ITI 1121. Introduction to Computing II

Course **requirements**

by

Marcel Turcotte

Version January 6, 2020

Preamble

Preamble

Learning objectives

Learning objectives

- **Inform** about the course requirements.
- **Explain** the general learning objectives in this course.
- **Know** the university regulations on academic fraud.

Readings:

- This document, as well as these:
  - Plan de cours
  - Academic Integrity

Preamble

Plan

Plan

1. Preamble
2. Syllabus
3. Academic integrity
4. Content
5. Epilogue
6. Prologue

Marcel Turcotte : short bio!

- **1965-** Born in **Montréal**
- **1995** Ph. D. in Computer Science, Université de **Montréal**
- **1995–97** University of **Florida** (U.S.), Chemistry Department
- **1997–00** Imperial Cancer Research Fund, **London**, England
- **2000-** University of **Ottawa**
  - **2006–08** **Coordinator** for the **coop** program (computer science)
  - **2010–11** **Coordinator** master in **bioinformatics**
  - **2012–18** **Vice-dean** of undergraduate studies

Source: blog.f1000.com

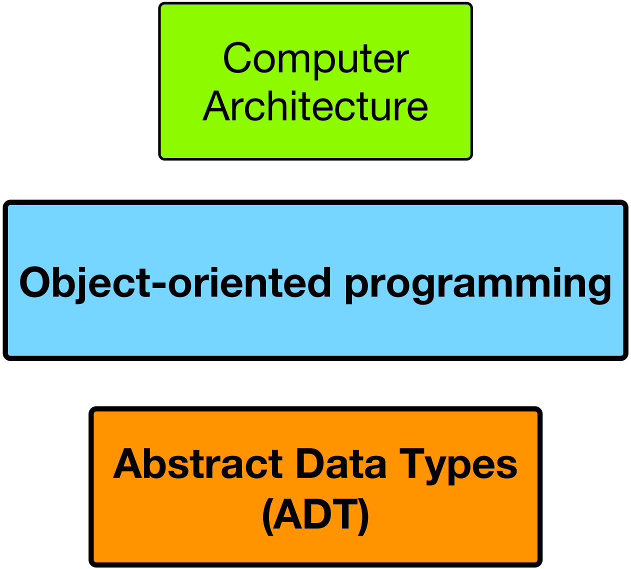
Marcel Turcotte : short bio!

1. RNA and DNA, regulation of gene expression, secondary structure, pattern inference, text algorithmic
2. **Software**: MC-Sym (co-author), eXtended-Dynalign, Profile-Dynalign, Seed, ACSEA, ModuleInducer, RiboFSM, MotifGP, Seed$_w$
3. Designing algorithms and data structures, applied artificial intelligence and machine learning

Source: QIAGEN/CLC bio

Syllabus

Course content

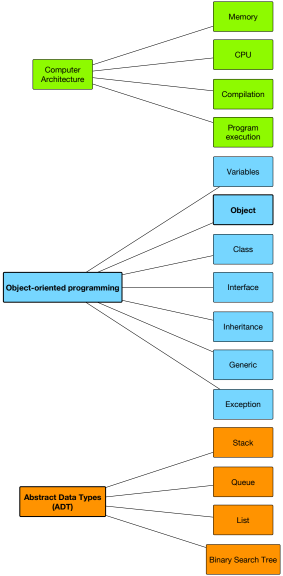


Computer Architecture

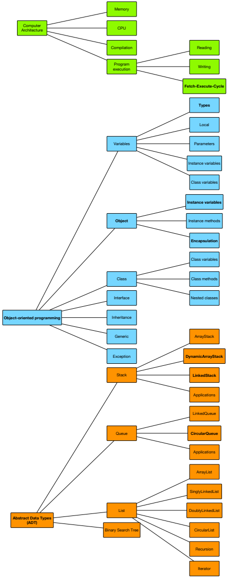
Object-oriented programming

Abstract Data Types (ADT)

Course content

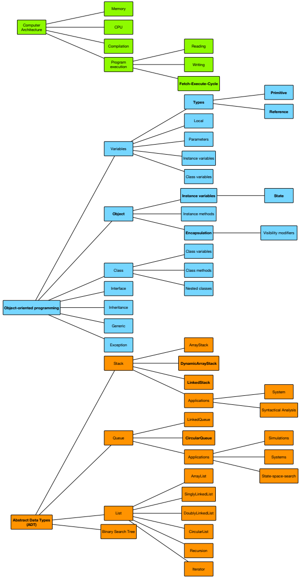


Course content



- Computer Architecture
  - Memory
  - CPU
  - Compilation
  - Program execution
    - Reading
    - Writing
    - **Fetch-Execute-Cycle**
- **Object-oriented programming**
  - Variables
    - **Types**
    - Local
    - Parameters
    - Instance variables
    - Class variables
  - **Object**
    - **Instance variables**
    - Instance methods
    - **Encapsulation**
  - Class
    - Class variables
    - Class methods
    - Nested classes
  - Interface
  - Inheritance
  - Generic
  - Exception
- **Abstract Data Types (ADT)**
  - Stack
    - ArrayStack
    - **DynamicArrayStack**
    - **LinkedStack**
    - Applications
  - Queue
    - LinkedQueue
    - **CircularQueue**
    - Applications
  - List
    - ArrayList
    - SinglyLinkedList
    - DoublyLinkedList
    - CircularList
    - Recursion
    - Iterator
  - Binary Search Tree

Course content



Take note!

- “Professors must supply a course syllabus during the first meeting with the students at the beginning of each course.” (Academic Regulation 8.5)
- The course syllabus will remain unchanged for the rest of the semester!

Evaluations

- Examen **midterm**: 25 %
- Examen **final**: 40 %
- **Assignments** (4): 25 %
- **Laboratories** (10–12): 10 %

You must get **at least 50% on the exams** to pass the course. Failure on the exams will result in failure on the course!

