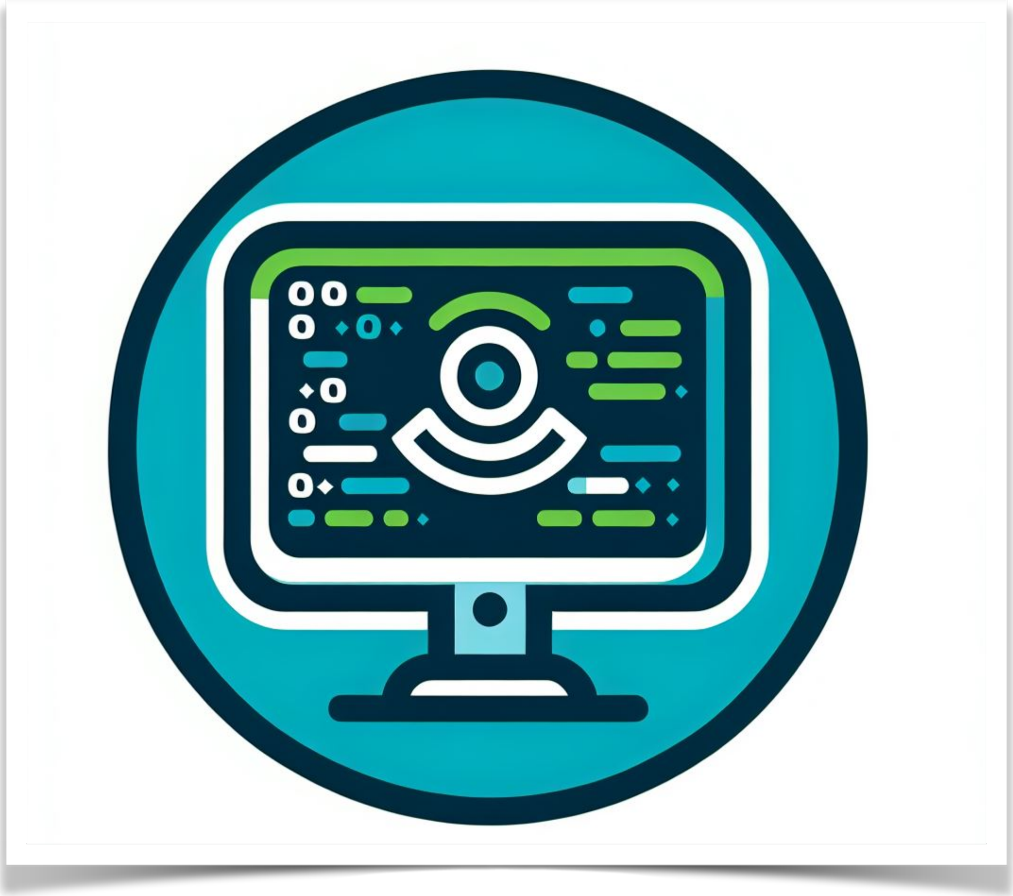


http://didawiki.di.unipi.it/doku.php/magistraleinformatica/psc/start

**PSC 2024/25** (375AA, 9CFU)

Principles for Software Composition

Roberto Bruni
http://www.di.unipi.it/~bruni/

# 01 - Introduction

# English vs Italian

# Classes

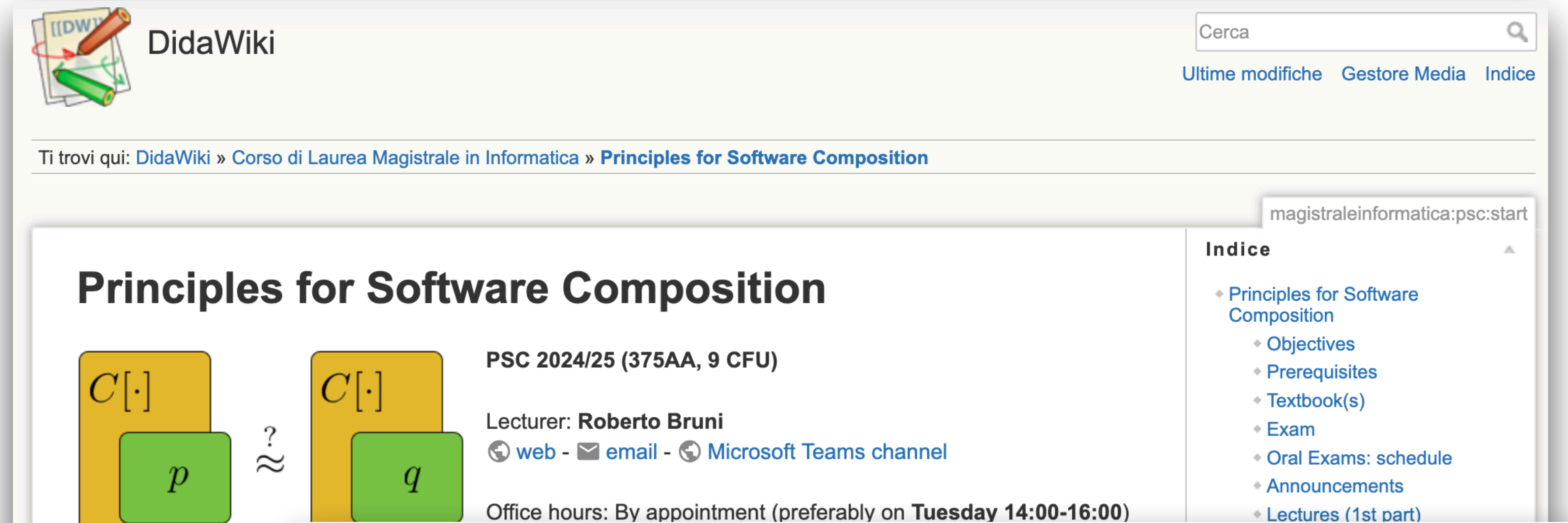
In presence, every

**Tuesday**: 16:00-18:00, A1

**Thursday**: 14:00-16:00, C1

**Friday**: 09:00-11:00, L1

# Course material



DidaWiki

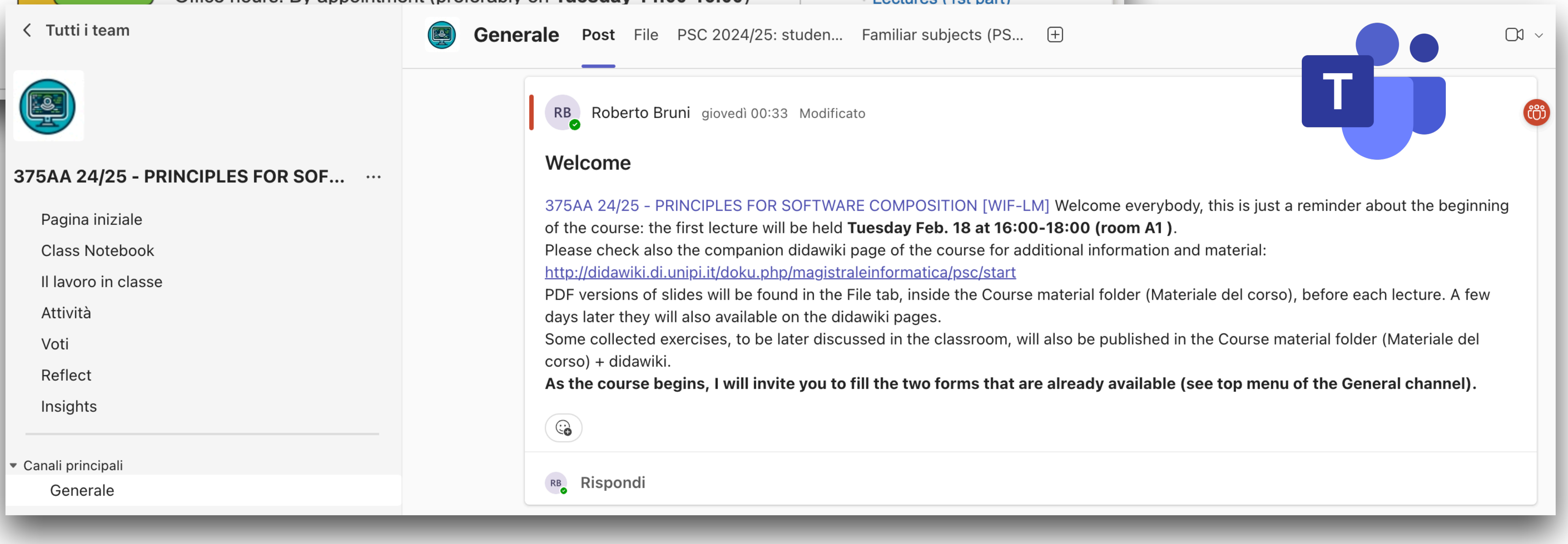
Ti trovi qui: DidaWiki » Corso di Laurea Magistrale in Informatica » **Principles for Software Composition**

## Principles for Software Composition

**PSC 2024/25 (375AA, 9 CFU)**

Lecturer: **Roberto Bruni**
web - email - Microsoft Teams channel

Office hours: By appointment (preferably on **Tuesday 14:00-16:00**)



RB Roberto Bruni giovedì 00:33 Modificato

**Welcome**

375AA 24/25 - PRINCIPLES FOR SOFTWARE COMPOSITION [WIF-LM] Welcome everybody, this is just a reminder about the beginning of the course: the first lecture will be held **Tuesday Feb. 18 at 16:00-18:00 (room A1 )**.
Please check also the companion didawiki page of the course for additional information and material:
http://didawiki.di.unipi.it/doku.php/magistraleinformatica/psc/start
PDF versions of slides will be found in the File tab, inside the Course material folder (Materiale del corso), before each lecture. A few days later they will also available on the didawiki pages.
Some collected exercises, to be later discussed in the classroom, will also be published in the Course material folder (Materiale del corso) + didawiki.
