15-251
Great Theoretical Ideas in Computer Science

Lecture 1:
Introduction to the course

Instructors:

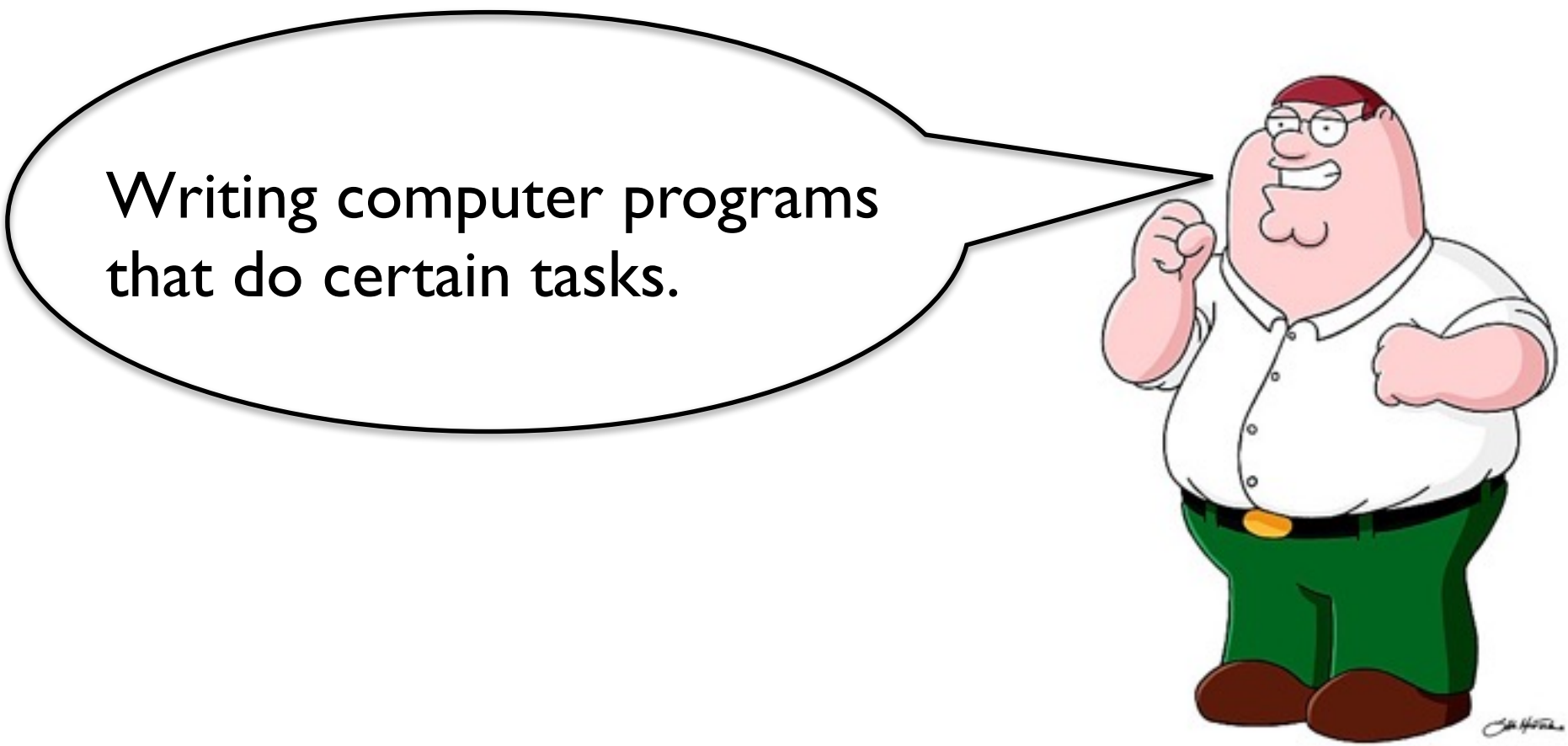
Ariel Procaccia

Anil Ada

September 1st, 2015

What is theoretical computer science?

What is computer science?



What is computer science?

Is it branch of:

- science?
- engineering?
- math?
- philosophy?
- sports?