Examinations

Midterm

- Sunday **March 1, 2020**, **10:00 - 12:00**
- Closed books

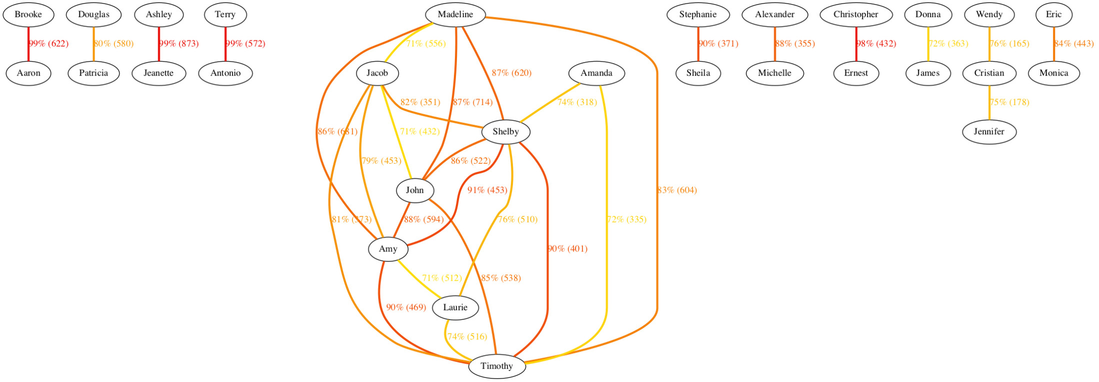
Final

- Closed books

Academic integrity

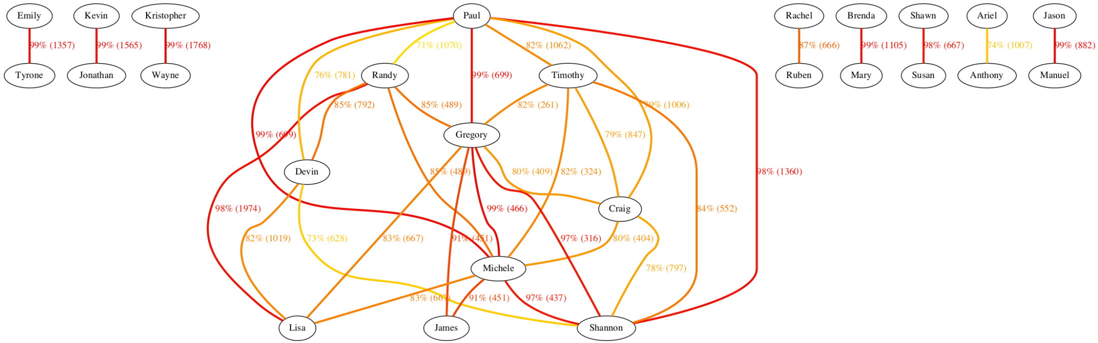
This applies to everybody!

- Last year, over **40 students** were alleged to have committed an academic fraud for their first assignment. . .



This applies to everybody!

- . . . and second assignment!
  - Each link shows the relationship between a **pair of submissions**.
    - The labels indicate the **degree of similarity** and the **number of lines of code** (in parentheses).



Academic integrity

Academic fraud: Any act by a student that **may result in a distorted academic evaluation** for that student or another student.

- **plagiarising** or **cheating** in any way;
- submitting work **not partially or fully the student's own**;
- submitting the **same work or a large part of the same piece of work in more than one course**;
- **falsifying** or **misrepresenting** an academic evaluation, using a **forged** or **altered** supporting document or facilitating the use of such a document;
- taking any action **aimed at falsifying an academic evaluation**.

Regulation:

- `https://www.uottawa.ca/administration-and-governance/academic-regulation-14-other-important-information`

Good practices

- Never publish your source code on **GitHub** publicly before the deadline for the submission of the assignment.
- You can publish your solution **after the deadline**.
  - However, you should withdraw your University of Ottawa **student number**, because someone else could misuse this information.
  - You also need to be careful, in some cases you may be publishing **source code that isn't yours**.
    - In particular, you could release source code that was **developed by the instructors**.

You’re the victim

- The **first victim** in copying someone else’s solution **is you**.
- You won’t learn “**how to learn**.”
- **Maybe** you understand the solution you copied.
    - However, you won’t have been able to learn **how to create those solutions in the first place**.
    - Of course, that will impact your ability to pass the **exams**.
    - But it will also impact your ability to find your **dream job**.
    - **Interviews** for software engineering and computer science positions are very thorough.
    - You’ll be asked to **solve problems** that may in fact be similar to the ones we have in our assignments.

Be extremely careful, in some cases, **the students didn't know their solution had been copied**, which makes this message important for everyone.

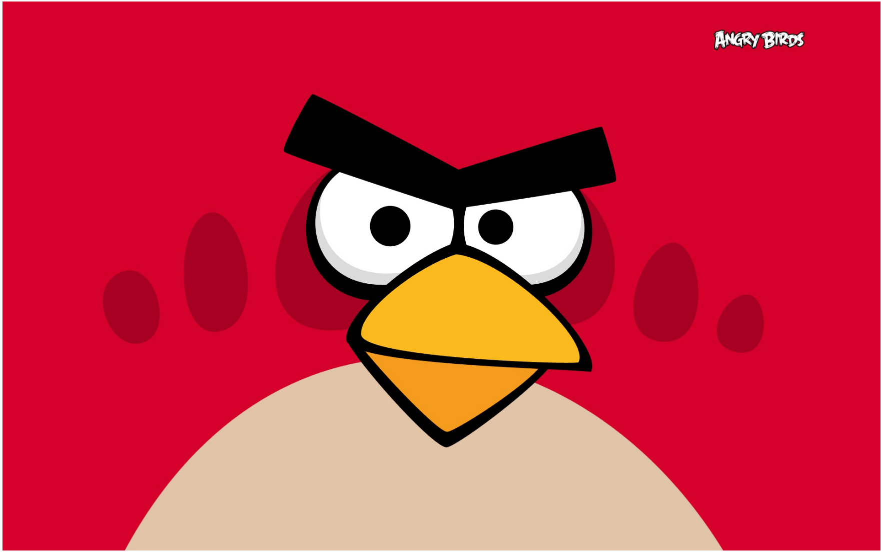
Academic fraud: scenarios

Raise your hand if you intend to cheat in this class!

- Someone's asking for access to your assignment.
  - This person is a friend.
  - It's too late. This poor soul won't finish in time. . .
- You're stuck.
  - You choose to collaborate with one or more teams. . .
- The files are forwarded to the representative of the Dean: the student gets **0**, for **this assignment**, for **all assignments**, for **the course**, to take an **ethics course**, **expulsion** from the university, etc.

