

TDT4102 – C++ OOP

Lecture 1: Information Compilation

Bart Iver van Blokland

Before we begin..

- My apologies for any language errors
- Slides are in English
- You can send feedback on the normal lectures here:
~ bart.van.blokland@ntnu.no

Today..

1. Answer questions about the course:
 - **Who are these people?**

Bart Iver van Blokland

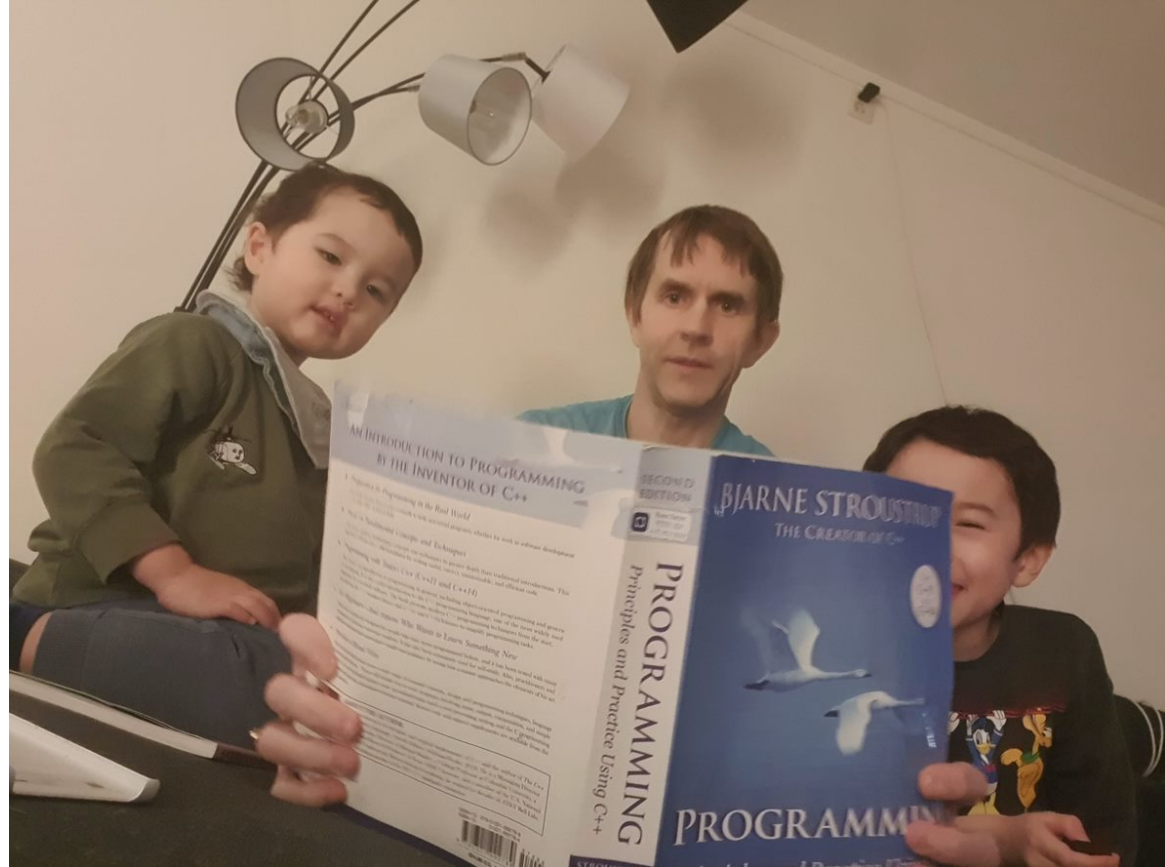
- Assistant Professor at the Department of Computer Science (IDI)
- Interests:
 - Computer Graphics
 - GPU Programming
 - 3D Object Recognition
 - High Performance Computing
- Free time:
 - Kayaking
 - Hiking
 - Video and boardgames

<https://github.com/bartvbl>



Rune Sætre

- Associate Professor at the Department of Computer Science (IDI)
- Leader of Tekna NTNU and HTV Akademikerne NTNU
- Interests:
 - Artificial Intelligence
 - Natural Language Processing
 - Software Engineering
 - Health IT
- Free time:
 - Climbing
 - Skydiving
 - Biking
 - Swimming