**As the course begins, I will invite you to fill the two forms that are already available (see top menu of the General channel).**

# Who am I?

http://www.di.unipi.it/~bruni

bruni@di.unipi.it

Office hours: by appointment
preferably
Tuesday 14:00-16:00

# Research topics (theses?)

False alarm detection in Abstract Interpretation

Formal approaches to code obfuscation

Quantum Computation and concurrency models

Modelling and analysis of biological systems

Graphical specification languages

Algebraic approaches to structured graphs

Rewrite rules for reversible languages

# Who are you?

First Name:

Last Name:

Enrollment number:

email:

Bachelor degree:

MSc course of enrollment:

Please fill the form!

Forms

# Who are you?

First Name: John

Last Name: Smith

Enrollment number: 123456

email: john.smith@email.com

Bachelor degree: Comp. Sci., Pisa, IT

MSc course of enrollment: Comp. Sci. - SW

Please fill the form!

Forms

# Interaction protocol

when I will ask questions such as:

“does my program $c$ satisfy the property $\psi$?”

there are the 3 possible answers

○ yes
○ no
○ don’t know

you are welcome to take your time... but then
you MUST select one of them to answer questions in these classes

# The Course

# Some quotes

*Computer science is no more about computers than astronomy is about telescopes*
- Edsger W. Dijkstra

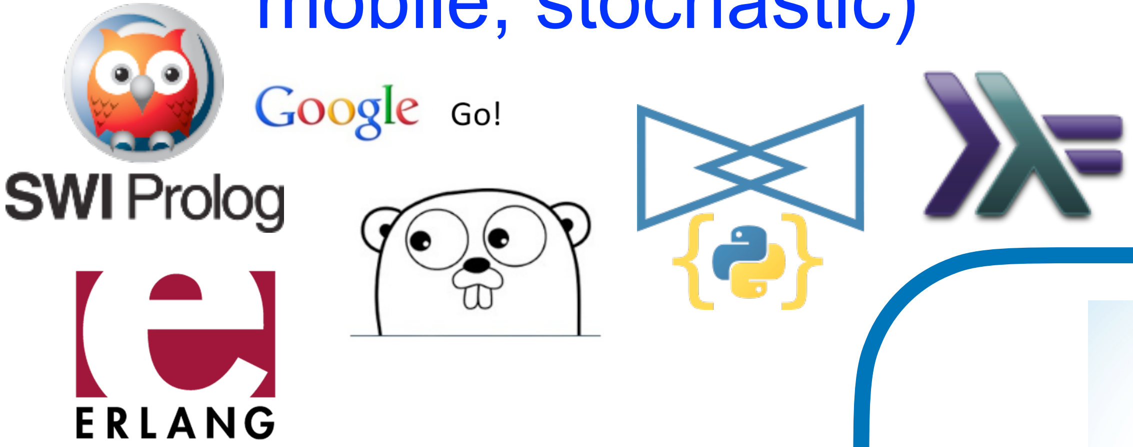
*Studying programming languages without formal semantics would be like studying physics without math*
- from the web

*All models are wrong, but some are useful*
- George Box

*Subjects are divided in two categories:*
*1) too difficult matters, that CANNOT be studied*
*2) easy matters, that DO NOT NEED to be studied*
- back of a t-shirt