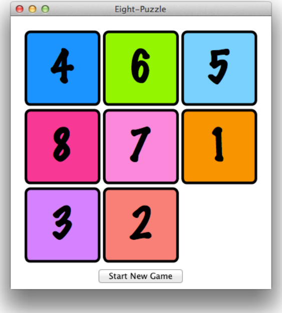
Analogy with high-level sports

Policy on the use of electronic devices

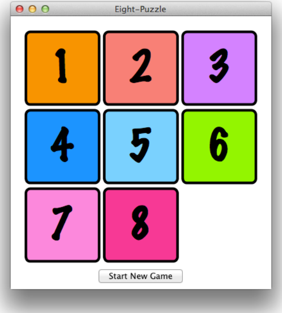


Content

Assignments

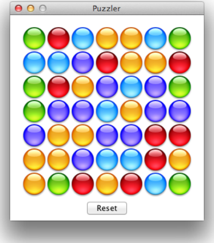


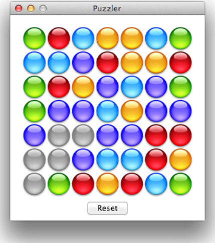
4	6	5
8	7	1
3	2	

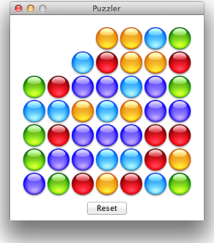


1	2	3
4	5	6
7	8	

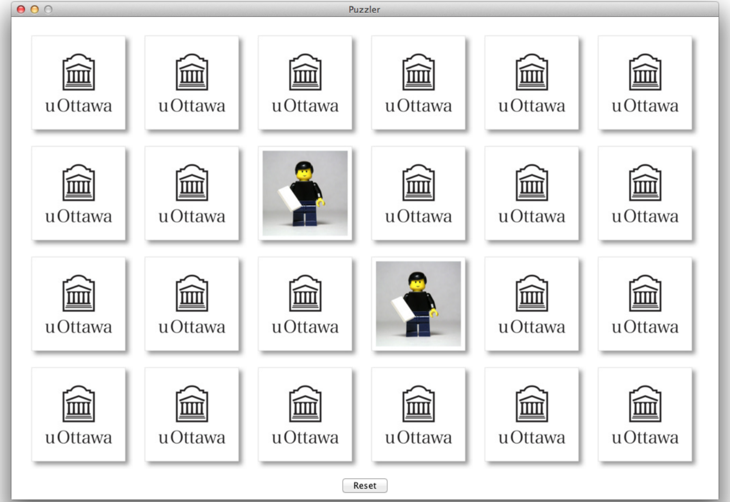
Assignments



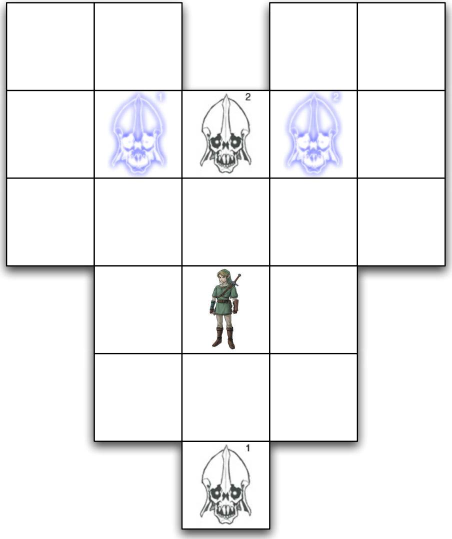




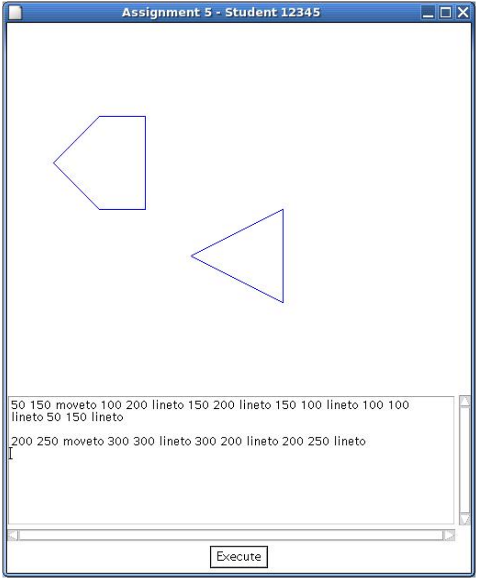
Assignments



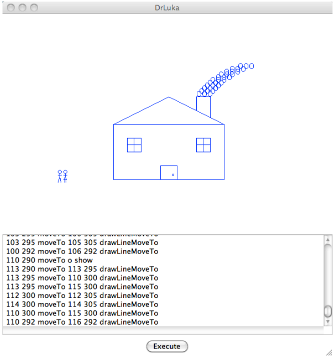
Assignments



Assignments

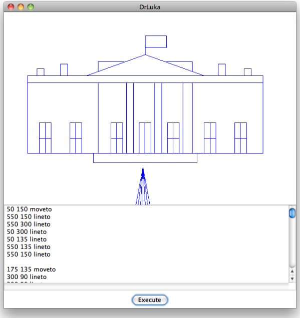


```
50 150 moveto 100 200 lineto 150 200 lineto 150 100 lineto 100 100
lineto 50 150 lineto

200 250 moveto 300 300 lineto 300 200 lineto 200 250 lineto
```



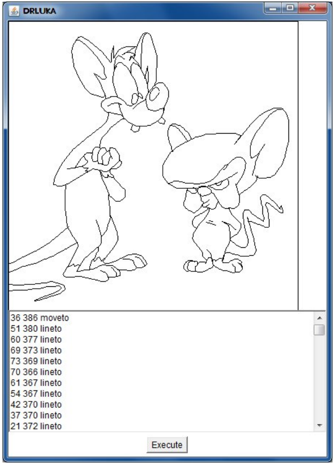
```
103 295 moveTo 105 305 drawLineMoveTo
100 292 moveTo 106 292 drawLineMoveTo
110 290 moveTo o show
113 290 moveTo 113 295 drawLineMoveTo
113 295 moveTo 110 300 drawLineMoveTo
113 295 moveTo 115 300 drawLineMoveTo
112 300 moveTo 112 305 drawLineMoveTo
114 300 moveTo 114 305 drawLineMoveTo
110 300 moveTo 115 300 drawLineMoveTo
110 292 moveTo 116 292 drawLineMoveTo
```