Anita Bueno Lindmoen



- Vit.Ass. at the Department of Computer Science (IDI)
- Master Student in Electrical Systems
- Black Belt in Canvas Fu
- Elite operative in the Student Emergency Special Response Unit
- Guru of the C++ language

Rory Fitzgerald



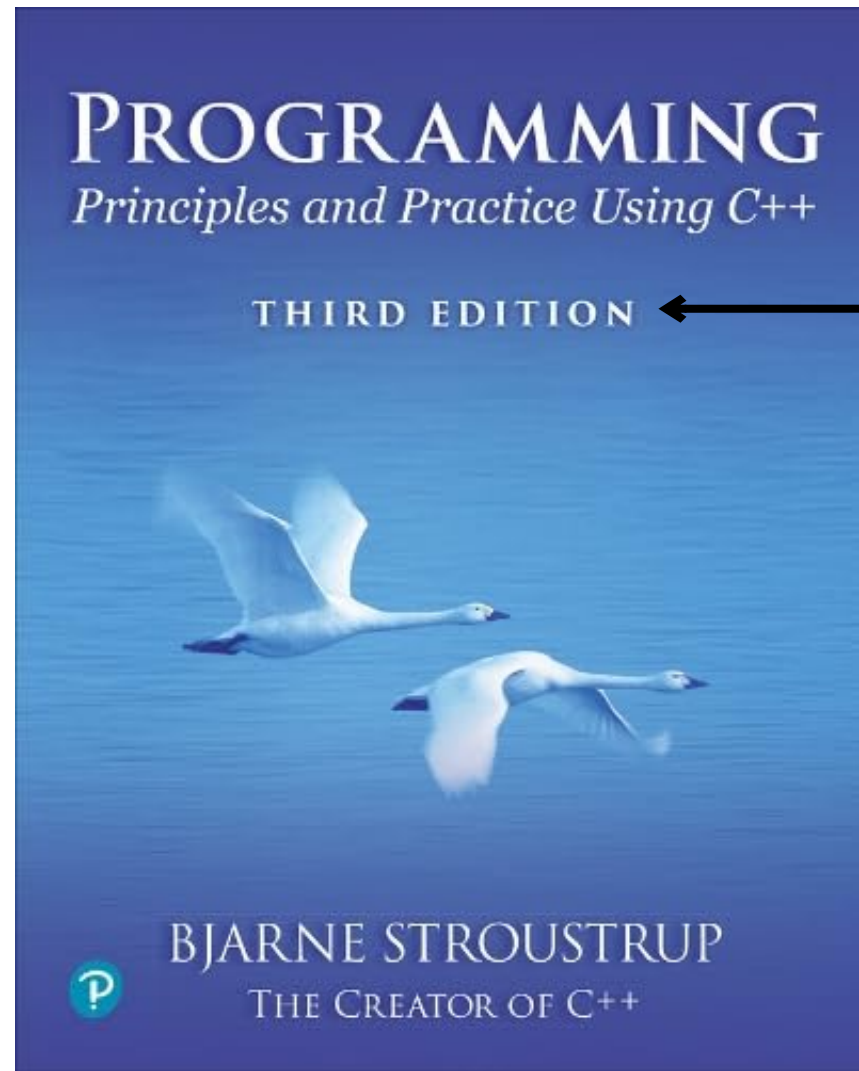
- Vit.Ass. at the Department of Computer Science (IDI)
- Master Student in Cybernetics
- Black Belt in Canvas Fu
- Elite operative in the Student Emergency Special Response Unit
- Guru of the C++ language

Today..

1. Answer questions about the course:
 - Who are these people?
 - **Where do we find information?**

May? Yes of course you may! <3

*Recommended
reading*



Note: **third** edition!

Canvas: Announcements and Course Information

Account

Dashboard

Courses

Calendar

Inbox

History

Commons

Help

2026 VÅR

Home

Syllabus

Pages

Announcements

Panopto

Nettside

Piazza

Inginious (øvingssystem)

Modules

Assignments

Collaborations

Grades

Files

Rubrics

Quizzes

Outcomes

Discussions

People

Settings

TDT4102-26V > Syllabus

6d View as Student

Immersive Reader

Recent Announcements

Løsning på Mac-problemer

Hei alle Mac-brukere, Det viste seg å være mer kluss med Ma...