Physics

Theoretical physics

- come up with mathematical models

  Nature's language is mathematics

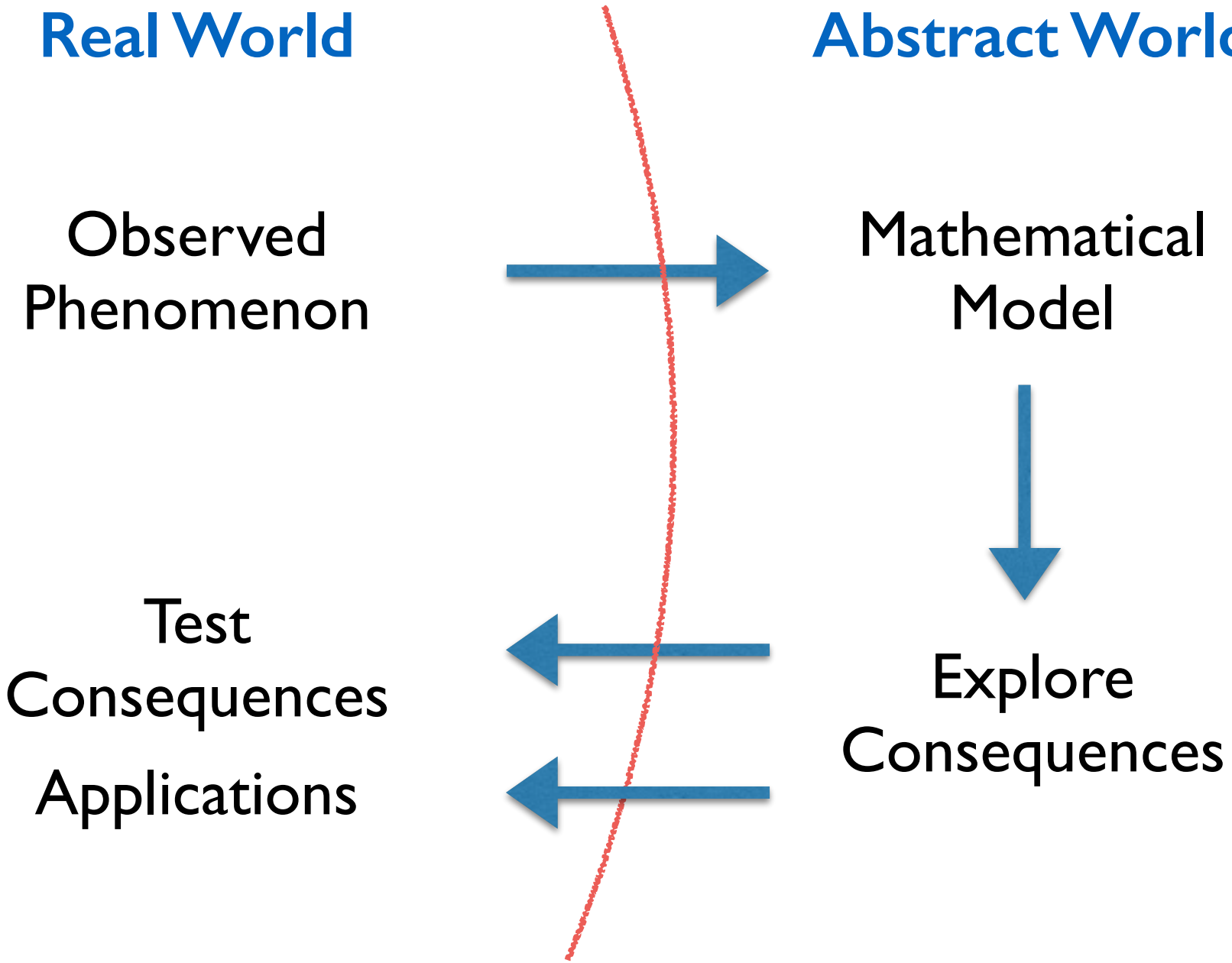
- derive the logical consequences

Experimental physics

- make observations about the universe
- test the model with experiments

Applications/Engineering

The role of theoretical physics



Theoretical Physics

- science?
- engineering?
- math?
- philosophy?
- sports?

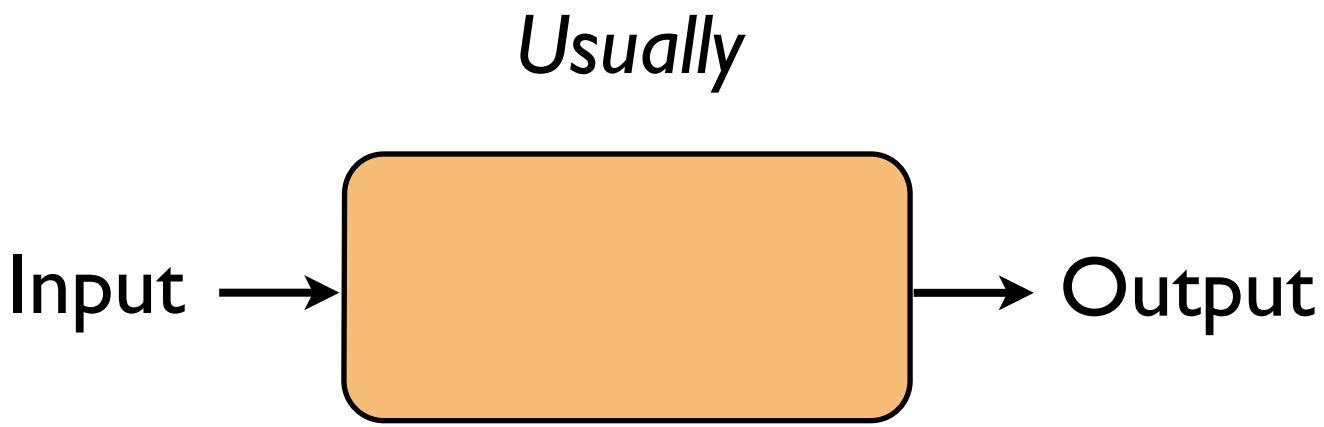
Computer Science

The science that studies computation.

Computation: manipulation of information/data.

Algorithm: description of how the data is manipulated.

Computational problem: the input-output pairs.



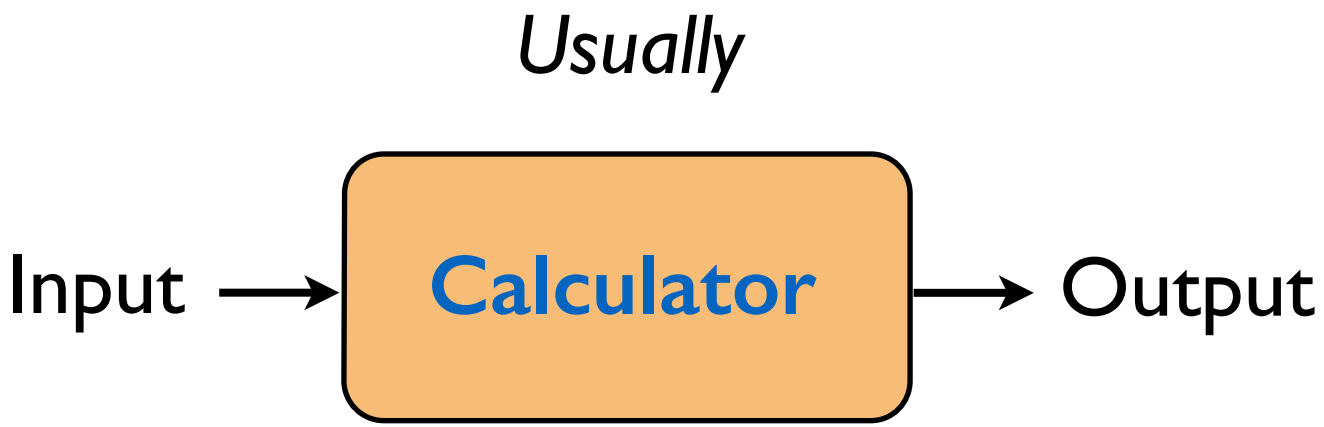
Computer Science

The science that studies computation.