Samuel Bostock 2010



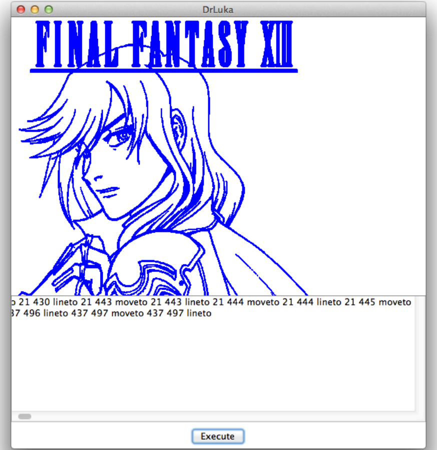
```
50 150 moveto
550 150 lineto
550 300 lineto
50 300 lineto
50 135 lineto
550 135 lineto
550 150 lineto

175 135 moveto
300 90 lineto
```

Liam Shea Williams 2010



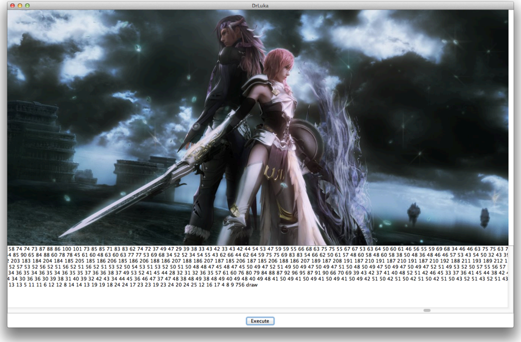
```
36 386 moveto
51 380 lineto
60 377 lineto
69 373 lineto
73 369 lineto
70 366 lineto
61 367 lineto
54 367 lineto
42 370 lineto
37 370 lineto
21 372 lineto
```

Quentin Smith 2011



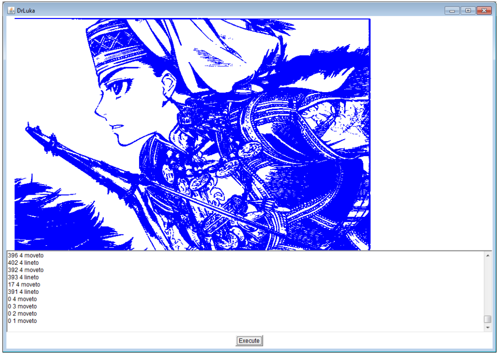
```
o 21 430 lineto 21 443 moveto 21 443 lineto 21 444 moveto 21 444 lineto 21 445 moveto
37 496 lineto 437 497 moveto 437 497 lineto
```

Olivier Gagnon 2012



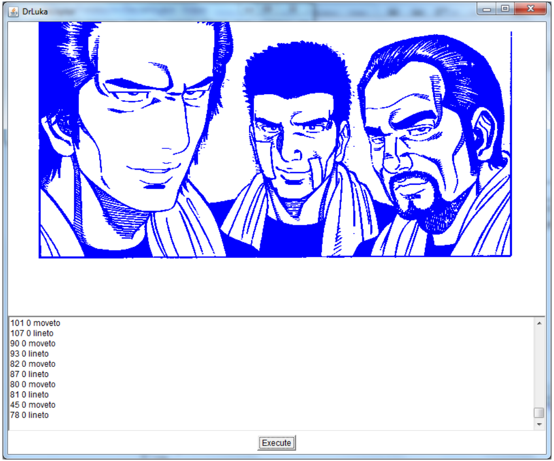
```
58 74 74 73 87 88 86 100 101 73 85 85 71 83 83 62 74 72 37 49 47 29 39 38 33 43 42 33 43 42 44 54 53 47 59 59 55 66 68 63 75 75 55 67 67 53 63 64 50 60 61 46 56 55 59 69 68 34 46 46 63 75 75 63 7
4 85 90 65 84 88 60 78 78 45 61 60 48 63 60 63 77 77 53 69 68 34 52 52 34 54 55 43 62 66 44 62 64 59 75 75 69 83 83 54 66 62 50 61 57 48 60 58 48 60 58 38 50 48 36 48 46 46 57 53 43 54 50 32 43 3
2 203 183 184 204 184 185 205 185 186 206 185 186 206 188 186 207 188 186 207 187 185 206 187 185 206 188 186 207 189 187 208 191 187 210 191 187 210 191 187 210 192 188 211 193 189 212 1
52 57 53 52 56 52 51 56 52 51 56 52 51 53 52 50 54 53 51 53 52 50 51 50 48 48 47 45 48 47 45 50 49 47 52 51 49 50 49 47 50 49 47 51 50 48 50 49 47 50 49 47 50 49 47 52 51 49 53 52 50 57 55 56 57
34 36 35 34 36 35 34 36 35 35 37 36 36 38 37 49 53 52 41 45 44 28 32 31 32 36 35 57 61 60 76 80 79 84 88 87 92 96 95 87 91 90 66 70 69 39 43 42 37 41 40 48 52 51 42 46 45 33 37 36 41 45 44 38 42 4
4 34 30 36 36 30 39 38 31 40 39 32 42 43 34 44 45 36 46 47 37 47 48 38 48 49 38 48 49 40 49 48 40 49 48 41 50 49 41 50 49 41 50 49 41 50 49 42 51 50 42 51 50 42 51 50 42 51 50 43 52 51 43 52 51 43
13 13 5 11 11 6 12 12 8 14 14 13 19 19 18 24 24 17 23 23 19 23 24 20 24 25 12 16 17 4 8 9 756 draw
```

Olivier Gagnon 2012 (LVM Modifiée)



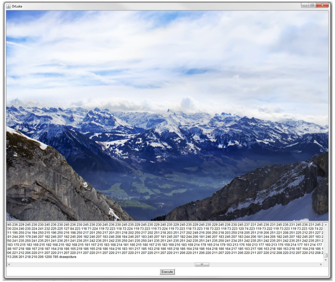
```
396 4 moveto
402 4 lineto
392 4 moveto
393 4 lineto
17 4 moveto
391 4 lineto
0 4 moveto
0 3 moveto
0 2 moveto
0 1 moveto
```