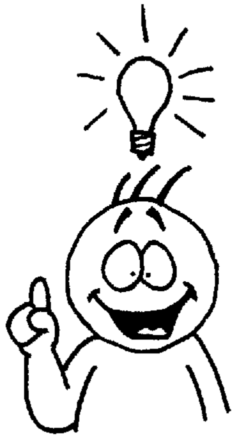
# Objectives

Programming paradigms
(imperative, declarative, higher order, concurrent, mobile, stochastic)



Mathematical frameworks (concrete & abstract)
(domains, inference rules, transition systems, λ-calculus, process algebras)

Understand
(recursion, semantics, compositionality)



Reason
(induction, modal and temporal logics, behavioural and logical equivalences)

Explain
(correctness, compliance, performance)

# The approach

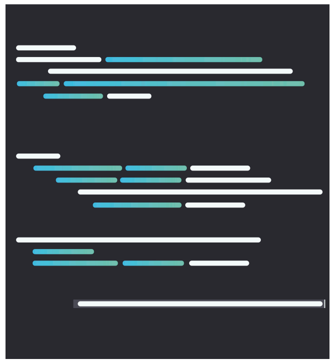
(in their simplest form,
still Turing equivalent)

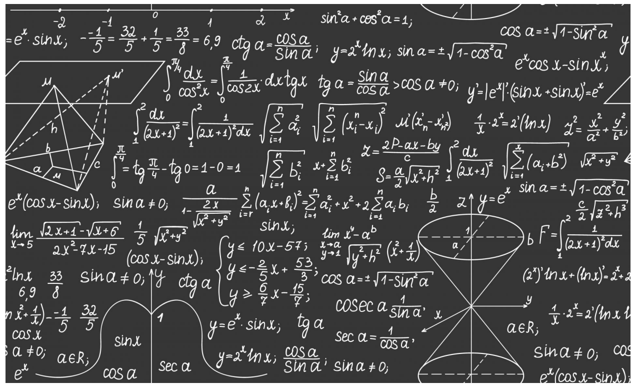
programming paradigms

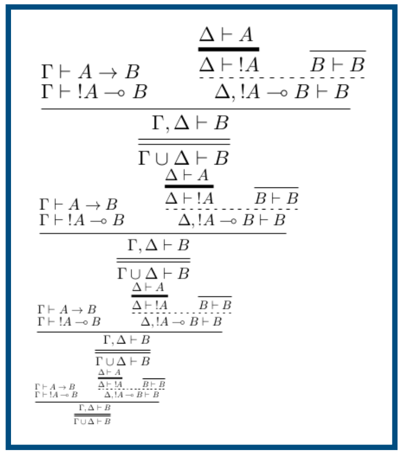
mathematical frameworks

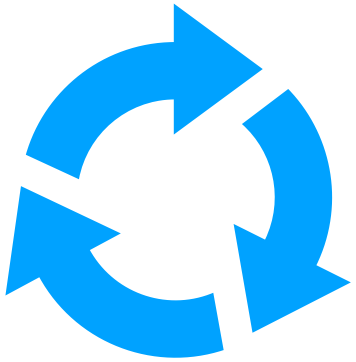
meta-properties
+
proof techniques

(for all programs
or just
some classes of programs)









models

programs

specifications

# Key question

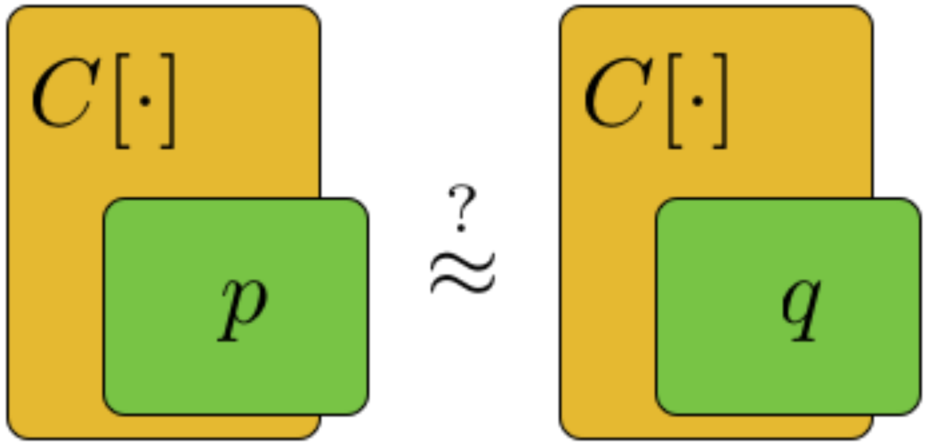
Given two programs *p* and *q*:

Do they behave the same?

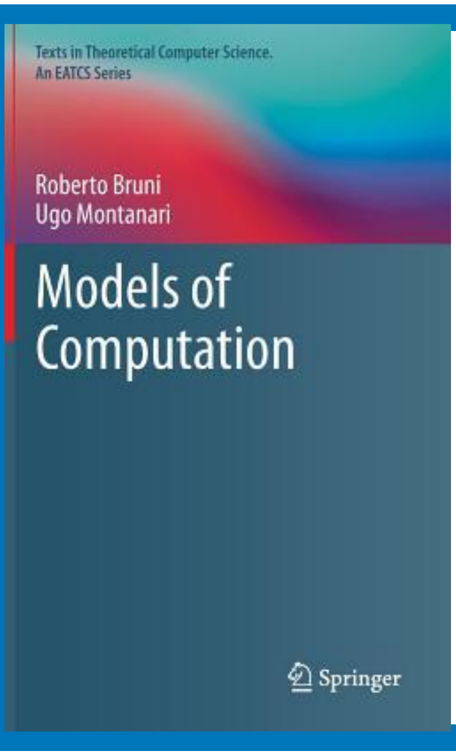
are they "equivalent"?

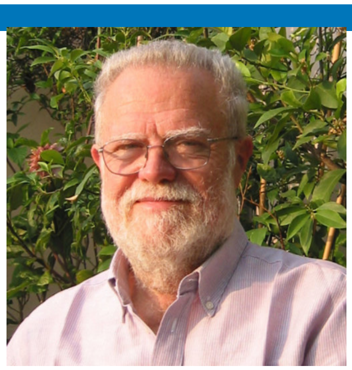
Is it safe to replace one with the other in any context?

are they "congruent"?

