

programming is terrible

lessons learned from a life wasted

@tef

2

hello, iamtef, I am a bad programmer

I took things apart to find out how they work. Now I am surprised they do.

I am a bad programmer (tests, docs, apology code review, bugs)

we can do better.

Good and Bad Programmers
How Culture Dictates Code
Indoctrination vs Learning
Being Successful vs Being Good

3

Not just complaints, Tips, hope.

mistakes i've made etc. mythos.

people rather than code.

std disclaimer, opinions, not work, or reality




YMMV
HTH
HAND

4

Some find useful, gives me trouble.

I am wrong. Doesn't stop bloggers.

A good place to start



my code is better
than your code

(sing it)

5

false dichotomy of good and bad programmers.

they normally mean this

Programmers who are like me.

Programmers who are not like me.

6

cargo cult their personality -> success

Programmers who use my
favourite language.

Programmers who don't.


7

blub paradox.

aside: unmaintainable lisp, was replaced with perl.

aside: lisp would have prevented 9/11

imply good or bad.



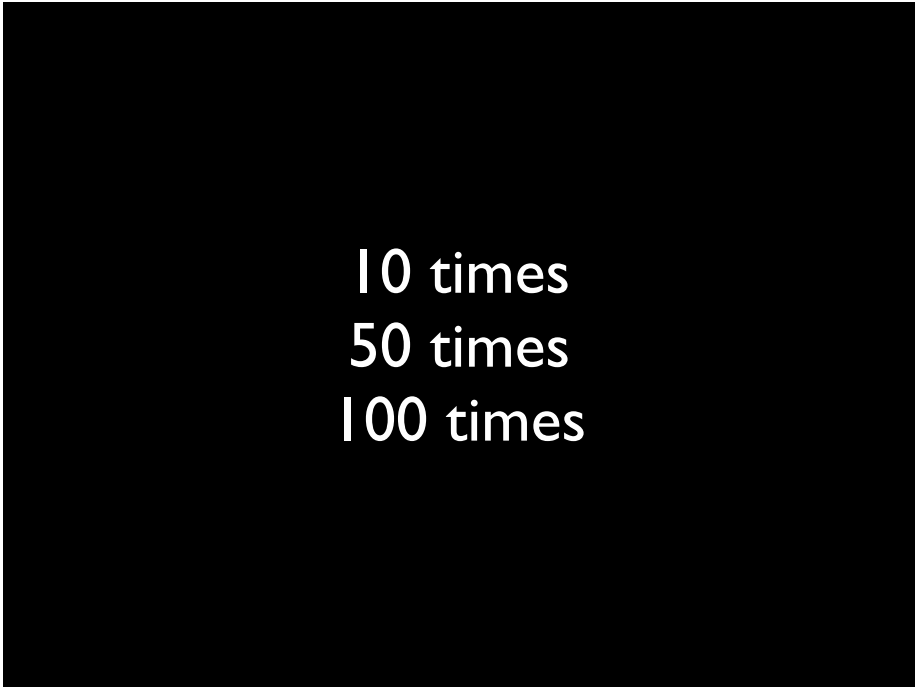
Programmers who share my
political views.

Programmers who don't.

8

mash types into politics for people who understand neither.

why? it is easy. simple answer to a hard question.
also blog hits. especially emotionally charged/trolling



10 times
50 times
100 times

9

the myth of genius. rockstar, ninja, founder, entrepreneur
drowning in puberty/machismo
all bollocks, faulty study, repeated ad-nauseum
nothing to learn cos they're smart or thick
'A type programmer' = 'Easily exploited hard worker'
Worst good/bad



Programmers who are men.

Programmers who aren't.

10

kill yourself now.

Trials on two groups about belief in magic penis abilities.
You are bad person and bad programming
tabs vs spaces joke.

so, if there are two types, this is what I think it looks like

Programmers who know they
make mistakes.

Programmers who think they don't

11

A little of A, a little of B.

Sometimes refuse to try, sometimes refuse to learn.

Biggest mistake is optimism.



“You would think that...”

12

Optimism is still necessary or we would go mad.
Chronically underestimate work.
classic programmer complaint, underling optimism,
could be better, or they could do better.
I am a cynic. We fix bugs but not people.



PEBKAC

13

Mistakes come from environment too.
We need to find out why bug happened
If we want to stop writing so many bugs

Not endemic, systematic.

“...organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations”

14

Code reflects social structures.

You want a service, the team needs to provide/run it, not hand over code.

God objects come from god programmers.

Groups make mistakes too.