Computation: manipulation of information/data.

Algorithm: description of how the data is manipulated.

Computational problem: the input-output pairs.



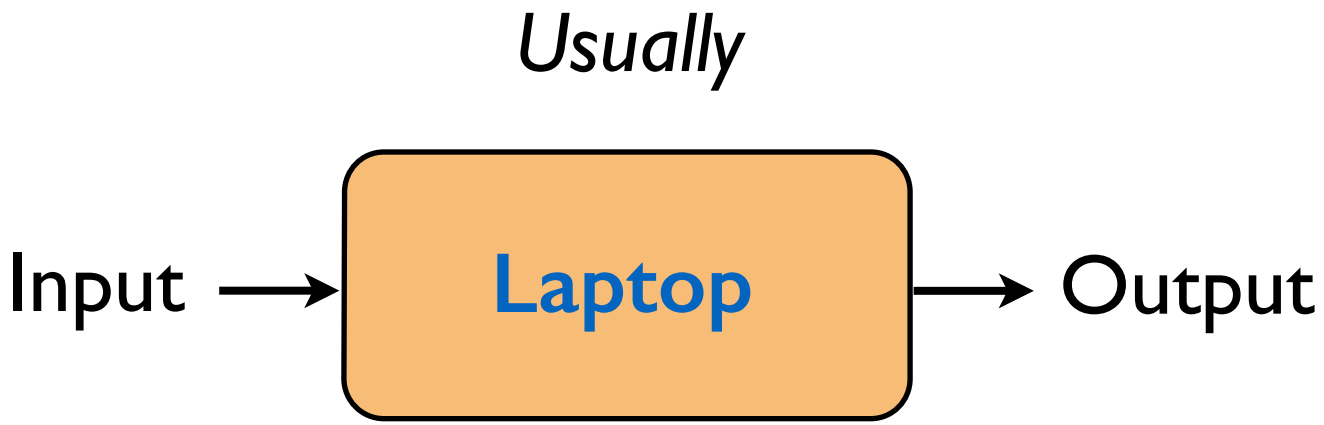
Computer Science

The science that studies computation.

Computation: manipulation of information/data.

Algorithm: description of how the data is manipulated.

Computational problem: the input-output pairs.



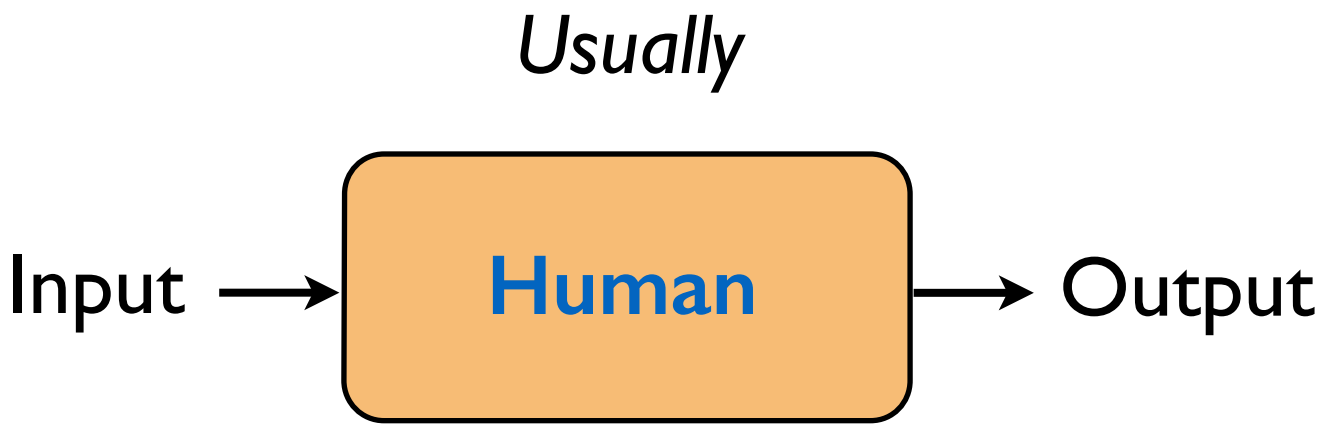
Computer Science

The science that studies computation.

Computation: manipulation of information/data.

Algorithm: description of how the data is manipulated.

Computational problem: the input-output pairs.