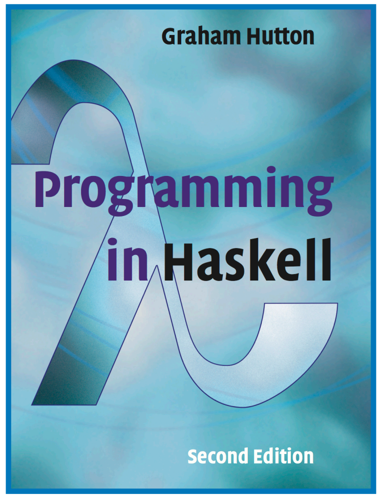


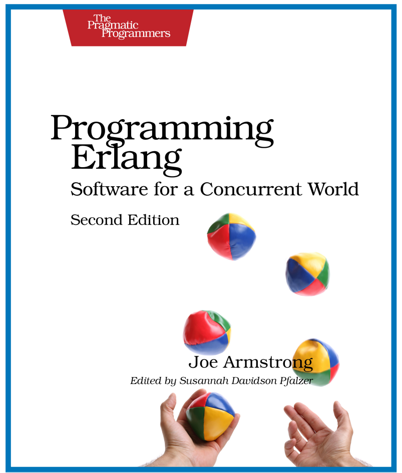
# Textbooks

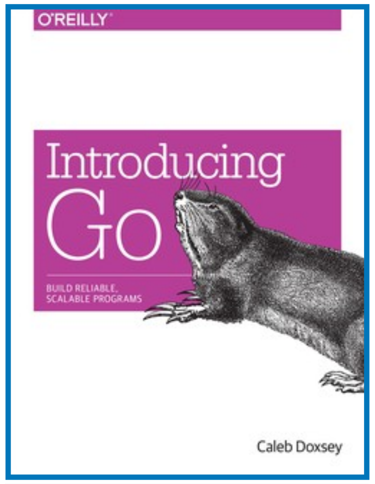


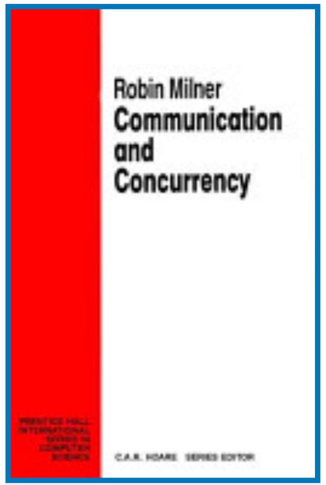


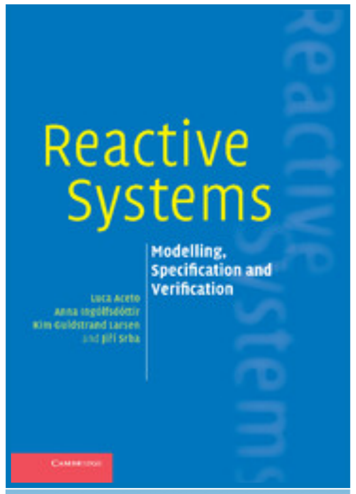
Roberto Bruni and Ugo Montanari
Models of Computation
Texts in Theoretical Computer Science (an EATCS series)
https://www.springer.com/book/9783319428987

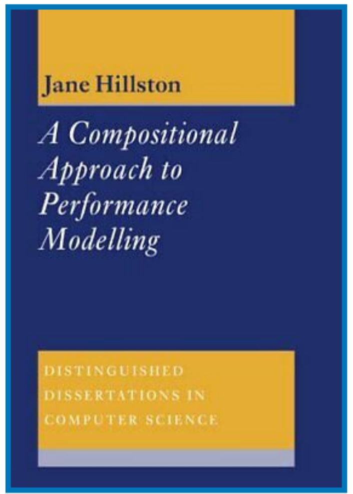



The Pragmatic Programmers
Programming Erlang
Software for a Concurrent World
Second Edition
Joe Armstrong
Edited by Susannah Davidson Pfalzer




Robin Milner
Communication and Concurrency
PRENTICE HALL INTERNATIONAL SERIES IN COMPUTER SCIENCE
C.A.R. HOARE SERIES EDITOR

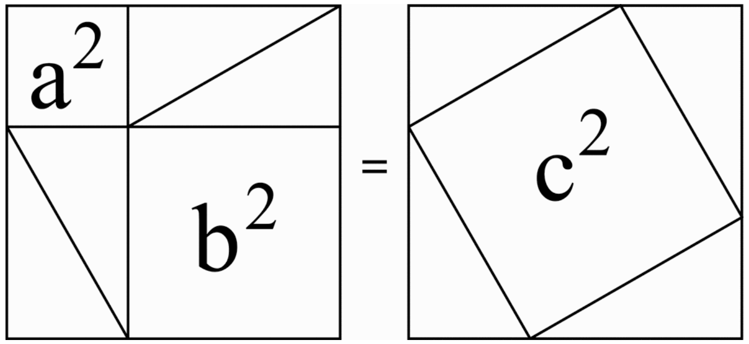



Jane Hillston
A Compositional Approach to Performance Modelling
DISTINGUISHED DISSERTATIONS IN COMPUTER SCIENCE

# Course activities

attend classrooms:
ask questions!
(sleep quietly)

learn theorems:
(drink many coffees)



do some thinking:
solve ALL your
homeworks
(at least try to)

give the exam:
time for a party!

# Be proactive!

Let’s spell out definitions together

```
% find the least (non-unitary) divisor p of n>1
% example: for n=21 the result is p=3
p := 0;
x := 2;
while (..........................................................) do {
    if ( n%x == 0 ) then {
        p := x;
    } else {
        x := x + 1;
    }
}
```

# Be proactive!

Correct me if I’m wrong

```
% find the index i of the last occurrence of n in a
% example: for n=5, a=[3,5,7,5,9] the result is i=3
i := length(a)-1;
while ( i>0 && n!=a[i] ) do {
    i := i-1;
}
```

# Be proactive!

Sometimes tricky questions!

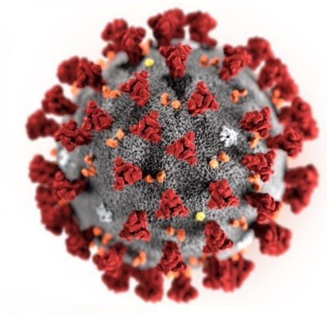
**Can you find the the mistake?**

1 2 3 4 5 6 7 8 9

# Exam

In past years, the evaluation was based on written and oral exams.

Since the covid-19 emergency, and for the current period, the evaluation will be solely based on a final oral exam.

