Course requirements. Software problems.

Saulius Gražulis

Vilnius, 2020

Vilnius University, Faculty of Mathematic and Informatics
Institute of Informatics

This set of slides may be copied and used as specified in the Attribution-ShareAlike 4.0 International license

Hand in assignments

- The VMA (Moodle) system:
    - https://emokymai.vu.lt/?lang=en
    - course “Fundamentals of Programming Methodology“
    - A short 4 question 10 min. quiz before each lecture – from the material of the previous lectures;
        - Quizes give you 1 point in the final mark.

Grading

- Intermediate test/quiz (in the mid-term, fro half of the course material);
  - 1.5 points
- Egzam test/quiz
  - 1.5 points
- Assignments
  - 5 points for assignments
  - formally, at one assignments must be handed in to pass the exam.
  - 4 mandatory assignments yield 5 points;
  - a *subtractive* grading system will be used:
    - an ideally perfomed assignment is worth 100%;
    - each error will attract negative points, depending on the importance of the error, with explanation;
    - If you score too few points for practical assignemnts, additional assignments can be offered as extra credits.
- Final presentation
  - 1 point

Rules of video conferencing

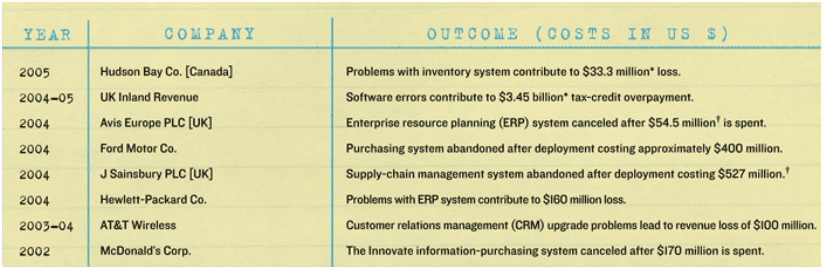
- Only the speaker has the microphone ON; all others must switch their microphones **OFF**;
- The speaker **should** have camera on (VU rules!);
- When a teacher asks you a question, you **must** raise your hand and, when assigned a talk slot, switch on you microphone and answer; or type your answer in the chat, as requested (VU rules!);
- To ask a question, use the “raise hand” functionality and/or post your question to the forum;
- Recordings of lectures are copyrighted; it is **not permitted** to post them outside the university without consent of the lecturer and university officials;
- If the connection is interrupted, **do not leave** the lecture – the teacher will set up alternative connection in several minutes;

So what is the problem?

To put it quite bluntly: as long as there were no machines, programming was no problem at all; when we had a few weak computers, programming became a mild problem, and now we have gigantic computers, programming had become an equally gigantic problem. In this sense the electronic industry has not solved a single problem, it has only created them, it has created the problem of using its products.

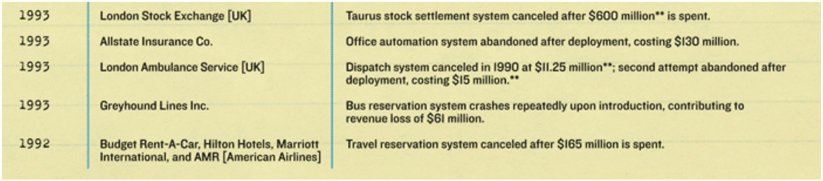
Edsger W. Dijkstra
ACM Turing Lecture 1972
EWD340 The Humble Programmer

Hall of shame



YEAR	COMPANY	OUTCOME (COSTS IN US $)
2005	Hudson Bay Co. [Canada]	Problems with inventory system contribute to $33.3 million* loss.
2004–05	UK Inland Revenue	Software errors contribute to $3.45 billion* tax-credit overpayment.
2004	Avis Europe PLC [UK]	Enterprise resource planning (ERP) system canceled after $54.5 million† is spent.
2004	Ford Motor Co.	Purchasing system abandoned after deployment costing approximately $400 million.
2004	J Sainsbury PLC [UK]	Supply-chain management system abandoned after deployment costing $527 million.†
2004	Hewlett-Packard Co.	Problems with ERP system contribute to $160 million loss.
2003–04	AT&T Wireless	Customer relations management (CRM) upgrade problems lead to revenue loss of $100 million.
2002	McDonald's Corp.	The Innovate information-purchasing system canceled after $170 million is spent.

…



1993	London Stock Exchange [UK]	Taurus stock settlement system canceled after $600 million** is spent.
1993	Allstate Insurance Co.	Office automation system abandoned after deployment, costing $130 million.
1993	London Ambulance Service [UK]	Dispatch system canceled in 1990 at $11.25 million**; second attempt abandoned after deployment, costing $15 million.**
1993	Greyhound Lines Inc.	Bus reservation system crashes repeatedly upon introduction, contributing to revenue loss of $61 million.
1992	Budget Rent-A-Car, Hilton Hotels, Marriott International, and AMR [American Airlines]	Travel reservation system canceled after $165 million is spent.

https://spectrum.ieee.org/computing/software/why-software-fails
[first accessed: 2012-09-09, last accessed: 2020-08-24]

Therac-25 disaster...

