COSC-111
Introduction to Computer Science I

Sections 2 & 3

Lee Spector

Registration

- Sections 2 & 3: Same course
- Section 1 (Matteo Riondato): Same core content, some differences
- Some lab switching possible, but can't mix Matteo/Lee
- If not registered but want to be, request on ACData and email me
- **You must sign sign-in sheet to remain registered**

- Introductions
- Course information
- Computation and programming
- First programs

- **Introductions**
- Course information
- Computation and programming
- First programs

You

- Name
- Pronouns
- Year
- Major or primary interests

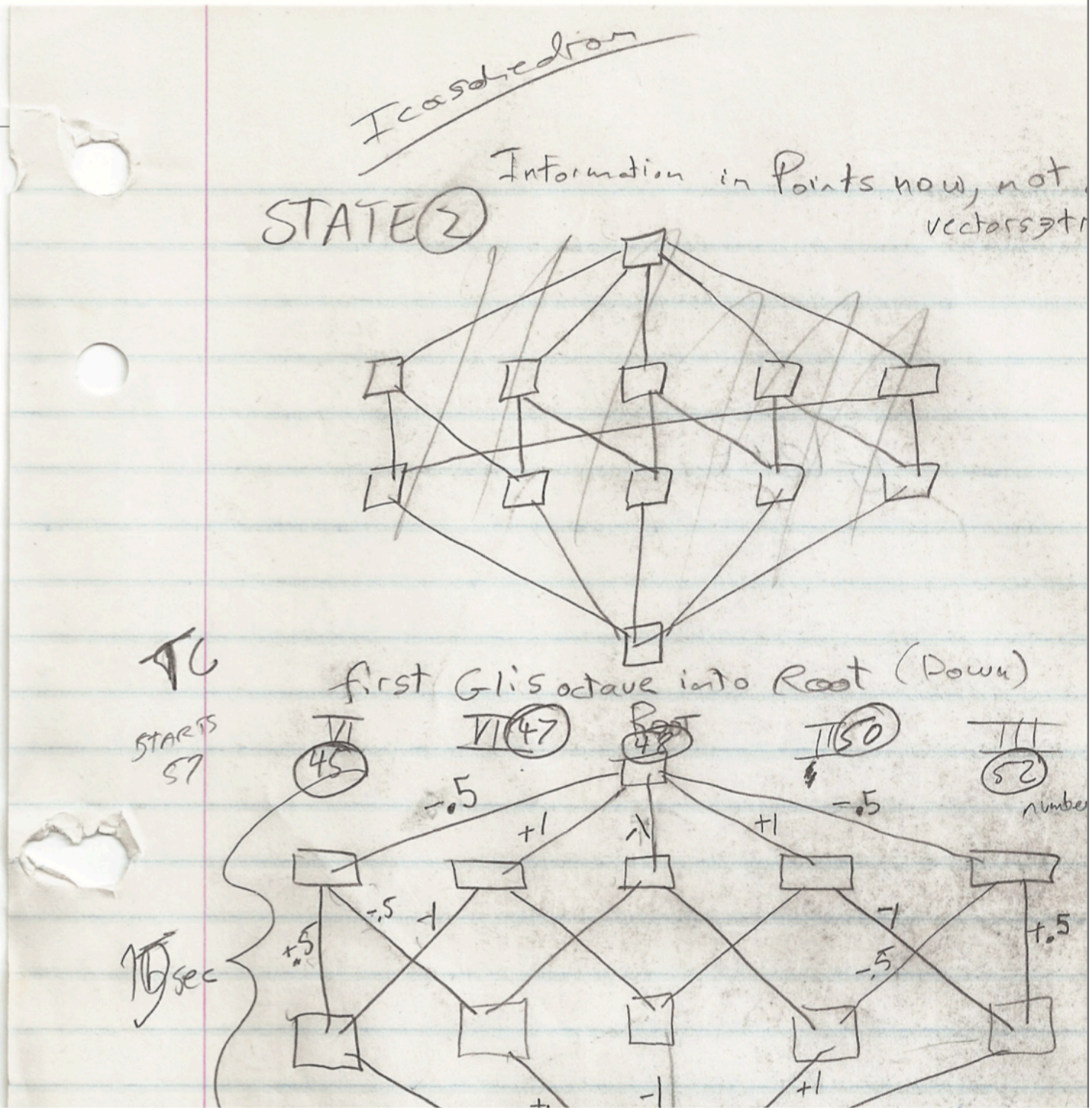
MUSIC

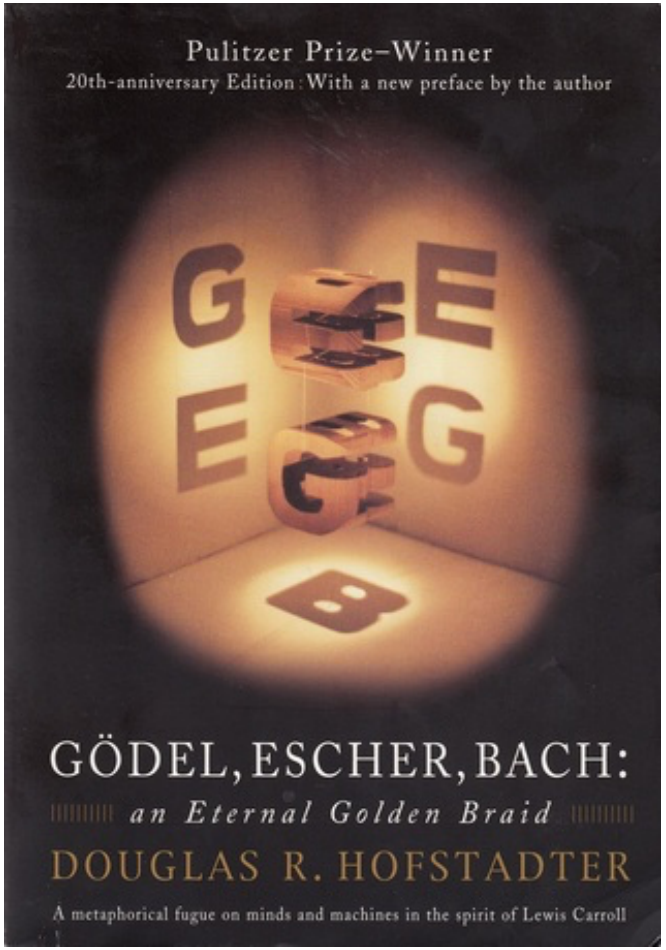
The Spector Sound

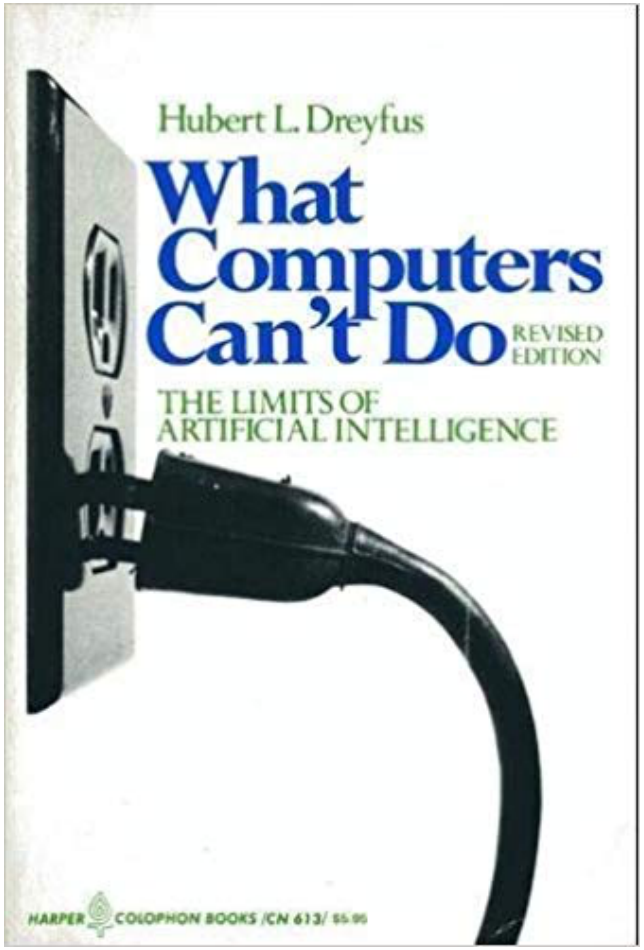
This diagram is a composition called "Geometric Transformations (Nude Brunch)," created in 1981 by Lee Spector '84 in a TIMARA class taught by Gary Lee Nelson. In a windowless room on the fourth floor of Mudd, students in the class composed music by