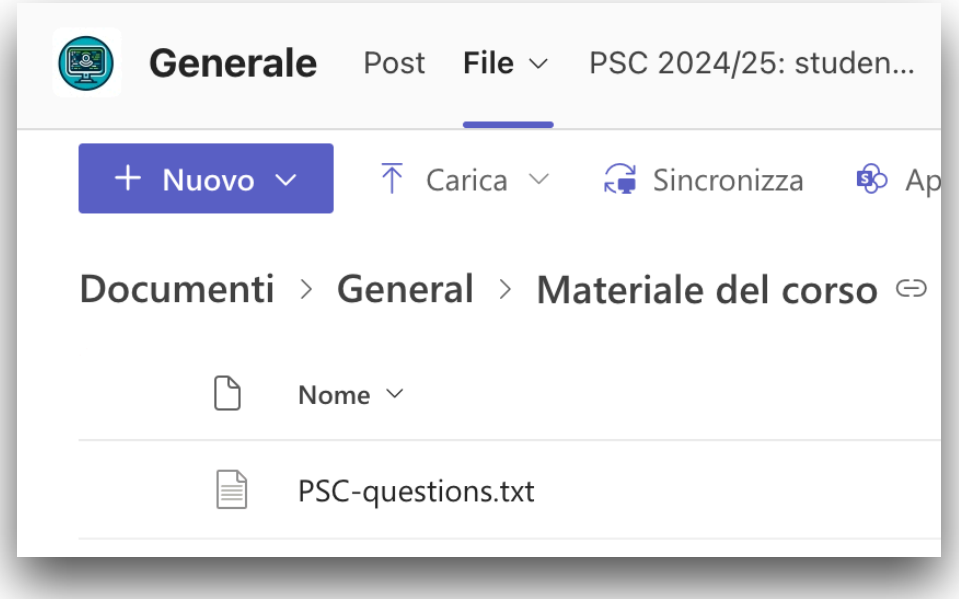


Registration to exams is mandatory:
https://esami.unipi.it/esami

The exam will typically consist of:

1. three to four preliminary questions
2. one exercise (analogous to past written exams)
3. redoing one of the proofs seen in the course
4. some additional questions

The list of preliminary questions is available on Microsoft Teams, in the File tab (`PSC-questions.txt`)



# A sample exam

What is a complete partial order?

What are the rules of the type system of HOFL?

How is iteration achieved in CCS?

Why only positive normal forms are considered in the mu-calculus?

Consider the HOFL term

$$t \overset{\text{def}}{=} \mathbf{rec}\ f.\ \lambda x.\ \mathbf{if}\ x\ \mathbf{then}\ (x, \mathbf{fst}(f\ x))\ \mathbf{else}\ (\mathbf{snd}(f\ x), x)$$

1. Find the principal type of $t$.
2. Find the denotational semantics of $t$.

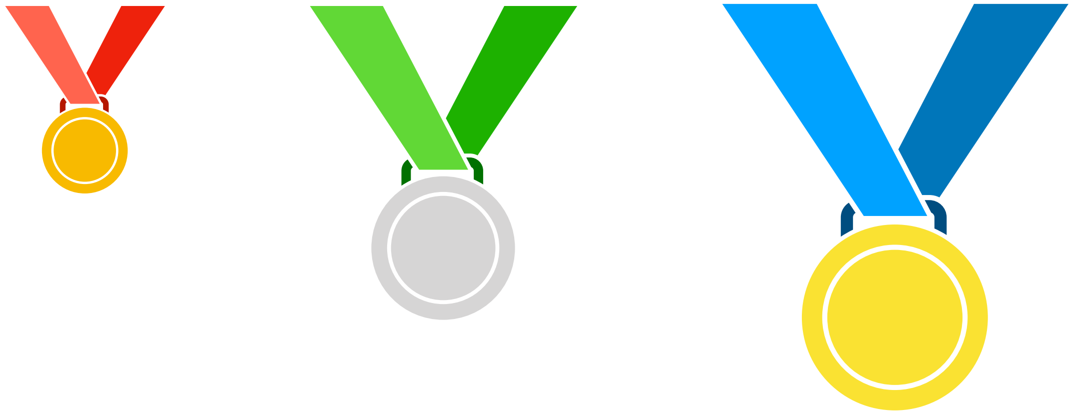
Prove the Switch Lemma

Given the initial state distribution and a DTMC,
how do we compute the state distribution at time 3?

# Badges?

No mid-terms<br>
No self-evaluation tests<br>
During the course: some “badge” exercises

Submit your solutions by email to earn<br>
bronze / silver / gold  badges<br>
(no extra scores, but be proud of yourselves)



# Prerequisites

## Basic set theory

$\varnothing$ $\quad A \cap B$ $\quad A \cup B$ $\quad A \setminus B$ $\quad \overline{A}$

$a \in A$ $\quad A \subset B$ $\quad A \subseteq B$ $\quad A \times B$

$a \notin A$ $\quad A \not\subset B$ $\quad A \cap B = \emptyset$

$\mathbb{N}$ $\quad \mathbb{Z}$ $\quad \mathbb{Q}$ $\quad \mathbb{R}$ $\quad \mathbb{B}$

$N \subseteq \mathbb{N}$ $\quad N \in \wp(\mathbb{N})$ $\quad S \subseteq \wp(\mathbb{N})$

# Prerequisites

Basic set theory: functions, relations

$$f : A \to B$$

$$R \subseteq A \times B$$

functions as relations

$$R_f \triangleq \{(a, f(a)) \mid a \in A\}$$

sets as functions
(characteristic function)

$$f_N : \mathbb{N} \to \mathbb{B}$$

$$f_N(n) \triangleq \begin{cases} 1 & n \in N \\ 0 & \text{otherwise} \end{cases}$$

$$N = \{n \mid f_N(n) = 1\}$$

# Prerequisites

## First order logic

ff false $0$ F tt true $1$ T $P \wedge Q$ $P \vee Q$ $\neg P$

$\exists x.\ P(x)$ $\forall x.\ P(x)$ $P \Rightarrow Q$ $P \Leftrightarrow Q$

meaning of implication!

$$P \Rightarrow Q$$

$$Q \vee \neg P$$

$$\neg Q \Rightarrow \neg P$$

order of quantifiers matters!