Liban Osman 2012



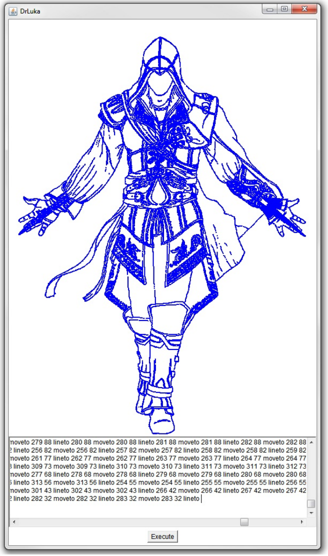
```
101 0 moveto
107 0 lineto
90 0 moveto
93 0 lineto
82 0 moveto
87 0 lineto
80 0 moveto
81 0 lineto
45 0 moveto
78 0 lineto
```

Liban Osman 2012



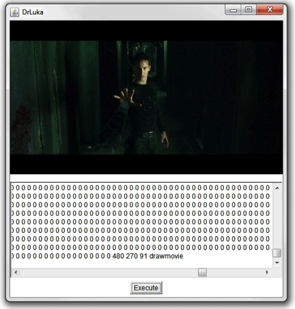
```
45 236 229 245 236 230 245 236 230 245 236 230 245 236 230 245 236 230 245 235 229 245 235 229 245 235 229 245 235 229 245 235 229 245 236 230 245 237 231 245 236 231 245 236 231 245 236 231 245 2
30 224 240 230 224 241 232 225 225 127 84 223 118 71 224 119 72 223 118 72 224 119 73 223 119 73 224 119 73 223 118 73 223 118 73 223 119 73 223 120 74 223 119 73 222 119 73 223 119 73 223 120 74 22
11 190 250 214 194 250 215 196 250 216 198 250 217 201 250 217 201 251 218 202 250 217 202 251 218 203 251 217 202 249 216 200 250 218 203 250 218 205 251 219 206 251 221 208 251 225 212 251 227 2
81 244 205 179 246 207 182 246 207 182 245 206 182 246 207 183 246 208 184 246 207 183 245 207 181 245 207 182 244 206 181 246 207 183 245 207 183 244 206 181 244 206 182 245 207 182 245 207 183 2
50 241 235 250 241 235 251 241 236 251 241 236 251 242 236 251 242 235 250 241 235 250 241 236 251 241 235 251 242 235 251 241 235 250 241 234 251 242 236 251 242 235 251 242 235 251 242 236 251 2
183 170 215 183 169 215 182 168 215 182 168 215 181 167 215 183 168 214 181 166 215 180 167 215 183 169 216 183 169 214 179 165 214 179 163 213 176 160 213 177 160 213 176 159 214 177 161 214 177
88 167 218 188 167 218 187 164 218 186 165 219 186 165 218 186 164 216 183 161 217 185 163 218 186 165 218 186 164 218 186 164 218 186 164 217 185 163 217 186 163 218 186 163 218 187 164 218 186 1
06 221 212 206 220 211 207 220 211 207 220 211 207 220 211 207 220 211 207 220 211 206 220 211 206 220 211 207 220 211 206 220 211 207 220 212 208 220 212 207 220 212 208 2
13 206 201 218 210 206 1200 785 drawpicture
```

Matthew Horton 2012



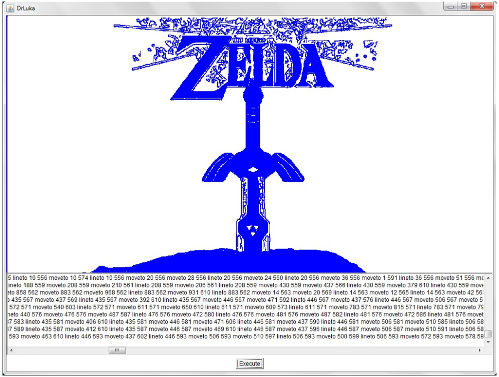
```
noveto 279 88 lineto 280 88 moveto 280 88 lineto 281 88 moveto 281 88 lineto 282 88 moveto 282 88
2 lineto 256 82 moveto 256 82 lineto 257 82 moveto 257 82 lineto 258 82 moveto 258 82 lineto 259 82
noveto 261 77 lineto 262 77 moveto 262 77 lineto 263 77 moveto 263 77 lineto 264 77 moveto 264 77
3 lineto 309 73 moveto 309 73 lineto 310 73 moveto 310 73 lineto 311 73 moveto 311 73 lineto 312 73
noveto 277 68 lineto 278 68 moveto 278 68 lineto 279 68 moveto 279 68 lineto 280 68 moveto 280 68
5 lineto 313 56 moveto 313 56 lineto 254 55 moveto 254 55 lineto 255 55 moveto 255 55 lineto 256 55
noveto 301 43 lineto 302 43 moveto 302 43 lineto 266 42 moveto 266 42 lineto 267 42 moveto 267 42
2 lineto 282 32 moveto 282 32 lineto 283 32 moveto 283 32 lineto
```

Matthew Horton 2012



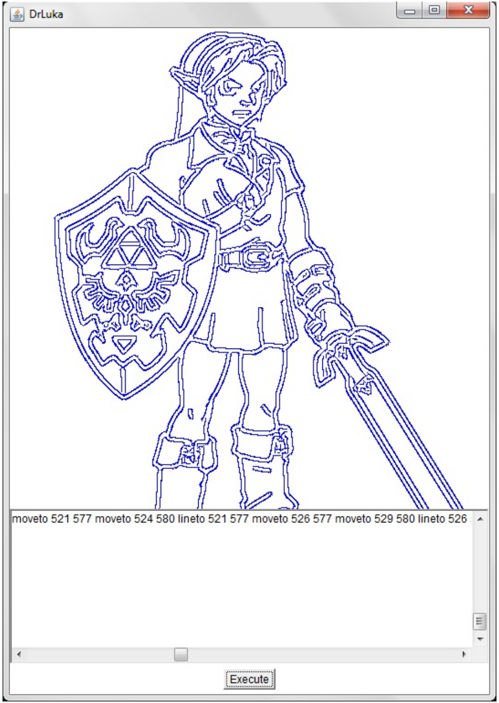
```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 480 270 91 drawmovie
```