creating code on computer terminals that were connected to a mainframe computer (Spector, a professor of computer science at Hampshire College, believes it was a Xerox Sigma 9) in the building's basement.

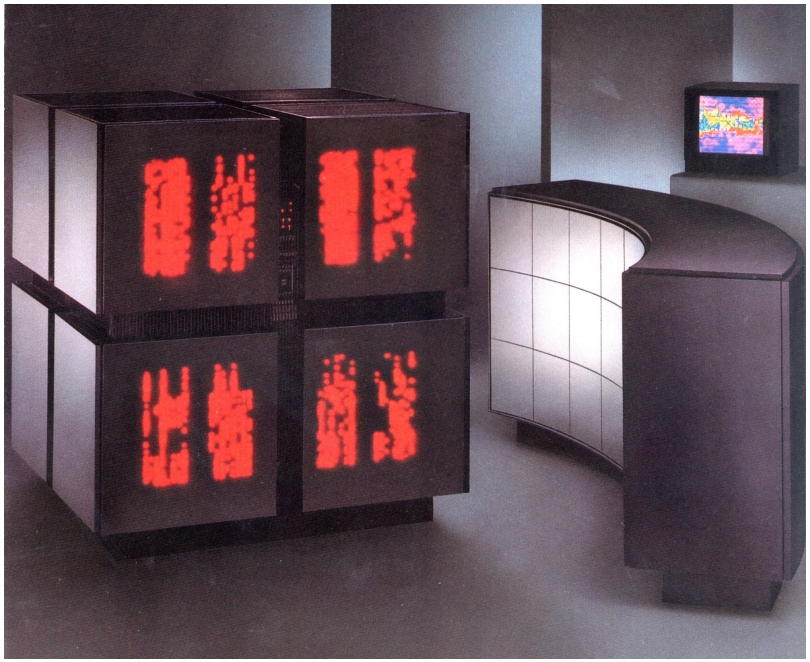
"'Composition' was development of the ideas and the code," he says, "and 'performance' was getting the computer to produce the sound.

"I happened to be a big Buckminster Fuller fan at the time," Spector explains, "and I decided to make a piece that translated the geometric objects he described in his




Pulitzer Prize–Winner
20th-anniversary Edition: With a new preface by the author
GÖDEL, ESCHER, BACH:
an Eternal Golden Braid
DOUGLAS R. HOFSTADTER
A metaphorical fugue on minds and machines in the spirit of Lewis Carroll


Hubert L. Dreyfus
What Computers Can't Do
REVISED EDITION
THE LIMITS OF ARTIFICIAL INTELLIGENCE
HARPER COLOPHON BOOKS /CN 613/ $5.95



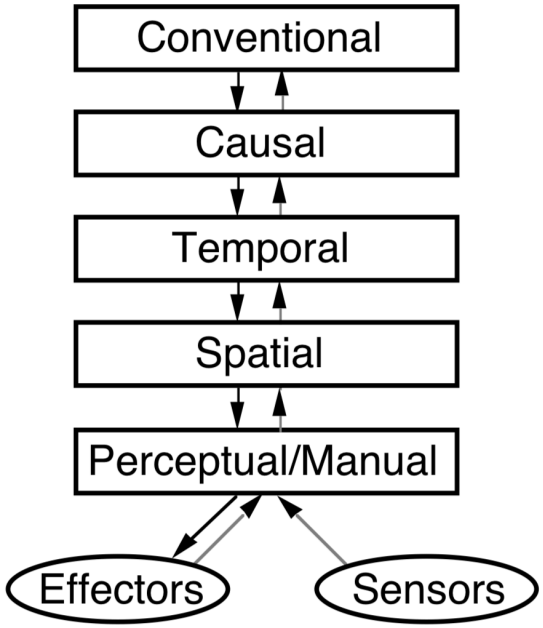


Figure 7. The specific levels of APE.