Some of the most widely cited **software-related accidents** in safety-critical systems involved a computerized radiation therapy machine called the Therac-25. Between June 1985 and January 1987, six known accidents involved massive overdoses by the Therac-25 -- **with resultant deaths and serious injuries**.

Software engineering. The Therac-25 accidents were fairly unique in having software coding errors involved – most computer-related accidents have not involved coding errors but rather errors in the software requirements such as omissions and mishandled environmental conditions and system states. Although **using good basic software-engineering practices** will not prevent all software errors, it **is certainly required as a minimum**.

(Leveson et al. 1993);
http://www.cse.msu.edu/~cse470/Public/Handouts/Therac/Therac_5.html
[last accessed: 2020-09-13]

Approaches for reliable software

- Didelė dalis problemų kyla dėl nesusikalbėjimo, todėl:
  - Programuokime tvarkingai (**kodavimo stilius**!)
  - **Skaitykime** programas, rašykime **aiškiai**
  - **Dokumentuokime** savo programas (komentarai, **versijų kontrolė**)
- Rašykime **mažas** programas (Unix!)
- Patikrinkime, kaip mūsų programos veikia (**testai**)
- **Įrodykime**, kad mūsų programos teisingos

Do we know Perl?

- Ką daro/kaip veikia ši programa? :D

```
$s=2;

$d=500;
$w="A";$_='ZIsHPX=$s-Z*Z;$|C;J"sH=\nZ.";O!XNJ"0"x$d,"\n";exit}QZNpush
(F,Z%10PZIZD)}QXNpush(@W,X%10PXIXD)}subT{GMw>MW)OMw!=MWPZ=Mw;QE1NGZV>B)
OZV!=BPZK}1}subY{my(FPZ=0;X=Mw+1;QX>ZNXV+=ZV*S;X[E1]IXVDPXV%C0;E+}MYKO!X
[MY]PF}Q$dKNLF;S=2;@T=Y;@W=(0,0,@WPSC;QSNAOTNF=(KS,FPlast}S++}AZ[0]K;Z=0;S
=MW+1;QZ-SNB+=9-ZV;OB>C0NB-C0;Z[E1]K}E+}Q!U[MW]NMWK};JX[0]}J"\n";
';foreach$s(qw/ L(S,@TPLY; UV =1*.1 Z+ @Y return( qrt($s) =R(
prR -- @w= $# )
{ if( ); Te( int Ul Wl Xi [Z] Yi Zh wh $w
/){s;$w;$s;g;$w++}eval;
```

http://www.foo.be/docs/tpj/issues/vol2_3/tpj0203-0012.html

Coding style

- Programs must be written and formatted accurately;
- A uniform coding style myst be used

http://saulius-grazulis.lt/ saulius/paskaitos/VU/programavimo-metodologijos-pagrindai/reikalavimai/kodavimo-stilius/

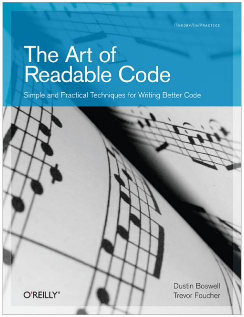
Code formatting

```
#! /usr/bin/perl

use strict;
use warnings;

my $selected_line;

while( <> ) {
    $selectd_line = $_ if rand() < 1/$.;
}

print $selected_line;
```

Commenting code

```
#! /usr/bin/perl
# Ši programa pasirenka atsitiktinę eilutę iš savo įvesties;
# kiekviena eilutė pasirenkama su vienoda tikimybe.

use strict;
use warnings;
use English;

my $selected_line;

while( my $current_line = <> ) {
    $selected_line = $current_line
        if rand() < 1/$INPUT_LINE_NUMBER;
}

if( defined $selected_line ) {
    print $selected_line;
} else {
    warn "No text lines in input data files"
}

# Įrodymas:
# rand() grąžina atsitiktinį skaičių intervale [0..1)
# Tebūnie n – eilutės numeris ($INPUT_LINE_NUMBER)
# Indukcijos bazė: kai n = 1, eilutė pasirenkama, nes rand() < 1
# Indukcijos žingsnis: kai n = N, eilutė pasirenkama su tikimybe 1/N;
#     - kitos eilutės lieka su tikimybe (N-1)/N;
#     - bet jos buvo pasirinktos su tikimybe 1/(N-1) (indukcijos prielaida),
#     - tad kiekvienos tikimybė yra ((N-1)/N) * (1/(N-1)) = 1/N Q.E.D.
```

Recommended additional reading



"The Art of Readable Code: Simple and Practical Techniques for Writing Better Code"

By Dustin Boswell, Trevor Foucher
Publisher: O'Reilly Media
Released: November 2011
Pages: 206

http://shop.oreilly.com/product/9780596802301.do?sortby=publicationDate
http://www.amazon.com/The-Readable-Code-Dustin-Boswell/dp/0596802293

References

Leveson, N. G. et al. (July 1993). “An investigation of the Therac-25 accidents”. In: *Computer* 26.7, pp. 18–41. DOI: 10.1109/mc.1993.274940.