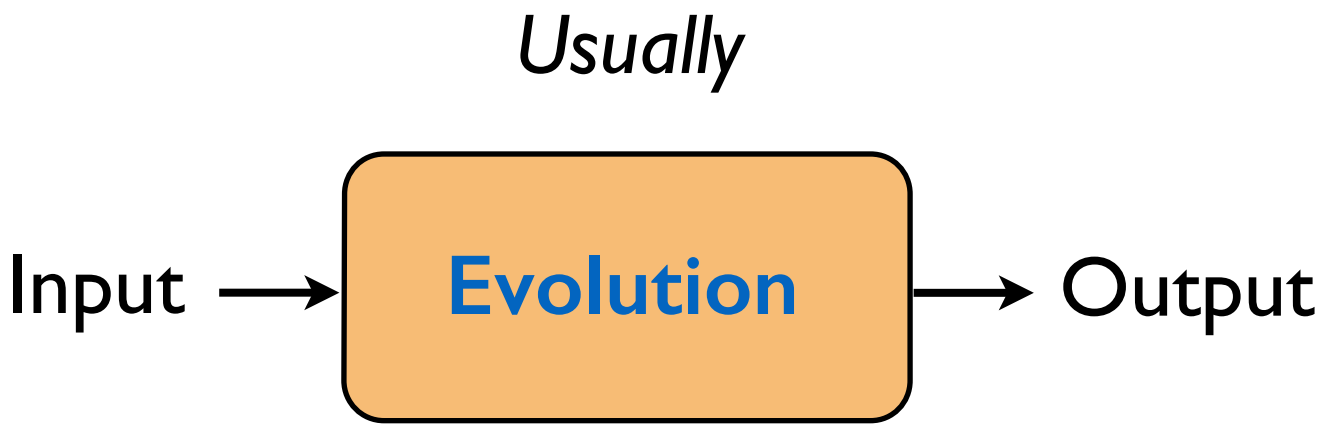
Computer Science

The science that studies computation.

Computation: manipulation of information/data.

Algorithm: description of how the data is manipulated.

Computational problem: the input-output pairs.



The computational lens

Computational physics

Computational biology

Computational chemistry

Computational neuroscience

Computational finance

…

Defining computer science

" Computer Science deals with the theoretical foundations of information and computation, together with practical techniques for the implementation and application of the foundations. "

- *Wikipedia*

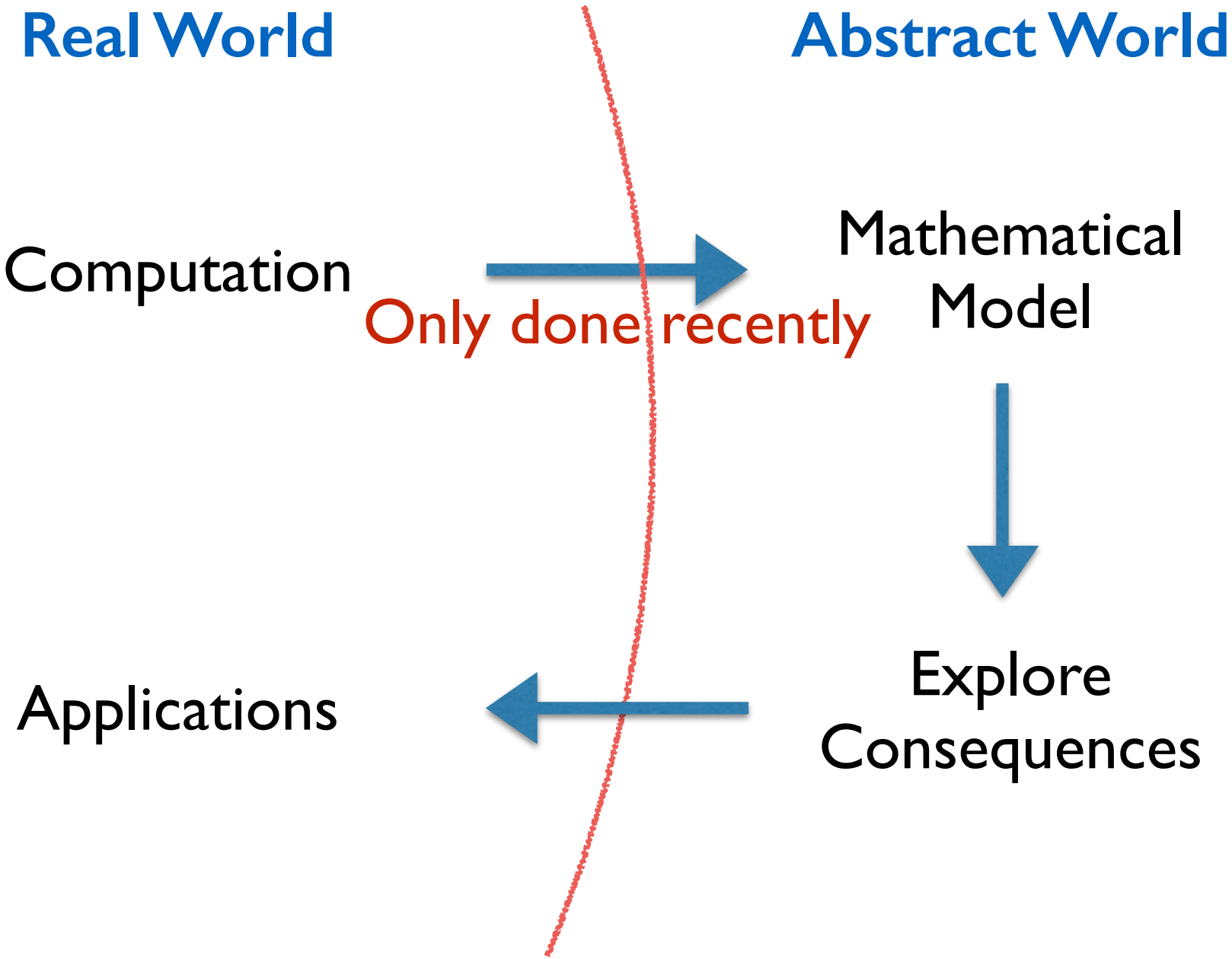
The role of theoretical computer science

Build a mathematical model for computation.

Explore the logical consequences.
Gain insight about computation.

Look for interesting applications.

The role of computer science



Simple examples of computation

$$
\begin{array}{r}
5127 \\
\times\ 4265 \\
\hline
25635 \\
307620 \\
1025400 \\
20508000 \\
\hline
21866655
\end{array}
$$

Doing computation by following a simple algorithm.

Simple examples of computation

Euclid's algorithm (~ 300BC):

```
def gcd(a, b):
    while (b != 0):
        t = b
        b = a % b
        a = t
    return a
```

We have been using algorithms for thousands of years.