ELSEVIER

COGNITIVE SCIENCE

Cognitive Science 25 (2001) 941–975

http://www.elsevier.com/locate/cogsci

Partial and total-order planning: evidence from normal and prefrontally damaged populations

Mary Jo Rattermann[a,*], Lee Spector[b], Jordan Grafman[c], Harvey Levin[d], Harriet Harward[e]

[a]*Department of Psychology, Franklin & Marshall College, Lancaster, PA 17604, USA*
[b]*School of Cognitive Science, Hampshire College, Amherst, MA 01002, USA*
[c]*National Institute of Neurological Disorders and Stroke, Building 10; Room 5C205; 10 Center Drive; MSC 1440, Bethesda, MD 20892-1440, USA*
[d]*Department of Physical Medicine and Rehabilitation, Baylor University College of Medicine, 1333 Moursound Avenue, A 205, Houston, TX 77030, USA*
[e]*Callier Center for Communication Disorders, University of Texas at Dallas, 1966 Inwood Road, Dallas TX 75235, USA*

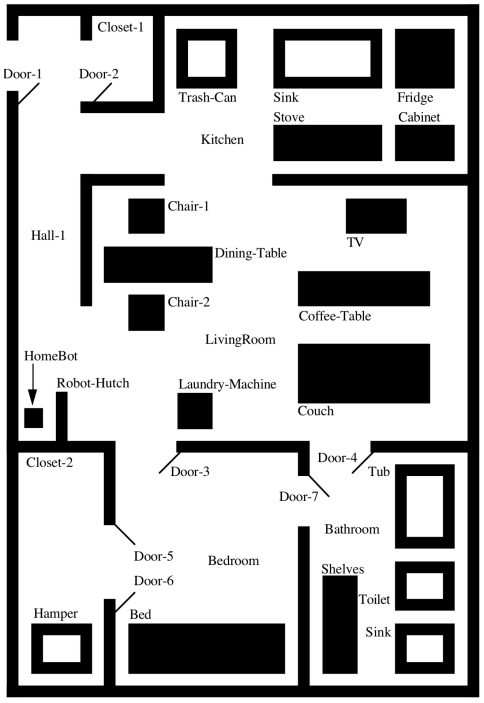
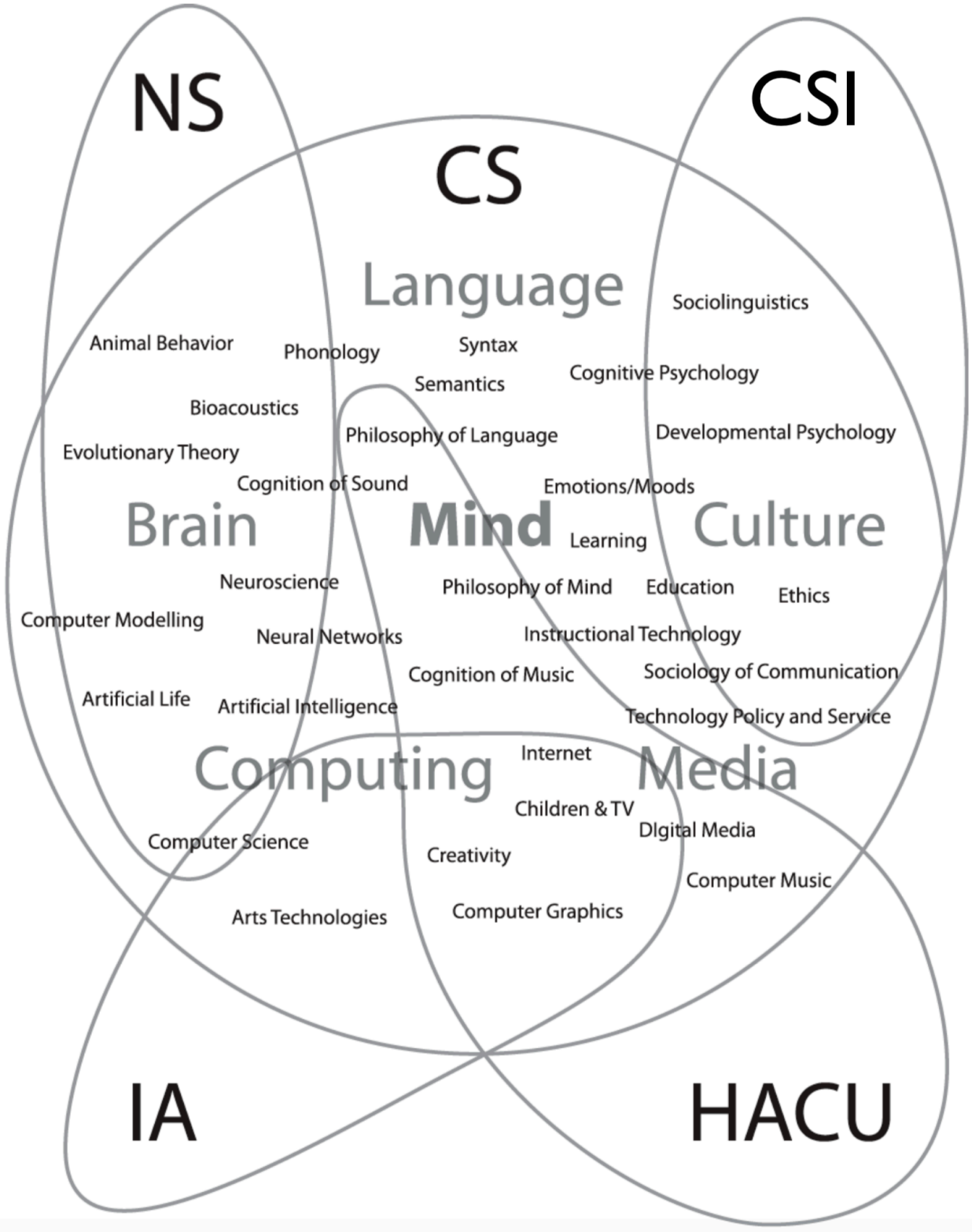


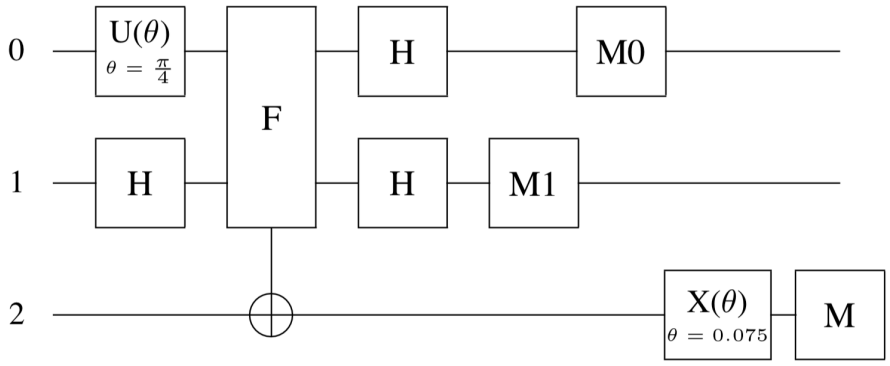
Figure 19. HomeBot's domain.