$$\forall n \in \mathbb{N}.\ \exists m \in \mathbb{N}.\ n < m$$

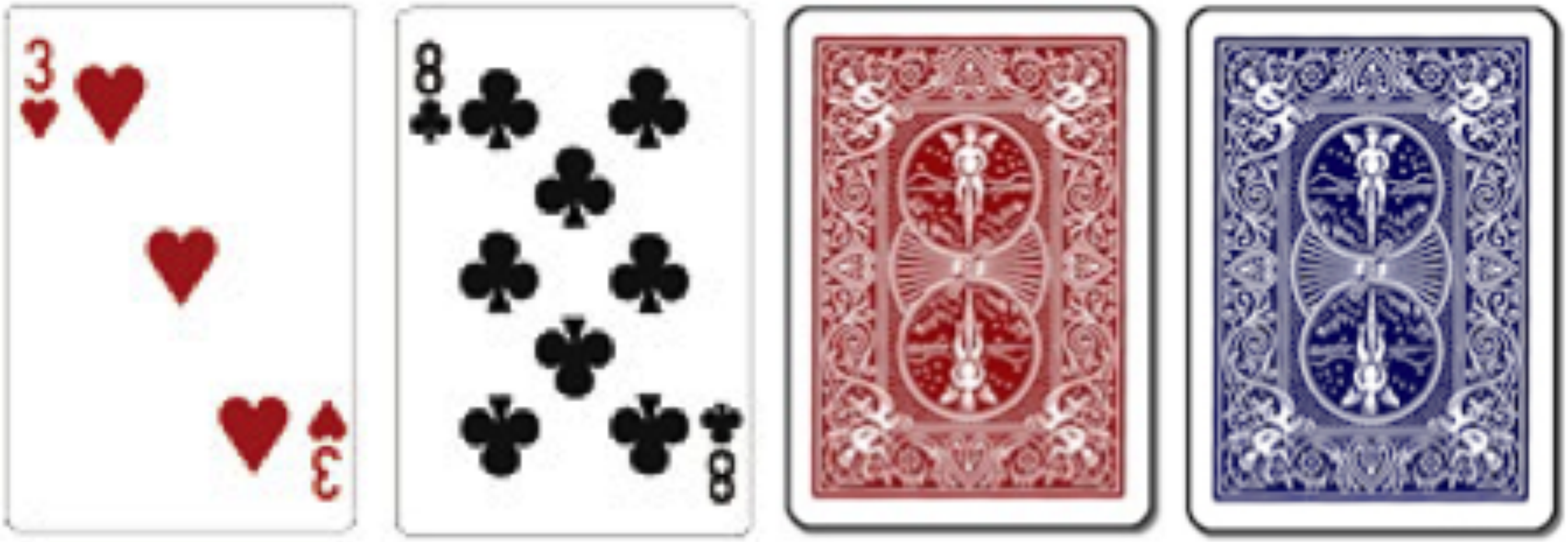
$$\exists m \in \mathbb{N}.\ \forall n \in \mathbb{N}.\ n < m$$

# Implication is a tricky concept

under which circumstances is the following promise broken?

“*If Rob Bery wins the election, I promise to leave the country*”

# (Peter Cathart) Wason Selection Task (4 cards game)



**Which cards must be turned over** to make sure the following statement is true?

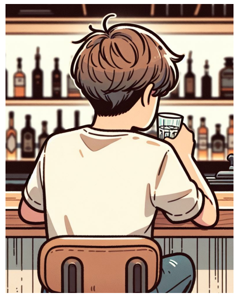
**“If the front face of a card bears an even number, then its back face is red.”**

# (Peter Cathart) Wason Selection Task (with beers)

A policeman enters a local in Florida where a large sign reminds customers that:

**“to drink beer you must be over 16 years old”**

There are four customers in the local:



1) a boy who is drinking water,

2) a girl who is drinking beer,

3) an elderly lady who is drinking at a table, and

4) a 15-year-old teenager who is drinking at the counter.

**Which customers should the policeman check** to verify that the rule is respected?

# Prerequisites

Strings and context-free grammars

Alphabet $A$ $\qquad A^n \triangleq \underbrace{A \times \cdots \times A}_{n} \qquad A^* \triangleq \bigcup_{n \in \mathbb{N}} A^n$

$$\begin{aligned}
\mathbb{B} &= \{0, 1\} \\
\mathbb{B}^0 &= \{\epsilon\} \\
\mathbb{B}^1 &= \{0, 1\} \\
\mathbb{B}^2 &= \{00, 01, 10, 11\} \\
\mathbb{B}^3 &= \{000, 001, 010, 011, 100, 101, 110, 111\} \\
&\ldots \\
\mathbb{B}^* &= \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}
\end{aligned}$$

# Prerequisites

## Strings and context-free grammars

Alphabet $A$ $\quad A^n \triangleq \underbrace{A \times \cdots \times A}_{n}$ $\quad A^* \triangleq \bigcup_{n \in \mathbb{N}} A^n$

$$\mathbb{B}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$$

$$\mathsf{A} ::= \epsilon \mid 0\ \mathsf{A} \mid 1\ \mathsf{B}$$

$$\mathsf{B} ::= 0\ \mathsf{B} \mid 1\ \mathsf{A}$$

$$\mathsf{A} \to 0\ \mathsf{A} \to 0\ 1\ \mathsf{B} \to 0\ 1\ 1\ \mathsf{A} \to 0\ 1\ 1\ \epsilon = 0\ 1\ 1$$

$\mathcal{L}(\mathsf{A}) = ?$ $\qquad \mathcal{L}(\mathsf{B}) = ?$

# Prerequisites

Inductive and recursive definitions

$$\begin{aligned} 0! &\triangleq 1 \\ (n+1)! &\triangleq n! \cdot (n+1) \end{aligned}$$

$$\begin{aligned} A^0 &\triangleq \{\epsilon\} \\ A^{(n+1)} &\triangleq A \times A^n \end{aligned}$$