Formalizing computation

We have been using algorithms for thousands of years.

Algorithm/Computation was only formalized in the 20th century!

Someone had to ask the right question.

David Hilbert, 1900

The Problems of Mathematics

"Who among us would not be happy to lift the veil behind which is hidden the future; to gaze at the coming developments of our science and at the secrets of its development in the centuries to come? What will be the ends toward which the spirit of future generations of mathematicians will tend? What methods, what new facts will the new century reveal in the vast and rich field of mathematical thought?"

Entscheidungsproblem

Hilbert's 10th problem

Is there a finitary procedure to determine if a given multivariate polynomial with integral coefficients has an integral solution?

e.g. $5x^2yz^3 + 2xy + y - 99xyz^4 = 0$

Entscheidungsproblem (1928)

Is there a finitary procedure to determine the validity of a given logical expression?

e.g. $\neg\exists x, y, z, n \in \mathbb{N} : (n \geq 3) \wedge (x^n + y^n = z^n)$

(Mechanization of mathematics)

Entscheidungsproblem

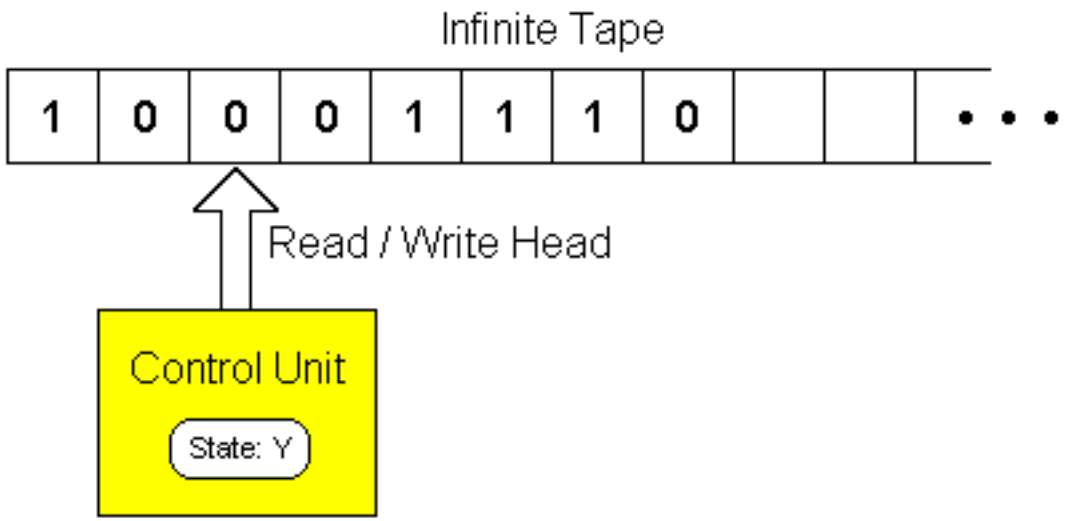
Fortunately, the answer turned out to be NO.

This question was answered by Church and Turing.

It lead Turing to define (1936) what we now call a Turing Machine.

The formalization of "computation"

Turing Machine:



→ **Universal Turing Machine**

The mathematical model for your laptop.

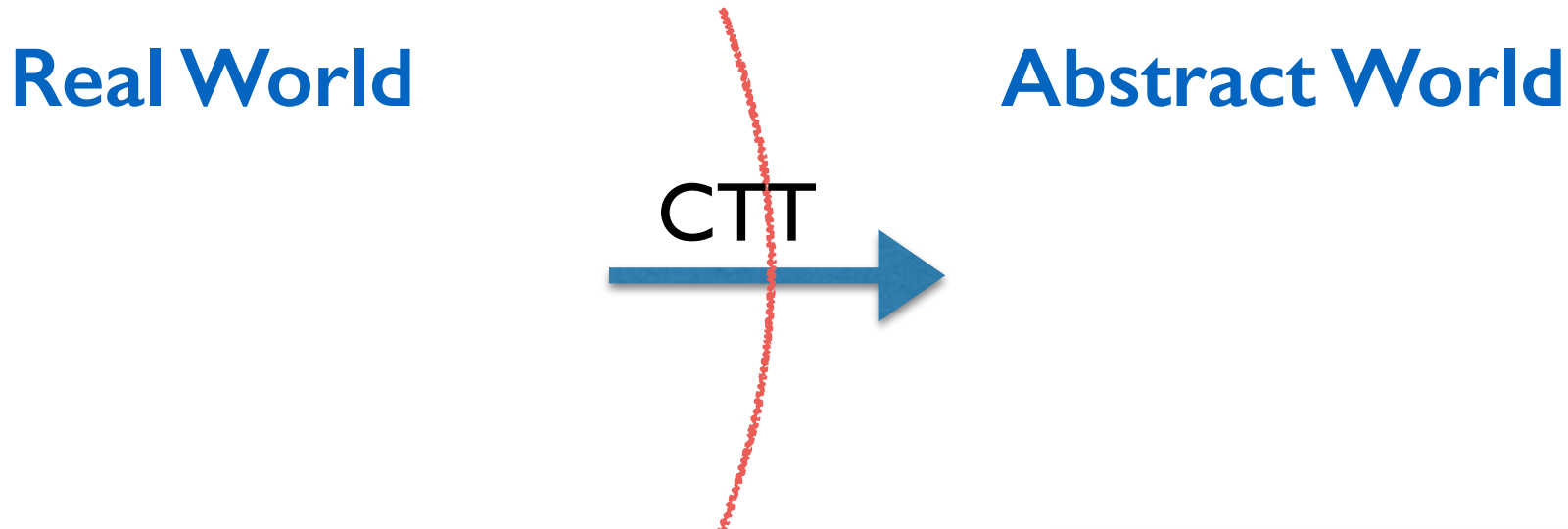
Church-Turing Thesis

Church-Turing Thesis:

The intuitive notion of "computable" is captured by functions computable by a Turing Machine.