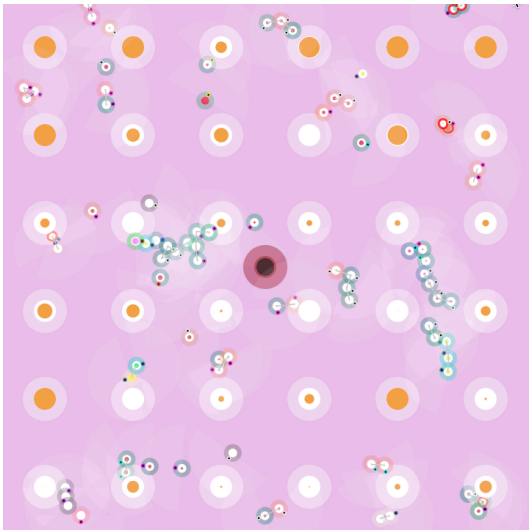

NS
CS
CSI
Language
Sociolinguistics
Animal Behavior
Phonology
Syntax
Cognitive Psychology
Semantics
Bioacoustics
Philosophy of Language
Developmental Psychology
Evolutionary Theory
Cognition of Sound
Emotions/Moods
Brain
Mind
Culture
Learning
Neuroscience
Philosophy of Mind
Education
Ethics
Computer Modelling
Neural Networks
Instructional Technology
Cognition of Music
Sociology of Communication
Artificial Life
Artificial Intelligence
Technology Policy and Service
Computing
Internet
Media
Children & TV
Dlgital Media
Computer Science
Creativity
Computer Music
Arts Technologies
Computer Graphics
IA
HACU

Advanced Topics in Artificial Intelligence
Algorithmic Arts
Animals and Animats: Natural and Artificial Intelligence and Behavior Artificial Intelligence
Artificial Intelligence in 3D Virtual Worlds
Beginning Coding for Science
Biocomputational Developmental Ecology
Code Immersion
Cognitive Science Fiction
Computational Models of Biological Systems
Computer Science Projects
Computing Concepts: Creative Machines?
Creative Programming Workshop
Current Issues in Cognitive Science
Evolutionary Computation
Genetic Programming
Hypertext
Introduction to Artificial Intelligence
Introduction to Cognitive Science
Introduction to Computer Science
Programming Creativity
Programming for Science
Programming Game Theory
Programming Language Paradigms
Quantum Computing with No Prerequisites of Any Kind
Radical Innovation in Digital Arts
Reasoning About Action
Research in Artificial Intelligence
Unconventional Computing
What Computers Can’t Do (limits of computing)
When Machines Talk (natural language processing)

Integrated Teaching & Research

- Undergrad/grad/faculty collaboration
- Wide range of project areas
- Five College research group focusing on evolutionary computing





- Introductions
- **Course information**
- Computation and programming
- First programs

Computer Science 111-02: Introduction to Computer Science I

NAVIGATION

Dashboard
Site home
Site pages
My courses
1920F
Computer Science
Introduction to Computer Science I 02 (COSC-111-02-1920F)
Participants
Grades
General
September 2 - September 8
September 9 - September 15
September 16 - September 22
September 23 - September 29
September 30 - October 6
October 7 - October 13
October 14 - October 20
October 21 - October 27
October 28 - November 3
November 4 - November 10
November 11 - November 17
November 18 - November 24
November 25 - December 1
December 2 - December 8
December 9 - December 15
December 16 - December 22

Syllabus

General course information and policies. Be sure to read all of this carefully.

Detailed Schedule

This is where you'll find detailed information about what we're doing each day, what you should read and turn in, etc. Check back frequently since this may be adjusted as we proceed.

Announcements

Anonymous Forum

Please post questions and answers to questions here.

Check for Print Reserves

E-reserves

Textbook Code

This is the code distributed with our textbook. We will use little of it directly, but I have posted it here for ease of access.

September 2 - September 8

Reminder regarding detailed schedule

Matteo's instructions for installing Java SDK and IntelliJ

RollDie.java

LATEST NEWS

Add a new topic...

(No announcements have been posted yet.)

QUICKSETS

Yes | No

Students see course?

Grades visible?

Update settings

More settings

"This block is not visible to students"

Amherst College, Fall 2019

COSC-111: Introduction to Computer Science I

Instructor: Professor Lee Spector (he/him), SCCE C211, lspector@amherst.edu

Class Meetings:

Section 02: MW 11:00-11:50 in SCCE A131, Lab F 10:00-10:50 or 11:00-11:50 in SCCE A331

Section 03: MW 2:00- 2:50 in SMUD 207, Lab F 1:00-1:50 or 2:00-2:50 in SCCE A331

Office hours:

Individual (sign up here): M 4:00-5:30, W 3:00-4:00

Open (group, no signup required): M 3:00-4:00, W 4:00-5:30

Additional times by appointment (email)

Teaching assistants:

Section 02: Ben Fleischman, in SCCS A131 Wednesday evenings, 7:00-9:00

Section 03: Caroline Shim, in SCCS A131 Sunday evenings, 7:00-9:00

Additional TA help available in SCCS A131 Sunday, Monday and Wednesday evenings, 7:00-9:00

Overview: This course is an introduction to computer science, designed for students with no previous computer science or programming experience. We will spend most of our time learning to write programs in the Java programming language, but while doing so we will also engage with a range of other topics including: techniques for designing step-by-step processes to solve problems (algorithms); applications of computers to the social sciences and the arts; fundamental concepts of artificial intelligence and artifical life; ethical issues related to the development and use of computer technology; alternative conceptions of programming and programming languages; and the limits of computation (complexity and computability theory).

Texts: We will read documents distributed on the class Moodle, along with selected parts of:

- *Introduction to Programming in Java: An Interdisciplinary Approach (2nd Edition)*, by Robert Sedgewick and Kevin Wayne **(S&W)**: You may purchase this (for example from Amazon) or read the copies on reserve at the library. This entire book is also the first half of *Computer Science: An Interdisciplinary Approach*, by the same authors. We will only be using material in the *Introduction to Programming in Java* part, but you can read it just as well from the *Computer Science* book if you want to get that instead (although it's bigger and more expensive).
- *Java Programming*, by Lyle McGeoch **(LM)**: Free online with Amherst login.

Software: IntelliJ IDEA

Support: My goal is for you to succeed in this course, to learn a lot and to earn a good grade, regardless of your background and regardless of whether or not your future academic plans involve more computer science. If you are having trouble, then I want to know about it and I want to help. Please make proactive use of the support systems built into the course (scheduled Q&A sessions, the Moodle forum, TA office hours, and my office hours), and contact me if you may need additional support such as a peer tutor.

