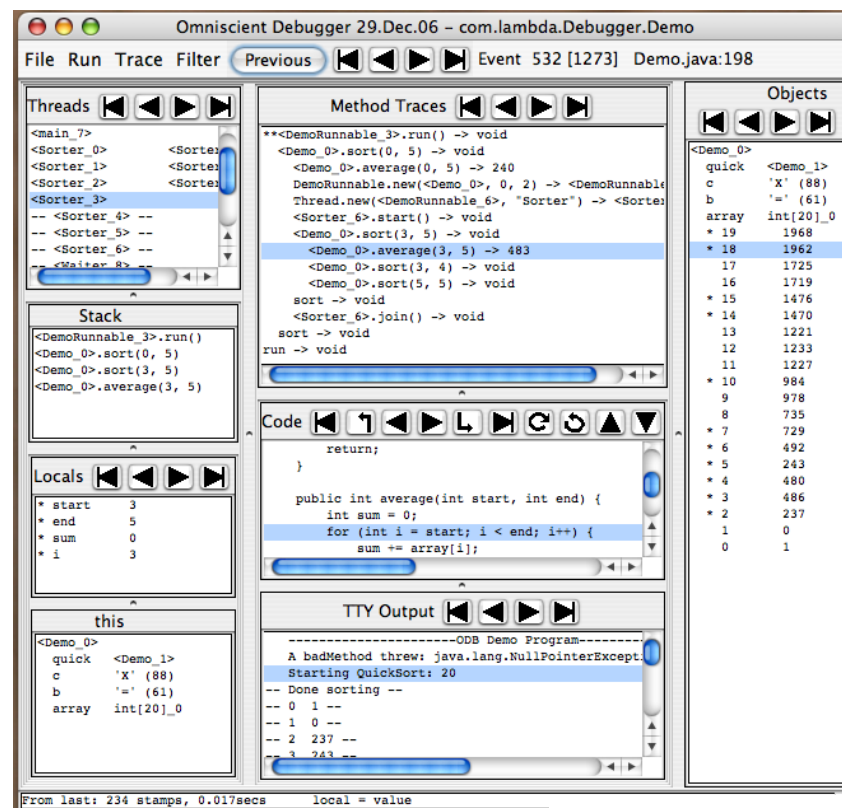
- Não tem a opção de voltar? (step back)

Depuração Onisciente

- É possível alcançar estados passados e futuros
- O depurador precisa armazenar estados anteriores
- Evita a reexecução do programa



Execução



Scalable Omniscient Debugging

Depuração Algorítmica

- Inversão de controle
- Baseada em uma série de perguntas para o programador
- Respostas do programador levam ao método defeituoso

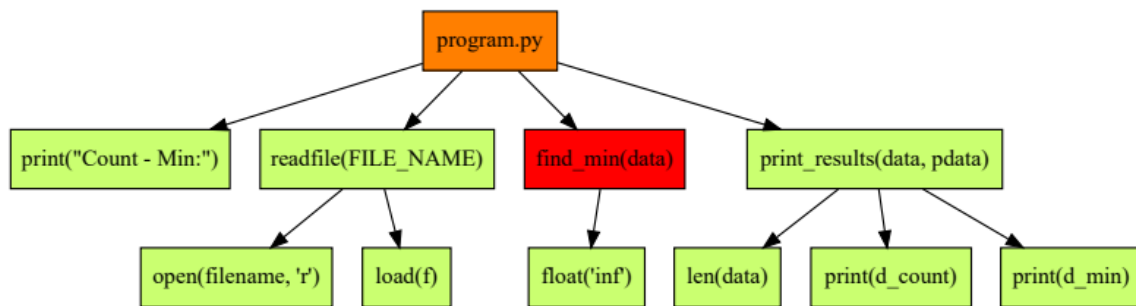
read_file(filename)

Está correto?

SIM

NÃO

Entradas	Saídas
"data.json"	[738, 967, 667, 122]



Provenance-enhanced Algorithmic Debugging

Depuração Estatística

- Análise de múltiplas execuções, algumas corretas e outras incorretas
- Estabelece relações entre linhas de código e execuções incorretas
- Indica ao programador as linhas suspeitas

```
def fatorial(n):
    if n == 0 or n == 1:
        return n
    else:
        return n * fatorial(n - 1)
```

*Visualization of Test Information
to Assist Fault Localization*



Conclusão

- Depuração de software é uma atividade custosa
- Área de pesquisa repleta de desafios
- “Gap” grande entre pesquisa e prática
- Muitas oportunidades para desenvolvimento de novas soluções

Pesquisa em Depuração de Software



<https://tinyurl.com/3xmh6xca>