(Physical) Church-Turing Thesis

Any computational problem that can be solved by a physical device, can be solved by a Turing Machine.



Theoretical computer science

- science?
- engineering?
- math?
- philosophy?
- sports?

2 Main Questions in TCS

Computability of a problem:

Is there an algorithm to solve it?

Complexity of a problem:

Is there an efficient algorithm to solve it?

- time
- space (memory)
- randomness
- quantum resources

Computational Complexity

Complexity of a problem:

Is there an efficient algorithm to solve it?

- time
- space (memory)
- randomness
- quantum resources

2 camps:

- trying to come up with efficient algorithms (algorithm designers)
- trying to show no efficient algorithm exists (complexity theorists)

Computational Complexity

2 camps:

- trying to come up with efficient algorithms (algorithm designers)
- trying to show no efficient algorithm exists (complexity theorists)

matrix multiplication

stock trading

protein structure prediction

simulation of quantum systems

integer factorization

Some other interesting questions

If a problem has a space-efficient solution
does it also have a time-efficient solution?

Can every randomized algorithm be derandomized efficiently?

Can we use quantum properties of matter to build faster computers?

What will you learn in this course?

Goals

1. Learn about the theoretical foundations of computation.
2. Learn the basic math we need, i.e. the language.
3. Become better problem solvers.
4. Become better at reasoning abstractly.
5. Become better at coming up with rigorous arguments.
6. Become better at paying attention to detail.
7. Become better at expressing yourself clearly.
8. Become better at working with other people.

This is a "big picture" course

This is a "big picture" course

Finite automata

Game theory

Turing machines

Cryptography

Interactive proofs

Graph theory

Learning theory

NP-completeness

Communication complexity

Quantum computation

Social choice

Markov chains

Approximation algorithms

Randomized algorithms

Online algorithms

Basic number theory

Probability

Video

A review of the course syllabus

A quick review of the course syllabus

Course webpage: www.cs.cmu.edu/~15251

A quick review of the course syllabus

Grading:

11 homework assignments, lowest score dropped

30%

2 midterm exams, lower score half weighted

10% + 20% = 30%

Oct 14, Nov 18
6:30pm-9:30pm

1 final exam

25%

~12 quizzes, lowest score dropped

10%

Participation (attending classes and recitations)

5%

A poll

What is your major?

- computer science
- mathematics
- physics
- computer/electrical engineering
- economics
- statistics
- music
- other
- beats me

Homeworks

Most important part of the course!

They are meant to be challenging.

Make use of the office hours!!!
(some questions are designed with this in mind)

Homeworks prepare you for the exams.

Homeworks

Homework System:

3 types of questions:
SOLO, GROUP, OPEN COLLABORATION

SOLO - work by yourself

GROUP - work in groups of 3 or 4

OPEN - work with anyone you would like from class

Homeworks

Homework System:

3 types of questions:
SOLO, GROUP, OPEN COLLABORATION

Don't share written material with anyone.

Erase public whiteboard when done.

Can search books to learn more about a subject.

Can't Google specific keywords from the homework.

Always cite your sources!

Think about a problem before you collaborate.

Homeworks

Homework System:

Homework writing sessions:

Wednesdays 6:30pm to 7:50pm at DH 2210

Write the solutions to a random subset of the problems.

You must practice writing the solutions beforehand!!!

20% credit reserved for presentation.

Homeworks

Homework System:

Feedback/grading:

Hopefully done by recitation.

You will know who graded which question.

Go see TA if:

- you are not happy with the points you got
- you don't understand why you lost points

Quizzes

First 9 minutes of recitation.

Just a check that you reviewed the lectures of that week.

Piazza

Everyone must sign up.

Course announcements will be made on Piazza.
You have to check it every day.

Great resource, make use of it.

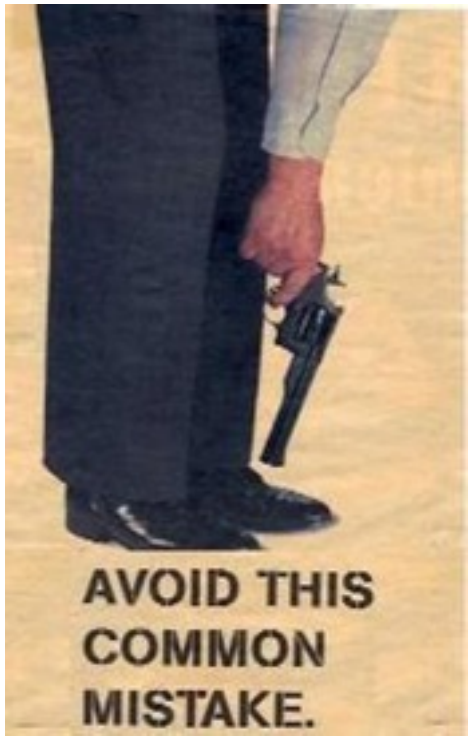
Please be polite.

Don’t give away any hints.