Posted on: Jan 7, 2026, 9:11 AM

Teknisk hjelpeøkt

Det er satt opp tekniske hjelpeøkter tirsdag kl. 08-10 og fred...

Posted on: Jan 5, 2026, 6:17 PM

Velkommen til TDT4102 Prosedyre- og objektorientert programmering!

Vi håper alle har hatt en god jul og godt nyttår og er klare for ...

Posted on: Jan 4, 2026, 8:00 AM

TDT4102-26V :: Prosedyre- og objektorientert programmering

Jump to Today

Edit

Vi benytter i hovedsak fagets [nettside](#)

Grunnleggende og praksis-orientert programmering med programmeringsspråket C++. Emnet dekker det viktigste i programmeringsspråket samt utvalgte deler av standardbiblioteket. Gjennom øvingene får studentene omfattende erfaring i konstruksjon, feilfinning og testing av programvare. Mer om emnet kan man finne i emnebeskrivelsen: <https://www.ntnu.no/studier/emner/TDT4102#tab=omEmnet>

Info om undervisningsplan, øvinger, vurderinger og innlevering finner man her: <https://tdt4102.idi.ntnu.no/posts/plan/>

Dette emnet har et obligatorisk øvingsopplegg hvor man må ha 8 av 12 øvinger godkjent for å komme opp til

Course Status

Published

Import Existing Content

Import from Commons

Choose Home Page

View Course Stream

Course Setup Checklist

New Announcement

Course Analytics

View Course Notifications

January 2026

28	29	30	31	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31
1	2	3	4	5	6	7

Course assignments are not weighted.

08.01.2026
01:46 p.m.

Course website: Assignment submission and links

TDT4102

Info ▾ Dokumentasjon Øvinger Vurderinger Piazza Admin ▾ Logg ut

Velkommen til TDT4102 (C++)

TDT4102 er et kurs i C++ som gir en grundig innføring i programmeringsteknikker og prinsipper. Her finner du all nødvendig informasjon for å lykkes i kurset, vil du ha mer generell informasjon sjekk ut den [offisielle emnebeskrivelsen](#).

Øverst på siden finner du nyttige lenker til informasjon om emnet, øvingsopplegget og dokumentasjonen til AnimationWindow.

Under Vurderinger ser du hvor mange øvinger du har godkjent, husk at du må ha **8 av 12** øvinger og Inspiraøvingen godkjent for å gå opp til eksamen!

Kunngjøringer og emneinnhold (øvinger, lysbilder og livekode) i emnet legges ut på [Blackboard](#)

<https://tdt4102.idi.ntnu.no/>

Piazza: Forum for getting help

The screenshot displays the Piazza forum interface. The top navigation bar includes the Piazza logo, a dropdown menu for 'TDT 4102', and links for 'Q & A', 'Resources', 'Statistics', and 'Manage Class'. A user profile for 'Håkon Haugann' is visible in the top right corner.

The left sidebar contains a list of questions, each with a title, a timestamp, and a status icon. The questions include:

- Øving 2, oppgave 3b** (1/17/21): Jeg skjønner ikke hva oppgaven ber om at man skal skrive ut i selve tabellen. Skal det være en tom tabell? *An instructor thinks this is a good question*
- Problem oppstår når jeg prøver å kjøre...** (1/17/21)
- Øving 2, Oppgave 1a** (1/17/21): Hei! Hvordan får jeg et tall skrevet tilbake til skjermen?? Jeg har også prøvd void istedenfor int. int inputAndPrint()
- Ikke feil i Makefile for Linux** (1/17/21): I eksempel-Makefilen for Linux er det en feil i to av filstiene; både GRAPH_LIB_INCLUDE og GRAPH_LIB_LIB (linje 6 og 7)
- Øving 1 oppgave 2g** (1/17/21): Kodens min på oppgave 2g vil ikke kjøre: void naivePrimeNumberSearch(int n){ for (int number = 2; number < n; ++&
- Øving 1 oppgave 2e** (1/17/21): Kodens min: void triangleNumbersBelow(int n) { int acc = 1; int num = 2; cout << "Triangle numbers below
- problem: "cout" is ambiguous** (1/17/21): fikk dette som problem på alle cout og cin???
- 4b, -nan(ind)** (1/17/21): jobber med oppgave 4b, men får kun svaret jeg skal printe ut som -nan(ind) på det foreløpige utkastet av oppgaven: void
- Problemer med setup** (1/17/21): Når jeg kjører en kode får jeg masse tekst før det som skal skrives ut. Ser at forelesere ikke får all denne teksten før
- 2g** (1/17/21): skjønner ikke hva som er feil her? har sett på de andre svarene og sjekket krøllparentesene mine, men ser ut som de skal