Accommodations: If you have a documented disability that requires accommodations, you will need to register with Accessibility Services for coordination of your academic accommodations. You can reach them via email at accessibility@amherst.edu, or via phone at 413-542-2337. Once you have your accommodations in place, I will be glad to meet with you privately during my office hours or at another agreed upon time to discuss the best implementation of your accommodations.

Attendance and participation: You are expected attend and participate in all classes and labs, except when you really can't or shouldn't, for example because of illness. While attendance will not be recorded and will not contribute *directly* to your grade, it will nonetheless contribute significantly, *indirectly*, by helping you to master the material, to complete the exercises, and to do well on the exams.

Grading: Midterm exam: 20%, Final exam: 30%, Exercises: 50%, with each week's exercises counting equally. The exercises for the last few weeks of the semester all focus on the development of a final project, so the final project will count as several weeks of exercises. Partial credit will be awarded for incomplete submissions, but **late submissions will receive no credit at all** (unless required by official accommodations or requested by your class dean). So it is a good idea to plan to submit everything at least a day or two before the deadline, and you should always submit *something* on time even if it is not complete.

Exercises:

- **Collaborative exercises: group work is encouraged.** Work on collaborative exercises starts in lab sessions, in groups. You will turn in your own answers, which must be unique, but these exercises will generally be relatively open-ended so that your own answers can naturally reflect your own interests and tastes. You should discuss your ideas, get help from classmates, and help them too, both during lab sessions and afterwards. Your answers can include code written by others (classmates or websites, etc.) as long as you make the source of every piece of code explicit.

- **Individual exercises: group work is forbidden.** You must not discuss or share answers to individual exercises with anyone except the professor or TAs. Individual exercises will usually involve re-using concepts from previous collaborative exercises. Here you have to demonstrate your understanding of those concepts on your own, and unless otherwise noted you can only include code that you have written yourself from scratch.

- **Practice exercises: optional, not to be submitted, discuss freely.** Answers to practice exercises will not be collected, but you are encouraged to complete them in order to ensure that you understand the material, and in order to prepare for the exams. You are also encouraged to ask about these in class Q&A sessions, on the Moodle forum, and in TA hours, and to discuss them with classmates in lab sessions. Practice exercises designated with a `(!)` are particularly recommended.

Labs: Lab sessions will be used primarily for starting work on newly assigned collaborative exercises. Time permitting, they may also be used to discuss and work on practice exercises.

Moodle forum: You are strongly encouraged to submit questions and answers to questions on the anonymous class Moodle forum, to which you should all be subscribed. I will not look at identities of posters unless this is necessary to deal with a problem, and identities will not be visible to classmates. While participation on the forum will not contribute directly to your grade, it is a valuable learning opportunity, whether you are asking questions, answering questions, or both.

Laptops, phones, and other devices: You should feel free to use whatever devices you want, but you should be 100% engaged with the class throughout every class meeting. If it helps then you can use devices to take notes, to look things up, maybe even to run small code examples related to the topic under discussion, etc. But you should be focused on the activity of the class at all times. No devices or other materials are permitted during exams, unless required by official accommodations.

Honor code: The Amherst College Honor Code applies to this course, as it does to all other Amherst College courses. You should attend carefully to the requirement for explicit attribution of sources for collaborative exercises, and to the prohibition of discussions or sharing of work on individual exercises. Sharing of exam questions or answers, at any time, is prohibited.

Schedule Overview:

- Motivation and goals
- Getting started with Java programming
- ASCII art
- Types, variables, expressions, operators
- Command-line arguments
- Random numbers and text
- Simulating elections
- User input
- Defining functions (static methods)
- Grammatical sentence generators, Madlibs
- Shape-drawing library
- Loops (`while` and `for`)
- Drawing and animation generators
- Simulating voter preferences
- File input and output
- Midterm exam

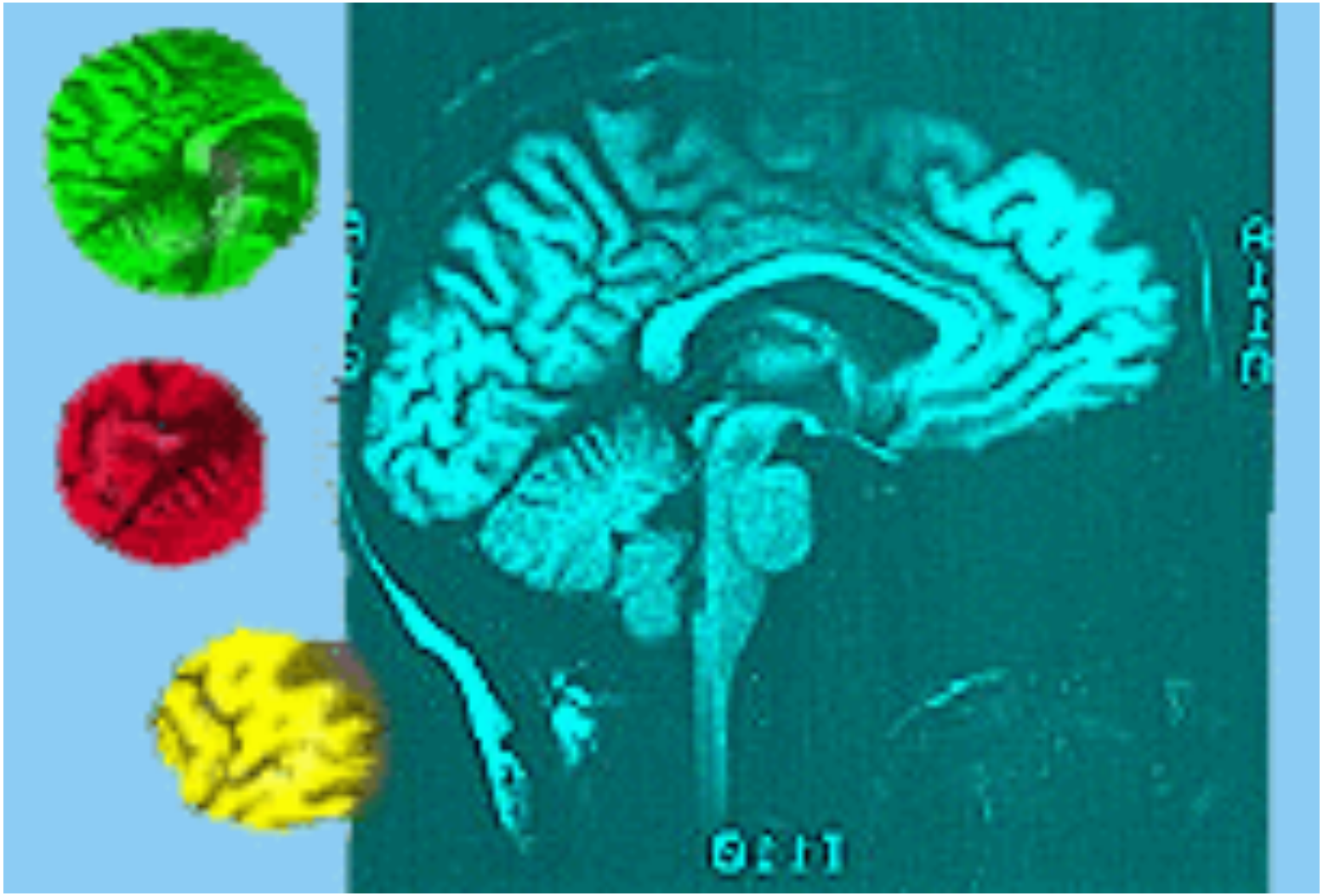
- Enhancing previously-written programs
- Arrays
- Cellular automata and the "game of life"
- Defining Java classes and objects
- Simulating instant-runoff elections and elections with primaries
- Recursion
- Fractal trees and ferns
- Object-oriented programming concepts
- Defining instance methods
- Exceptions
- Simulating antagonism across social groups
- Other programming paradigms
- Computing and social responsibility
- Final project development
- Computational complexity
- Computability and unsolvable problems
- Artificial intelligence
- Final projects due
- Final exam