The Bike Shed

15

design of nuclear power plant
parkinson's law.

when domain expertise isn't needed, everyone is an expert
everyone has an opinion, debate will never end
putting your thumb on it, etc

The Group Project

16

let's do stuff together, pool out ideas.
ideas no-one wants to actually d on their own
colaboration requires leading by example
doesn't mean ideas, ideas are multiplier
if ideas are all you have

The 'Goon' Project

17

We have a wiki, and a logo. All bikeshedding, no progress.
When you have enthusiasm and not much else.
Let's write a video game.
Let's build a darknet.
Sure, i'll make the wiki and the logo.



Waterfall

18

Not just people or teams, but methodologies.
Waterfall was introduced as a strawman by royce.
People read the first page, eh. that'll do

Project management is often control over measurement or feedback
Milestones handed out, we'll just have to do it perfectly.



Manhole Cover

19

Also how we get workers too. We can't find good programmers. We can't interview, or measure it. We end up doing brainteasers. What should you do when you get a brainteaser?



LEAVE

20

Wasson Selection Task

Not a quiz show host. See how you cope with stupid questions

It's good because our management are terrible.

But FACEBOOK, GOOGLE, MICROSOFT

Any company that does this is universally bad.

Occultism

21

Not science, Art, but rituals, cargo culting. Best practices are superstition.
Say we're scientists, but don't test code, methodologies, or assumptions.
At best, a craft.

We learn more from maintenance, fix, test, adapt, evolving. Than creating.
Not to deny experiments, 'top down the second time' still true.
Prototypes help you explore an idea, maintenance brings understanding.
Not just bad code, bad practices, but bad teaching

Teaching

22

Nostalgia and learning preferences. Do what I did and learn what I did. “What must they know” rather than “What do they want to learn and how?”

Learning Preferences matter. Need to encourage people to learn and explore on own
, rather than dictating course. Guidance and support is good. Adult vs child education.

If asked what to learn, I ask what they want to create.



my first language

23

they can do something fun with in an afternoon,
friends know and support, easy to install, doesn't
require learning ides. scripting languages.

don't worry about OO, or advanced things. get
something simple running and play.
learning programming should be a step of something fun

Learning Through Play

24

Find a sandbox, turtle graphics, music. Something where you get instant feedback, and easy tweaking. Get them to explain things to me, and questioning than rote/memorization (PUB STAT VOID MAIN). Computer is a tool for exploring ideas, not a beauntracraft with form filling.

Guilty: I am nostalgic too. For Logo



Seymour Papert Mindstorms

25

LOGO, Math World, Users build rules, no rules but their own.
Lets them see consequences, understand things (verb stuff)

Similar idea in math. death march through formulae, not actual
problem
solving. Learning is fun when you get to be creative.



view-source

26

other influence, get to play, get to change, get to tweak
programming is not just a way to do business rules,
but a way to understand things.

some bloggers (terrible atwood) tells people off for playing and
learning
as they did, and they are bad people.



REAL MEN USE C

27

C is useful, uses and popular. It's also hard, demanding. Not a good first lang.

Hard to do simple things quickly. C is character building
Shell before C.



USED IN INDUSTRY

28

C#/Java always simpler script instead atop runtime.
OO is hard to grok without learning why it is useful.
Hard to begin without forced ignorance.

Better second, not first.

Aside: Learning with caution, looking for excuse to avoid,
like on forums, want to avoid effort.

MATHS IS HARD

29

Maths/programming are related. Also floating point misconceptions. Need enough maths as your problem demands. Not many programs need things beyond counting. Spreadsheet knowhow good enough. Programming is part maths, not diff-geom, but same discipline of thought, reasoning, Programmers don't need to be maths, but are maths. Curry Howard, innit. Ultimate interdisciplinary, writing, critical reason, engineer discipline, maths reasoning. Overlooked, domain experience of problem.

How to be successful

30

I'm not good enough. I go mad. Other people seem to survive, and I will share strategies. Document badly, ignoring failure cases, types or args. too busy to share info, be writing code. Lots of code. With autocomplete. Wrappers, avoid stdlib. Reinvent, go crazy with adv. feat. Modules, abstracton everywhere, esp if it only makes sense when combined. Fix bugs by making new ones. Close bugs that get reopened byt someone else. Shifting problem. Non-det tests, hard setup. Be the central point of fail, or be replaceable. Create work for yourself, be a soliphist, sabotage is rewarded.

How to be good

31

Read code, learn from others. Professional writer. Ten years experience. All courses are about making new software not fixing existing things. no focus on maintenance, debugging or analysis.

Estm what hasn't been done before may be due to ignorance. Write cose as if it is wrong and you will have to delete it. Probably will. Fail quickly, get is wrong. Easier when you don't write too much code. Empathy good. Don't be a diva, an artist, don't try to be right, but make it less painful to be wrong. Easy to replace not easy to extend.



A final warning

32

the software industry is terrible, so is every other industry. retraining won't help you escape people.