Matthew Horton 2012

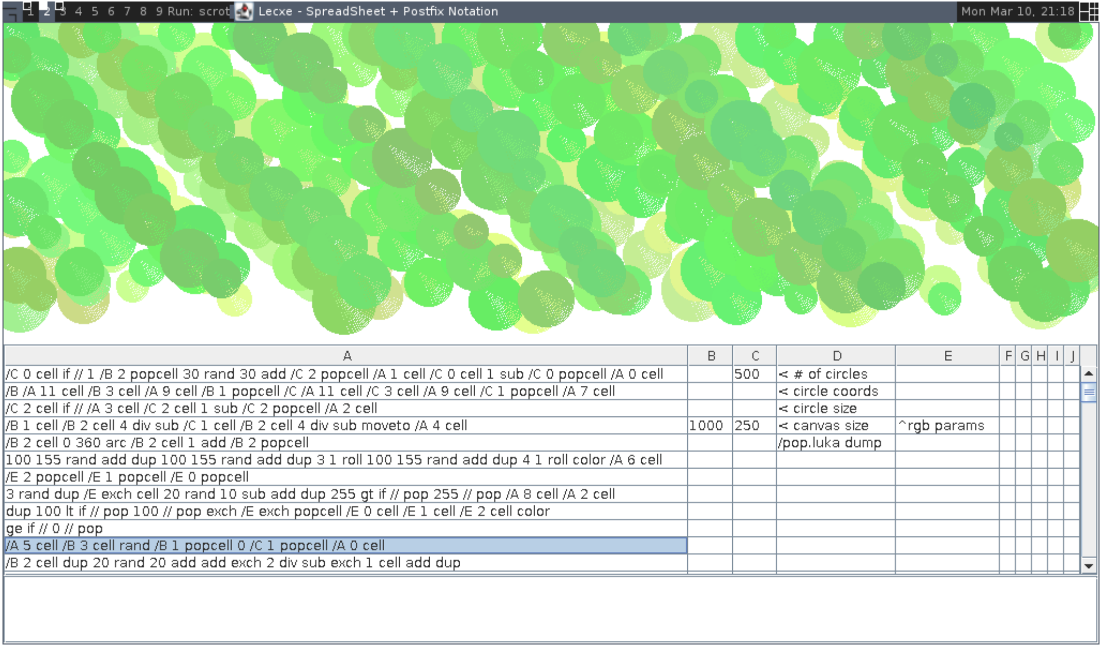


```
5 lineto 10 556 moveto 10 574 lineto 10 556 moveto 20 556 moveto 28 556 lineto 20 556 moveto 24 560 lineto 20 556 moveto 36 556 moveto 1 591 lineto 36 556 moveto 51 556 mo
ineto 188 559 moveto 208 559 moveto 210 561 lineto 208 559 moveto 206 561 lineto 208 559 moveto 430 559 moveto 437 566 lineto 430 559 moveto 379 610 lineto 430 559 move
eto 858 562 moveto 883 562 moveto 968 562 lineto 883 562 moveto 931 610 lineto 883 562 moveto 14 563 moveto 20 569 lineto 14 563 moveto 12 565 lineto 14 563 moveto 42 56
) 435 567 moveto 437 569 lineto 435 567 moveto 392 610 lineto 435 567 moveto 446 567 moveto 471 592 lineto 446 567 moveto 437 576 lineto 446 567 moveto 506 567 moveto 5
 572 571 moveto 540 603 lineto 572 571 moveto 611 571 moveto 650 610 lineto 611 571 moveto 609 573 lineto 611 571 moveto 783 571 moveto 815 571 lineto 783 571 moveto 79
neto 440 576 moveto 476 576 moveto 487 587 lineto 476 576 moveto 472 580 lineto 476 576 moveto 481 576 moveto 487 582 lineto 481 576 moveto 472 585 lineto 481 576 movet
7 583 lineto 435 581 moveto 406 610 lineto 435 581 moveto 446 581 moveto 471 606 lineto 446 581 moveto 437 590 lineto 446 581 moveto 506 581 moveto 510 585 lineto 506 58
7 589 lineto 435 587 moveto 412 610 lineto 435 587 moveto 446 587 moveto 469 610 lineto 446 587 moveto 437 596 lineto 446 587 moveto 506 587 moveto 510 591 lineto 506 58
593 moveto 463 610 lineto 446 593 moveto 437 602 lineto 446 593 moveto 506 593 moveto 510 597 lineto 506 593 moveto 500 599 lineto 506 593 moveto 572 593 moveto 578 59
```

Jonathan Ermel 2012

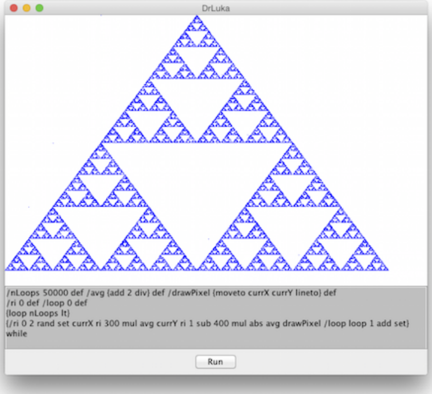


```
moveto 521 577 moveto 524 580 lineto 521 577 moveto 526 577 moveto 529 580 lineto 526
```


1 2 3 4 5 6 7 8 9 Run: scrot
Lecxe - SpreadSheet + Postfix Notation
Mon Mar 10, 21:18

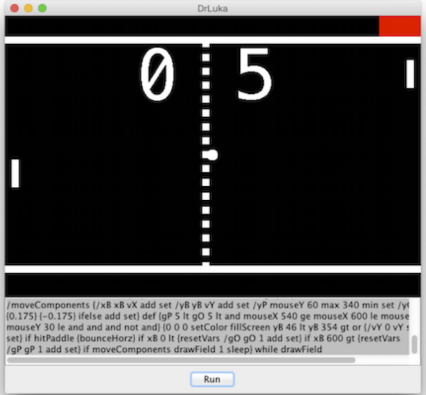
A	B	C	D	E	F	G	H	I	J
/C 0 cell if // 1 /B 2 popcell 30 rand 30 add /C 2 popcell /A 1 cell /C 0 cell 1 sub /C 0 popcell /A 0 cell		500	< # of circles						
/B /A 11 cell /B 3 cell /A 9 cell /B 1 popcell /C /A 11 cell /C 3 cell /A 9 cell /C 1 popcell /A 7 cell			< circle coords						
/C 2 cell if // /A 3 cell /C 2 cell 1 sub /C 2 popcell /A 2 cell			< circle size						
/B 1 cell /B 2 cell 4 div sub /C 1 cell /B 2 cell 4 div sub moveto /A 4 cell	1000	250	< canvas size	^rgb params					
/B 2 cell 0 360 arc /B 2 cell 1 add /B 2 popcell			/pop.luka dump						
100 155 rand add dup 100 155 rand add dup 3 1 roll 100 155 rand add dup 4 1 roll color /A 6 cell									
/E 2 popcell /E 1 popcell /E 0 popcell									
3 rand dup /E exch cell 20 rand 10 sub add dup 255 gt if // pop 255 // pop /A 8 cell /A 2 cell									
dup 100 lt if // pop 100 // pop exch /E exch popcell /E 0 cell /E 1 cell /E 2 cell color									
ge if // 0 // pop									
/A 5 cell /B 3 cell rand /B 1 popcell 0 /C 1 popcell /A 0 cell									
/B 2 cell dup 20 rand 20 add add exch 2 div sub exch 1 cell add dup									