- Introductions
- Course information
- **Computation and programming**
- First programs

Computation

?

Computation



Computation

(marble run gates)

Angewandte International Edition Chemie

GDCh A Journal of the German Chemical Society

Communication

A Molecular Circuit Regenerator to Implement Iterative Strand Displacement Operations

Nicole V. DelRosso, Prof. Dr. Sarah Hews, Prof. Dr. Lee Spector, Prof. Dr. Nathan D. Derr

First published: 21 March 2017 | https://doi.org/10.1002/anie.201610890 | Cited by: 4

Abstract

The predictable chemistry of Watson–Crick base-pairing imparts a unique structural programmability to DNA, enabling the facile design of molecular reactions that perform computations. However, many of the current architectures limit devices to a single operational cycle. Herein, we introduce the design of the “regenerator”, a device based on coupled enthalpic and entropic reactions that permits the regeneration of molecular circuit components.

Bit

0 1

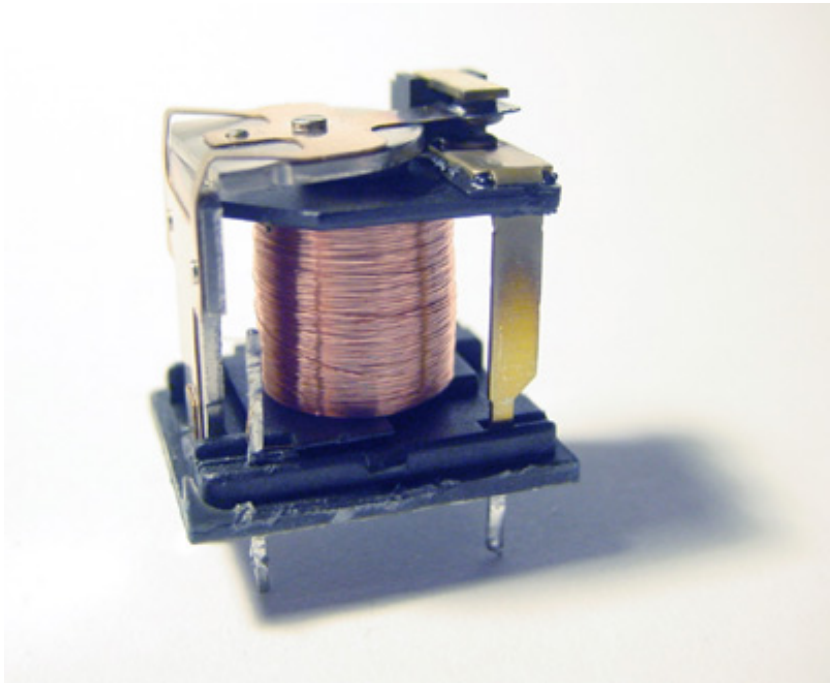
off on

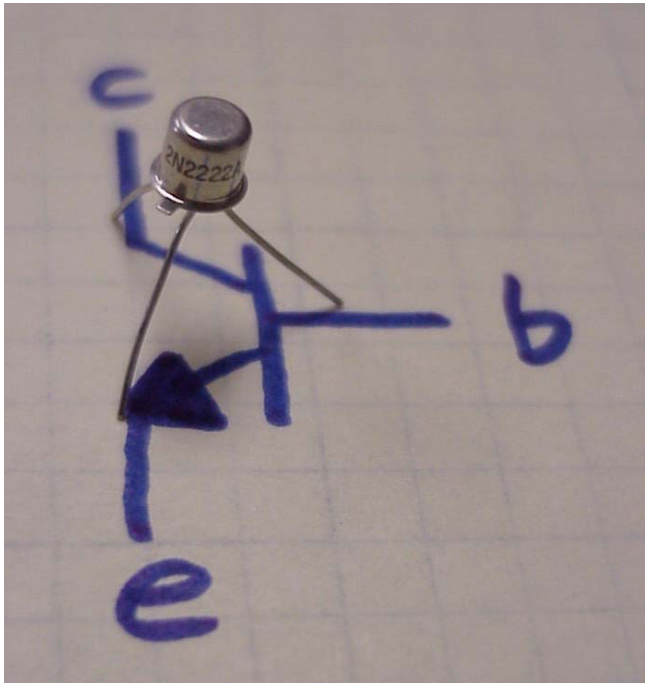
Can be used to represent *any* information

Binary Numbers

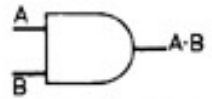
0
1
10
11
100
101
110
111
1000
1001
1010
1011
1100
1101
1110
1111