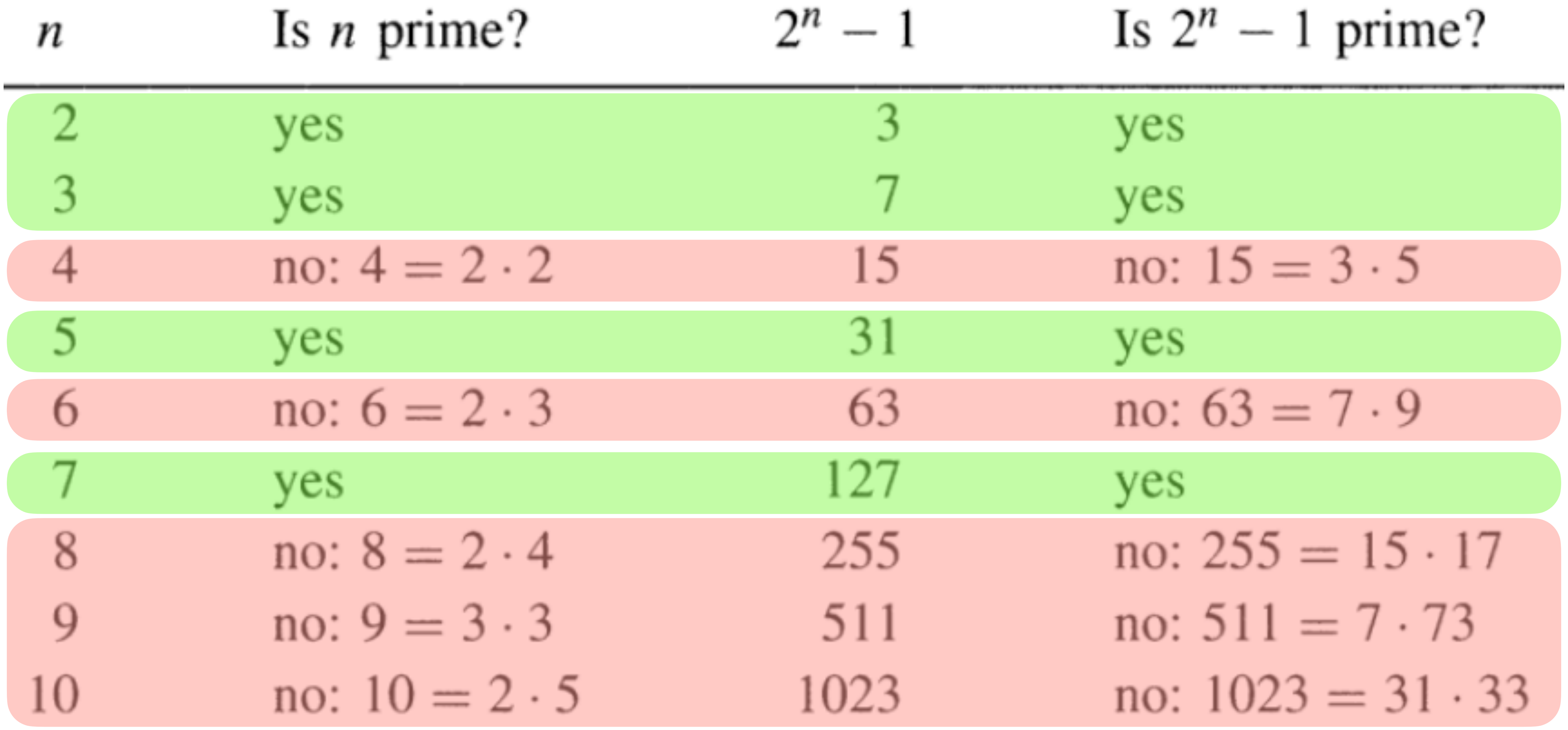
$$f(n) \triangleq \begin{cases} 1 & \text{if } n \leq 1 \\ f(n/2) & \text{if } n > 1 \wedge n\%2 = 0 \\ f(3n+1) & \text{otherwise} \end{cases}$$

$$f(12) = f(6) = f(3) = f(10) = f(5) = f(16) = f(8) = f(4) = f(2) = f(1) = 1$$

# Prerequisites

Conjectures vs theorems

a natural number ***p*** is **prime**
if it cannot be written as the product of two smaller numbers



| $n$ | Is $n$ prime? | $2^n - 1$ | Is $2^n - 1$ prime? |
|---|---|---|---|
| 2 | yes | 3 | yes |
| 3 | yes | 7 | yes |
| 4 | no: $4 = 2 \cdot 2$ | 15 | no: $15 = 3 \cdot 5$ |
| 5 | yes | 31 | yes |
| 6 | no: $6 = 2 \cdot 3$ | 63 | no: $63 = 7 \cdot 9$ |
| 7 | yes | 127 | yes |
| 8 | no: $8 = 2 \cdot 4$ | 255 | no: $255 = 15 \cdot 17$ |
| 9 | no: $9 = 3 \cdot 3$ | 511 | no: $511 = 7 \cdot 73$ |
| 10 | no: $10 = 2 \cdot 5$ | 1023 | no: $1023 = 31 \cdot 33$ |

# Prerequisites

## Conjectures vs theorems

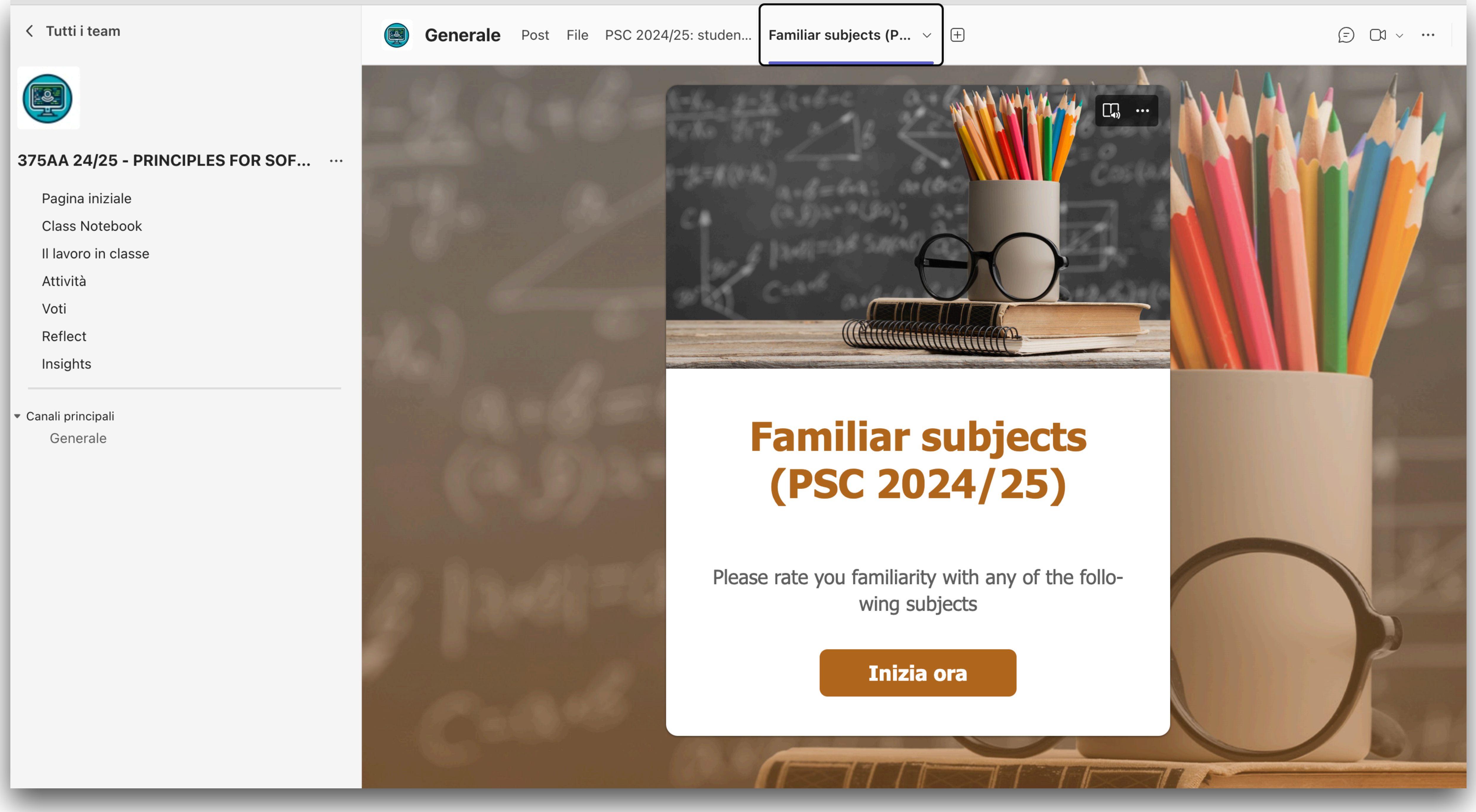
if $p$ is prime
then $2^p - 1$ is prime

if $n > 1$ is not prime
then $2^n - 1$ is not prime

Use any mean to prove or disprove the above conjectures

# Your background?

Please fill the form about “Familiar subjects”



< Tutti i team

375AA 24/25 - PRINCIPLES FOR SOF...