William Pearson 2014



```
/nLoops 50000 def /avg {add 2 div} def /drawPixel {moveto currX currY lineto} def
/ri 0 def /loop 0 def
{loop nLoops lt}
{/ri 0 2 rand set currX ri 300 mul avg currY ri 1 sub 400 mul abs avg drawPixel /loop loop 1 add set}
while
```

Matthew Pollex 2015



```
/moveComponents {/xB xB vX add set /yB yB vY add set /yP mouseY 60 max 340 min set /y
{0.175} {-0.175} ifelse add set} def {gP 5 lt gO 5 lt and mouseX 540 ge mouseX 600 le mouse
mouseY 30 le and and and not and} {0 0 0 setColor fillScreen yB 46 lt yB 354 gt or {/vY 0 vY
set} if hitPaddle {bounceHorz} if xB 0 lt {resetVars /gO gO 1 add set} if xB 600 gt {resetVars
/gP gP 1 add set} if moveComponents drawField 1 sleep} while drawField
```

Matthew Pollex 2015 (AI Pong)

Learning objectives of the course

- The main objective of the course is to learn how to implement and use **abstract data types**
  - In particular, lists, stacks, queues and binary search trees.
- In order to develop elegant and robust solutions, knowledge of the **object-oriented programming**, **encapsulation**, **inheritance** and **polymorphism** are necessary.
- We'll also be learn about **I/O**, **user interfaces**, and **JUnit**.

Difficulties

- The **high-level** concepts are simple!
- The main difficulty consists of **translating the high-level ideas into a concrete implementation**!

Difficulties

Google in Growth Mode in Montreal
La Presse, published February 1, 2011.
(`https://bit.ly/2rYp9TV`)

"(. . .) **very strong programming skills** (. . .) We don't just want people **who have ideas**. We want someone who can **sit down and code**."

"The directive we've received is to **hire as many good employees as we can recruit**. There is no quota on the quantity, but we have a very high quality threshold."

"The teams from Toronto, Ottawa and Waterloo will also benefit from a jolt of the accelerator."

We tend to look at the teacher/learner relationship the wrong way around: it's not that the teacher teaches; **it's that the student learns**.

Pragmatic Thinking and Learning — Andy Hunt

Difficulties

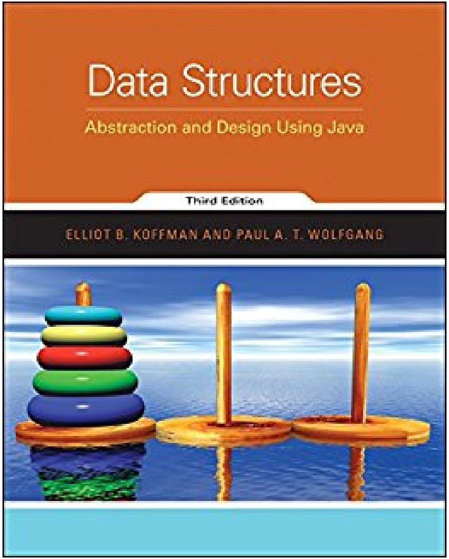
- Don't cut-and-paste source code, **rewrite the examples by yourself!**
- **Do all the assignments!**
- **Do all the laboratories!**
- Keep a **positive attitude** towards programming!
- **Smile!**

Difficulties (continued): a section from 2016

20	19%	A+
4	4%	A
11	10%	A-
3	3%	B+
5	5%	B
6	6%	C+
3	3%	C
1	1%	D+
1	1%	D
6	6%	E
3	3%	F
4	4%	EIN
3	3%	ABS
38	35%	Abandon

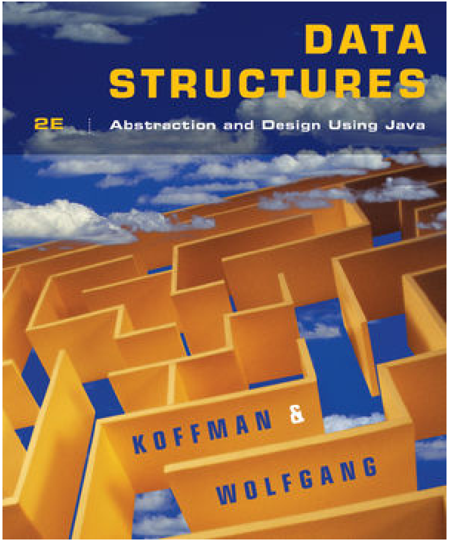
Textbook

- E. B. Koffman, P. A. T. Wolfgang. Data Structures: Abstraction and Design Using Java. John Wiley & Sons, 3e edition, 2016.


Data Structures
Abstraction and Design Using Java
Third Edition
ELLIOT B. KOFFMAN AND PAUL A. T. WOLFGANG

Textbook

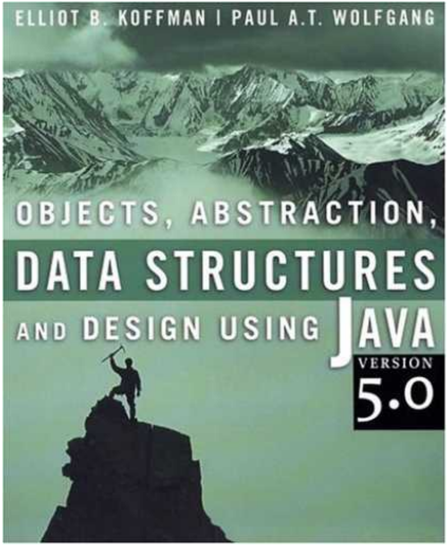
- E. B. Koffman and P. A. T. Wolfgang. Data Structures: Abstraction and Design Using Java. John Wiley & Sons, 2e edition, 2010.