Dec	Bin	Hex	Char	Dec	Bin	Hex	Char	Dec	Bin	Hex	Char	Dec	Bin	Hex	Char
0	0000 0000	00	[NUL]	32	0010 0000	20	space	64	0100 0000	40	@	96	0110 0000	60	`
1	0000 0001	01	[SOH]	33	0010 0001	21	!	65	0100 0001	41	A	97	0110 0001	61	a
2	0000 0010	02	[STX]	34	0010 0010	22	"	66	0100 0010	42	B	98	0110 0010	62	b
3	0000 0011	03	[ETX]	35	0010 0011	23	#	67	0100 0011	43	C	99	0110 0011	63	c
4	0000 0100	04	[EOT]	36	0010 0100	24	$	68	0100 0100	44	D	100	0110 0100	64	d
5	0000 0101	05	[ENQ]	37	0010 0101	25	%	69	0100 0101	45	E	101	0110 0101	65	e
6	0000 0110	06	[ACK]	38	0010 0110	26	&	70	0100 0110	46	F	102	0110 0110	66	f
7	0000 0111	07	[BEL]	39	0010 0111	27	'	71	0100 0111	47	G	103	0110 0111	67	g
8	0000 1000	08	[BS]	40	0010 1000	28	(	72	0100 1000	48	H	104	0110 1000	68	h
9	0000 1001	09	[TAB]	41	0010 1001	29	)	73	0100 1001	49	I	105	0110 1001	69	i
10	0000 1010	0A	[LF]	42	0010 1010	2A	*	74	0100 1010	4A	J	106	0110 1010	6A	j
11	0000 1011	0B	[VT]	43	0010 1011	2B	+	75	0100 1011	4B	K	107	0110 1011	6B	k
12	0000 1100	0C	[FF]	44	0010 1100	2C	,	76	0100 1100	4C	L	108	0110 1100	6C	l
13	0000 1101	0D	[CR]	45	0010 1101	2D	-	77	0100 1101	4D	M	109	0110 1101	6D	m
14	0000 1110	0E	[SO]	46	0010 1110	2E	.	78	0100 1110	4E	N	110	0110 1110	6E	n
15	0000 1111	0F	[SI]	47	0010 1111	2F	/	79	0100 1111	4F	O	111	0110 1111	6F	o
16	0001 0000	10	[DLE]	48	0011 0000	30	0	80	0101 0000	50	P	112	0111 0000	70	p
17	0001 0001	11	[DC1]	49	0011 0001	31	1	81	0101 0001	51	Q	113	0111 0001	71	q
18	0001 0010	12	[DC2]	50	0011 0010	32	2	82	0101 0010	52	R	114	0111 0010	72	r
19	0001 0011	13	[DC3]	51	0011 0011	33	3	83	0101 0011	53	S	115	0111 0011	73	s
20	0001 0100	14	[DC4]	52	0011 0100	34	4	84	0101 0100	54	T	116	0111 0100	74	t
21	0001 0101	15	[NAK]	53	0011 0101	35	5	85	0101 0101	55	U	117	0111 0101	75	u
22	0001 0110	16	[SYN]	54	0011 0110	36	6	86	0101 0110	56	V	118	0111 0110	76	v
23	0001 0111	17	[ETB]	55	0011 0111	37	7	87	0101 0111	57	W	119	0111 0111	77	w
24	0001 1000	18	[CAN]	56	0011 1000	38	8	88	0101 1000	58	X	120	0111 1000	78	x
25	0001 1001	19	[EM]	57	0011 1001	39	9	89	0101 1001	59	Y	121	0111 1001	79	y
26	0001 1010	1A	[SUB]	58	0011 1010	3A	:	90	0101 1010	5A	Z	122	0111 1010	7A	z
27	0001 1011	1B	[ESC]	59	0011 1011	3B	;	91	0101 1011	5B	[	123	0111 1011	7B	{
28	0001 1100	1C	[FS]	60	0011 1100	3C	<	92	0101 1100	5C	\	124	0111 1100	7C	|
29	0001 1101	1D	[GS]	61	0011 1101	3D	=	93	0101 1101	5D	]	125	0111 1101	7D	}
30	0001 1110	1E	[RS]	62	0011 1110	3E	>	94	0101 1110	5E	^	126	0111 1110	7E	~
31	0001 1111	1F	[US]	63	0011 1111	3F	?	95	0101 1111	5F	_	127	0111 1111	7F	[DEL]