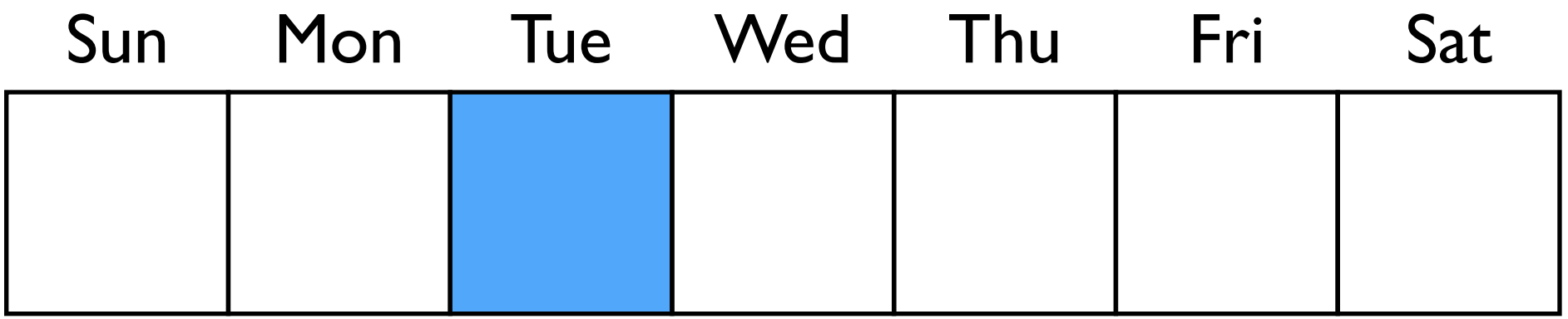
Office hours

See course webpage.

You have to use the OHs!



A typical week



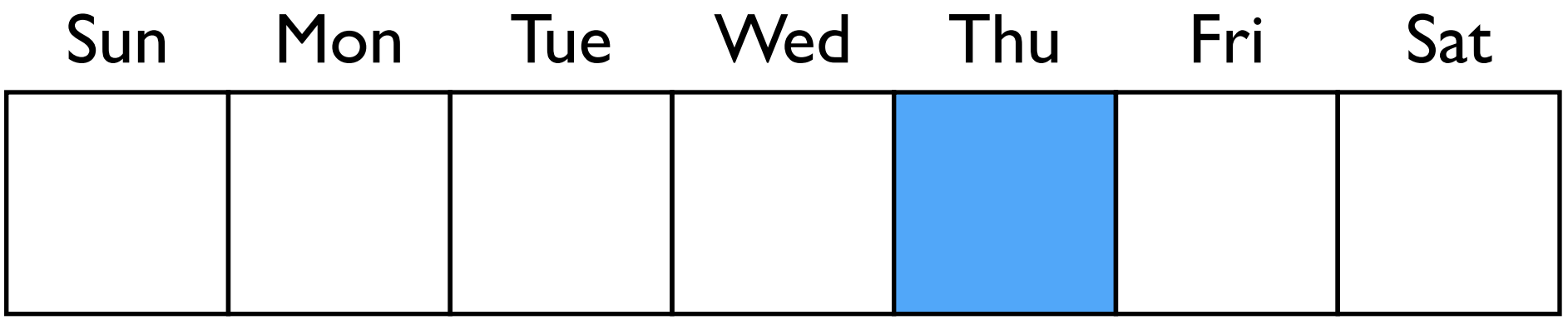
Lecture

Office hour (Anil)

A typical week

Sun	Mon	Tue	Wed	Thu	Fri	Sat

A typical week



Lecture

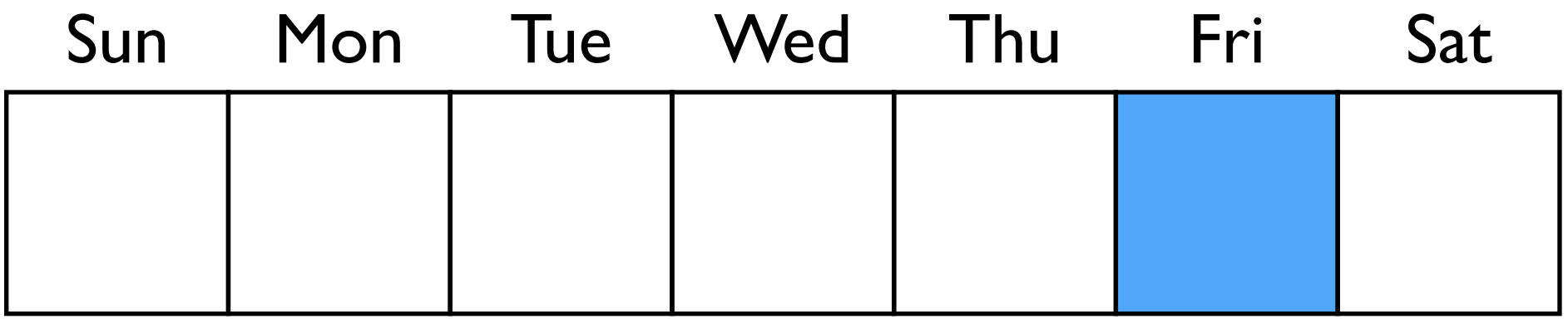
Office Hours - ask questions about lectures

Review that week's material.

Homework comes out.

Maybe start working on the SOLO problems.

A typical week



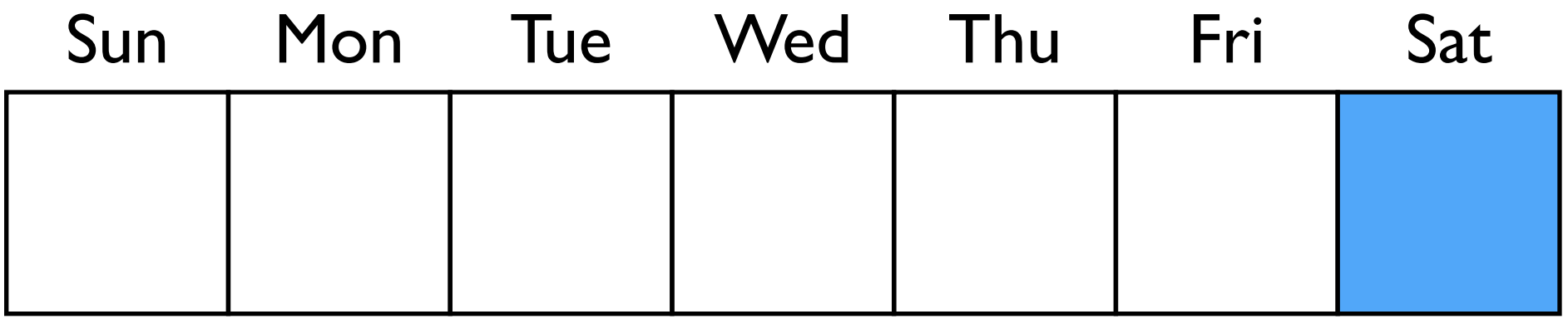
Recitation (quiz at the beginning)

Make progress on SOLO problems.