(Alternative) textbook

Could you recommend an affordable alternative?

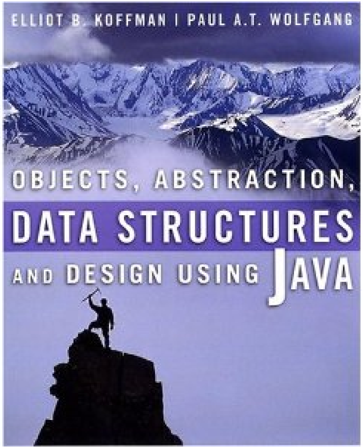
- E. Koffman and P. Wolfgang (**2005**) *Objects, Abstraction, Data Structures and Design: Using Java Version 5.0*. Wiley, 880 pages. (ISBN: 0-471-69264-6)



(Alternative) textbook

Could you recommend an affordable alternative?

- E. Koffman and P. Wolfgang (**2005**) *Objects, Abstraction, Data Structures and Design: Using Java*. Wiley, 864 pages. (ISBN: 0-471-46756-1)



(Alternative) textbook

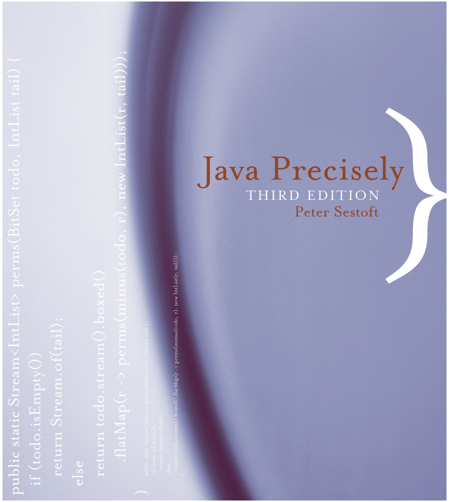
Could you recommend an even more affordable alternative?

- Java Structures: Data Structures in Java for the Principled Programmer by Duane A. Bailey
  - www.cs.williams.edu/∼bailey/JavaStructures/Book.html

Book on Java

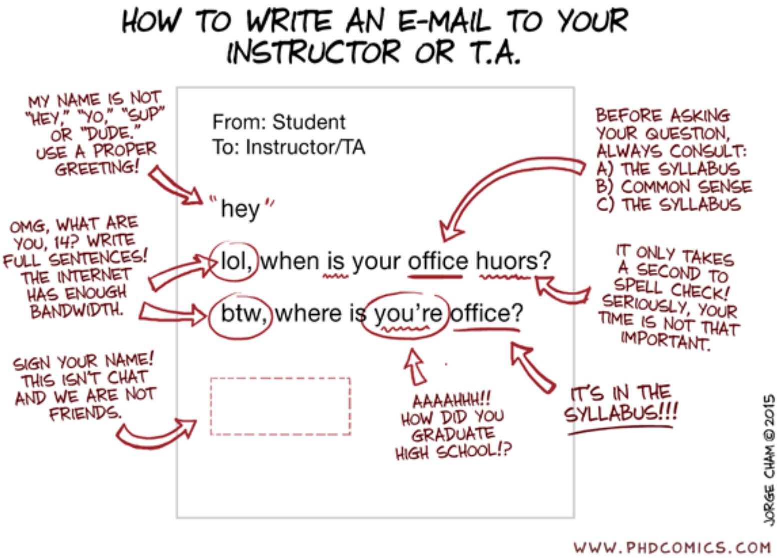
I am new to programming in Java!

- P. Sestoft. Java Precisely. The MIT Press, 3e edition, August 2016. (∼ 40 $)


Java Precisely
THIRD EDITION
Peter Sestoft
public static Stream<IntList> perms(BitSet todo, IntList tail) {
if (todo.isEmpty())
return Stream.of(tail);
else
return todo.stream().boxed()
.flatMap(r -> perms(minus(todo, r), new IntList(r, tail)));
}

Epilogue

Netiquette


HOW TO WRITE AN E-MAIL TO YOUR INSTRUCTOR OR T.A.
MY NAME IS NOT "HEY," "YO," "SUP" OR "DUDE." USE A PROPER GREETING!
From: Student
To: Instructor/TA
"hey"
lol, when is your office huors?
btw, where is you're office?
OMG, WHAT ARE YOU, 14? WRITE FULL SENTENCES! THE INTERNET HAS ENOUGH BANDWIDTH.
SIGN YOUR NAME! THIS ISN'T CHAT AND WE ARE NOT FRIENDS.
BEFORE ASKING YOUR QUESTION, ALWAYS CONSULT: A) THE SYLLABUS B) COMMON SENSE C) THE SYLLABUS
IT ONLY TAKES A SECOND TO SPELL CHECK! SERIOUSLY, YOUR TIME IS NOT THAT IMPORTANT.
AAAAHHH!! HOW DID YOU GRADUATE HIGH SCHOOL!?
IT'S IN THE SYLLABUS!!!
JORGE CHAM © 2015
WWW.PHDCOMICS.COM

title: "How To Write An E-mail To Your Instructor Or Teaching Assistant" - originally published 4/22/2015

Prologue

Summary

- You need to get at least 50% for the exam portion of the grades.
- **Do** all the assignments.
- **Do** all laboratories.
- **Don't succumb** to plagiarism.

Next Module

- Introduction to **object-oriented programming**

References I

E. B. Koffman and Wolfgang P. A. T.
***Data Structures: Abstraction and Design Using Java*.**
John Wiley & Sons, 3e edition, 2016.

E. B. Koffman and Wolfgang P. A. T.
***Data Structures: Abstraction and Design Using Java*.**
John Wiley & Sons, 2e edition, 2010.

P. Sestoft.
***Java Precisely*.**
The MIT Press, 3e edition, 2016.

Marcel **Turcotte**

Marcel.Turcotte@uOttawa.ca

School of Electrical Engineering and **Computer Science** (EECS)

University of Ottawa