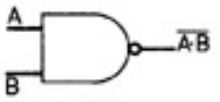

c
2N2222A
b
e

1



B	A	Output
0	0	0
0	1	0
1	0	0
1	1	1

2

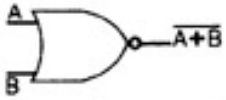


B	A	Output
0	0	1
0	1	1
1	0	1
1	1	0

3

B	A	Output
0	0	0
0	1	1
1	0	1
1	1	1

4



B	A	Output
0	0	1
0	1	0
1	0	0
1	1	0

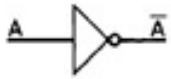
5

B	A	Output
0	0	0
0	1	1
1	0	1
1	1	0

6

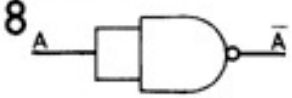
B	A	Output
0	0	1
0	1	0
1	0	0
1	1	1

7



A	Output
0	1
1	0

8



A	Output
0	1
1	0

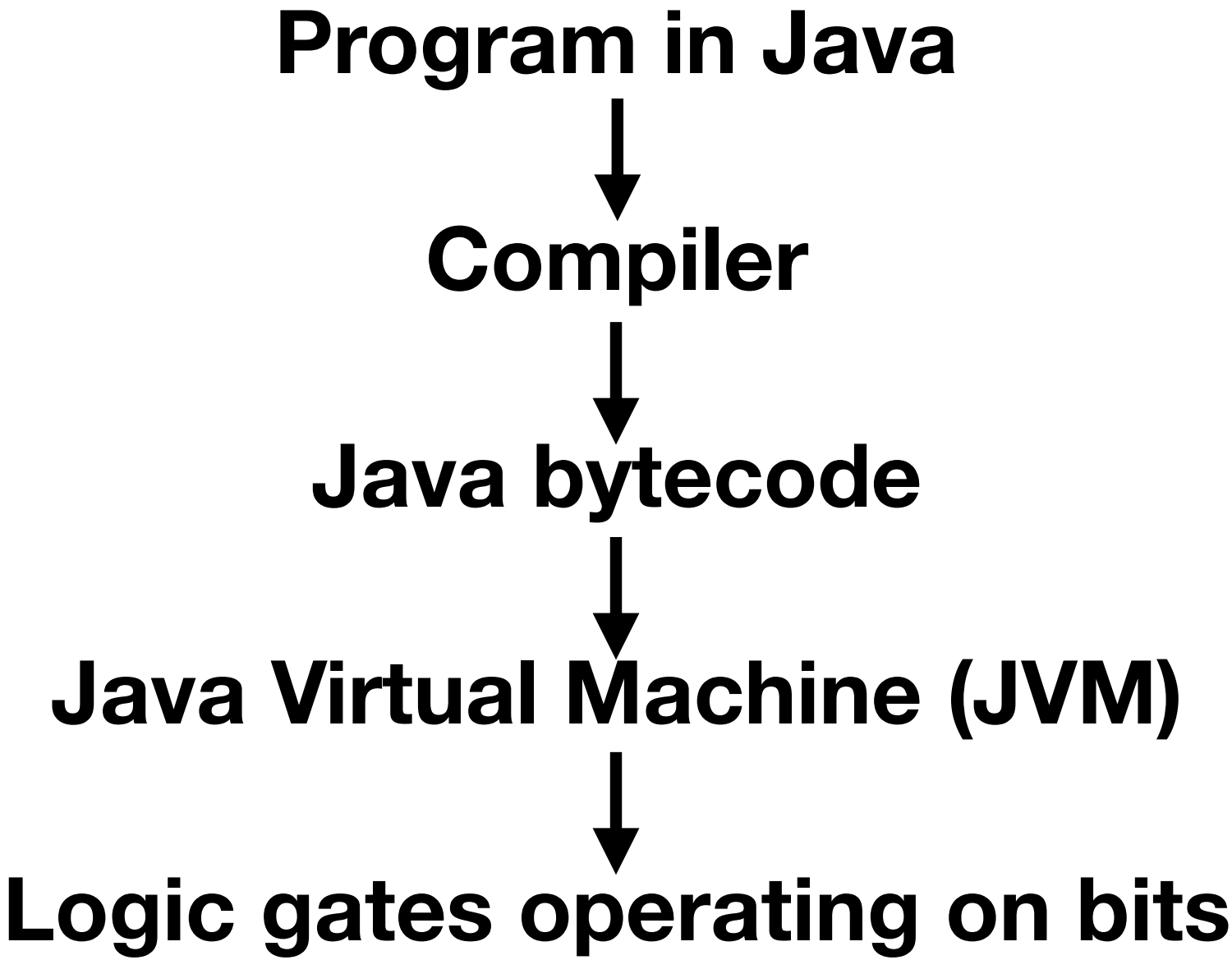
9

A	Output
0	1
1	0

Program in a high-level language

Compiler

Logic gates operating on bits


Program in Java
Compiler
Java bytecode
Java Virtual Machine (JVM)
Logic gates operating on bits

MIT CSAIL
@MIT_CSAIL

Describe programming in only six words.

We'll RT all the best ones.

Ours:

Turning ideas and caffeine into code.

#ProgrammingIn6Words #wednesdaywisdom

11:00 AM · Jun 26, 2019 · TweetDeck

401 Retweets **1.1K** Likes

Crafting instructions to do important things. @JeffDean

Build. Deploy. Test. Debug. Debug. Debug. @StackOverflow

It's a feature, not a bug @leslieasheppard

Making complex things appear simple. @Grady_Booch

Being forced to think very clearly @erikbryn

Try it. Get feedback. Learn. Repeat. @PragmaticAndy

Writing code to write less code @alhuelamo

Turning confused computers into helpful friends @ossia

The computer cooks. I write recipes. @marcorobotics

It ran on my machine yesterday @unrahu1

Why is this still not working?! @ASpittel

Did you remember to clear your cache? @aburke626

Oh WOW, it's working — but how? @codeorg

Who wrote this garbage? Oh, me. @Fobwashed

Hello, world. All are welcome here. @megahoch

Carol Willing
@WillingCarol

Replying to @MIT_CSAIL

Turning bits into works of wonder

11:09 PM · Jun 26, 2019 · Twitter for Android

Lee Spector
@leespector

Replying to @MIT_CSAIL

Orchestrating the flow of the universe

1:48 PM · Jun 26, 2019 · Twitter for iPad

M Plummer Fernandez
@M_PF

Replying to @prehensile @stephenfortune and @MIT_CSAIL

undo undo undo, phew, works again

5:52 PM · Jun 26, 2019 · Twitter Web Client

Maria Skoularidou
@skoularidou

Replying to @MIT_CSAIL

Commands are literal, your assumptions aren't.

6:03 PM · Jun 26, 2019 · Twitter Web Client

Tejas Jain
@jaintj95

Replying to @MIT_CSAIL

Pretend you know, secretly Stack Overflow.
#ProgrammingIn6Words

12:50 PM · Jun 26, 2019 · Twitter for Android

Leto Peel
@PiratePeel

Replying to @MIT_CSAIL

It almost never runs first time

5:08 AM · Jun 27, 2019 · Twitter for Android

Moshe Sipper
@moshesipper

Replying to @MIT_CSAIL

Ignoring the spec and using seven words

4:04 PM · Jun 26, 2019 · Twitter Web App

Carrie Cai
@Carryveggies

Replying to @MIT_CSAIL

A means for improving human lives.

(I'm disappointed but not surprised that so few of these 6-word phrases have mentioned humans anywhere in them.)

2:15 AM · Jun 27, 2019 · Twitter Web App

fi @thatlilfkr · 1d

it’s the beginning of the school year so i just want to remind new computer science majors that struggling to install/figure out software for classes is no indication of your programming potential! the first time i tried to install python i cried for 3 hours and i’m still here.

24 95 551

Embrace Mystery

- Selectively, temporarily
- **Really okay** for all but one line of HelloWorld.java to make **absolutely no sense** until later in the course
- Same for almost all fancy features of IntelliJ software
- Blocked? **ASK!**
- **Play**

- Introductions
- Course information
- Computation and programming
- **First programs**

```
////////////////////////////////////////////////////////////////////////////
// froggy

public class Main {
    public static void main(String[] args) {
        System.out.println("I had a little froggy.");
    }
}


////////////////////////////////////////////////////////////////////////////
// froggy in parts

public class Main {
    public static void main(String[] args) {
        System.out.print("I had");
        System.out.print(" a little");
        System.out.println(" froggy.");
    }
}


////////////////////////////////////////////////////////////////////////////
// froggy in parts in one line

public class Main {
    public static void main(String[] args) {
        System.out.println("I had" + " a little" + " froggy.");
    }
}

////////////////////////////////////////////////////////////////////////////
// froggy hopping across lines

public class Main {
    public static void main(String[] args) {
        System.out.println("My little froggy hopped \nand hopped \nand hopped.");
    }
}

////////////////////////////////////////////////////////////////////////////
// froggy hopping across lines with tabs

public class Main {
    public static void main(String[] args) {
        System.out.println("My little froggy hopped \n\tand hopped \n\t\tand hopped.");
    }
}
```

Reminders

- Read the section of the detailed schedule for Friday to know what to do to prepare for lab
- Email and/or come to office hours with questions or comments

Questions?