Start thinking about the GROUP problems.

Make appointments to meet with your group over the weekend.

A typical week



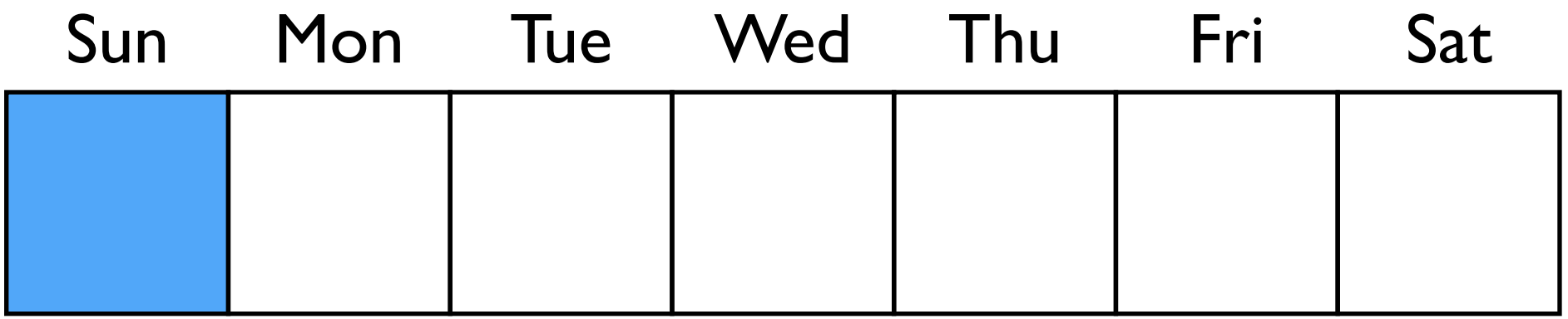
Meet with your group.

Make some progress on the questions.

Maybe solve some of them.

Go to office hours.

A typical week

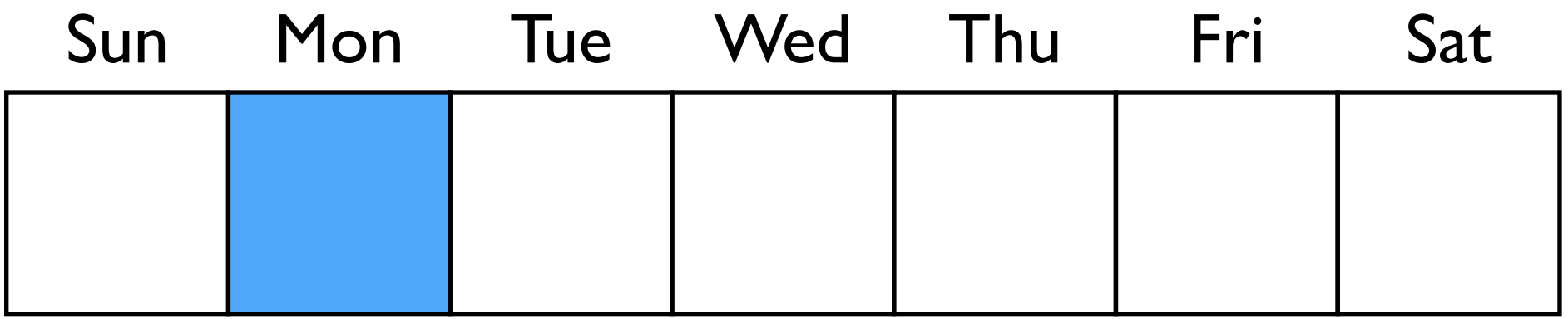


Meet with your group.

Go to office hours, get some help.

Solve some more problems.

A typical week



Finish up GROUP problems.

Go to office hours.

A typical week

Sun	Mon	Tue	Wed	Thu	Fri	Sat

Realize that you still need to do the OPEN problems!

Express hate towards the professors.

Lecture

Rush to OH to get help.

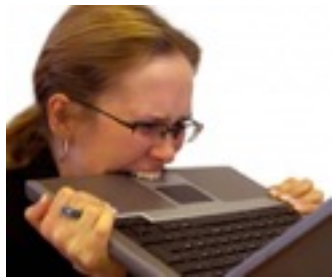
Don’t sleep until you solve the hardest problem.

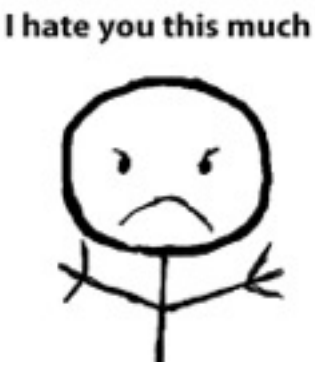
A typical week

Sun	Mon	Tue	Wed	Thu	Fri	Sat

Practice writing up the solutions to the problems.

Realize you have a mistake in one of the questions.





Express hate towards the professors.

Realize that you have to write the solution down once you think you have figured it out.

Keys to success in this course

- Take the quizzes seriously.
  (review the lectures on time)
- Use office hours. Use Piazza.
- Find good group members.
- If you are not happy with your group, break up.
- Take the "writing up the proof" part seriously.
- Make sure you understand the mistakes you make.
- Embrace the challenge.
  (if you are not challenged, complain)