Pagina iniziale

Class Notebook

Il lavoro in classe

Attività

Voti

Reflect

Insights

Canali principali

Generale

Generale · Post · File · PSC 2024/25: studen... · Familiar subjects (P...

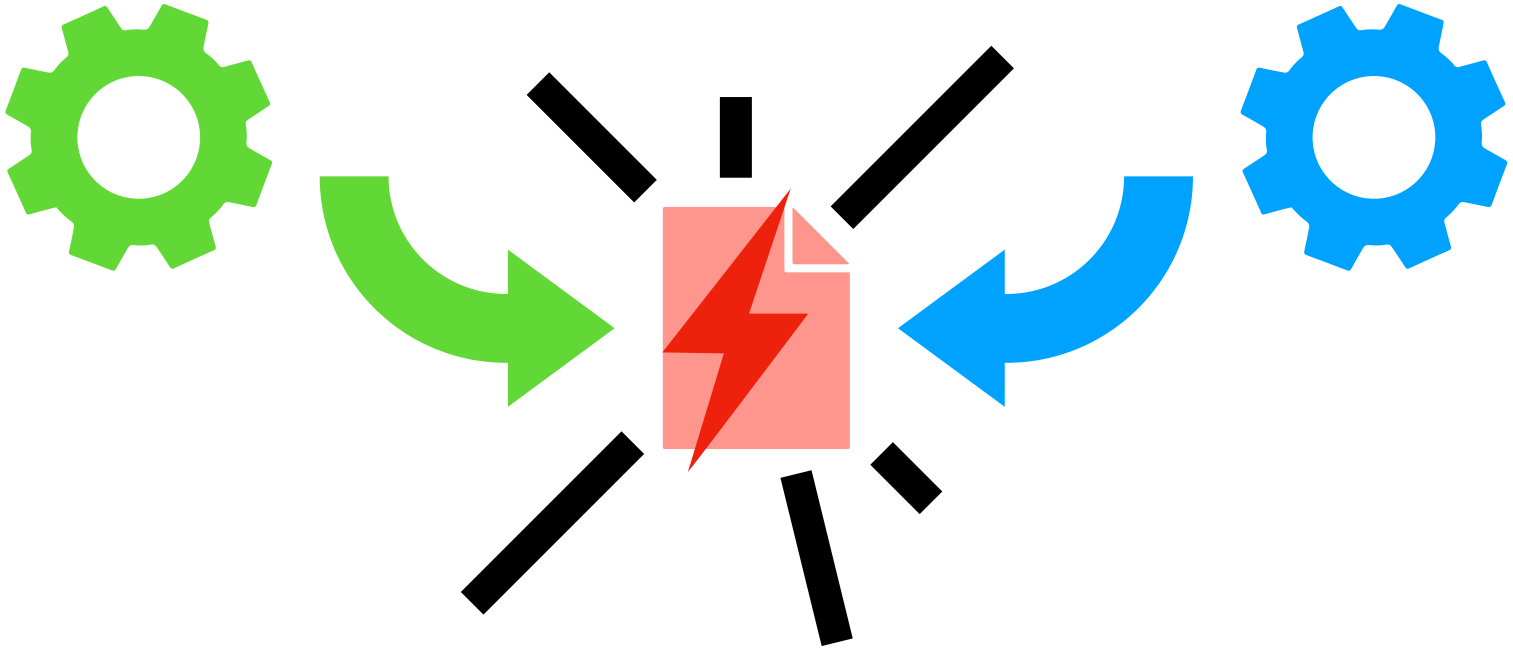
## Familiar subjects (PSC 2024/25)

Please rate you familiarity with any of the following subjects

Inizia ora

# An Appetiser

# The problem



Two concurrent processes share a single-use resource

They can communicate using shared memory

We want to guarantee that there are no conflicts when the processes access the resource

No strict alternation of naive turn taking is imposed

# Peterson’s mutual exclusion algorithm (1981)

```
% Two processes P1, P2
% Two boolean variables b1, b2 (both initially false)
% when Pi wants to enter the critical section, then it sets bi to true
% An integer variable k, taking values in {1,2}
% (initial value is arbitrary)
% the process Pk has priority over the other process
%
% Process P1 in pseudocode
while (true) {
    ...                            % non critical section
    b1 := true ;                   % P1 wants to enter the critical section
    k := 2 ;                       % P1 gives priority to the other process
    while (b2 && k==2) skip ;      % P1 waits its turn
    ...                            % P1 enters the critical section
    b1 := false                    % P1 leaves the critical section
}

% Process P2 is analogous to P1
```

# Which question?

Does Peterson’s algorithm work?

What does it mean that “it works”? What do we expect?

**(Progress)**
If the resource is available, no process is forced to wait

**(Bounded Waiting)**
No process will wait forever for the resource
*(otherwise the easiest solution is no one gets in)*

**(Mutual Exclusion)**
P1 and P2 are never in the critical section at the same time

# Hyman’s mutual exclusion algorithm (1966)

```
% Two processes H1, H2
% Two boolean variables b1, b2 (both initially false)
% when Hi wants to enter the critical section, then it sets bi to true
% An integer variable k, taking values in {1,2}
% (initial value is arbitrary)
% the process Hk has priority over the other process
%
% Process H1 in pseudocode
while (true) {
    ...                       % non critical section
    b1 := true ;              % H1 wants to enter the critical section
    while (k==2) {            % while H2 has priority
        while (b2) skip ;     % H1 waits
        k := 1;               % H1 sets priority to itself
    }
    ...                       % H1 enters the critical section
    b1 := false               % H1 leaves the critical section
}

% Process H2 is analogous to H1
```

# The question

Does Peterson’s algorithm satisfy mutual exclusion?

Does Hyman’s algorithm satisfy mutual exclusion?

For the answers be patient and wait early-May lectures