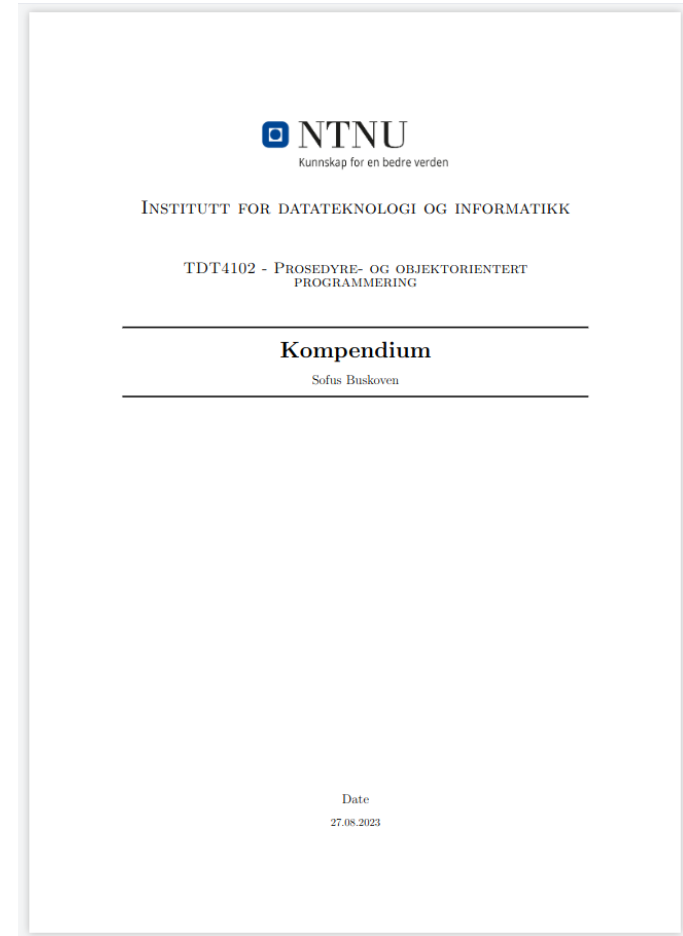
The main content area shows a detailed view of the question 'Øving 2, oppgave 3b'. The question text is: 'b) Definer en funksjon som skriver ut en gangetabell på skjermen (cout). La brukeren gi både bredde og høyde på tabellen. Velg selv navn for funksjonen. Slå opp i boken på setw for å finne såkalte manipulatorer som hjelper til med å skrive pene tabeller. Legg denne funksjonen til i testmenyen. Jeg skjønner ikke hva oppgaven ber om at man skal skrive ut i selve tabellen. Skal det være en tom tabell?'. The question has 169 views and is marked as a 'good question'. It was updated 11 months ago by an anonymous post.

Below the question, there is a section for 'the instructors' answer, where instructors collectively construct a single answer. The answer text is: 'Hvis jeg gir inn bredde 5 og høyde 3 skal programmet skrive en 3*5 gangetabell til skjerm:'. The answer is marked as a 'good answer' and was updated 11 months ago by Herman Berget.

At the bottom of the main content area, there is a section for 'followup discussions' for lingering questions and comments. It includes a 'Start a new followup discussion' button and a text input field for composing a new followup discussion.

Kompendium

- Written by another student who has taken TDT4102
- Only contains a few advanced topics
 - Will be published later when its contents become relevant



Today..

1. Answer questions about the course:

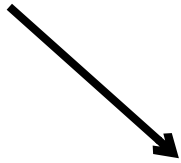
- Who are these people?
- Where do we find information?
- **What is the course about?**

May? Yes of course you may! <3

C++

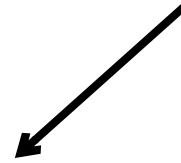
C ++

C



C

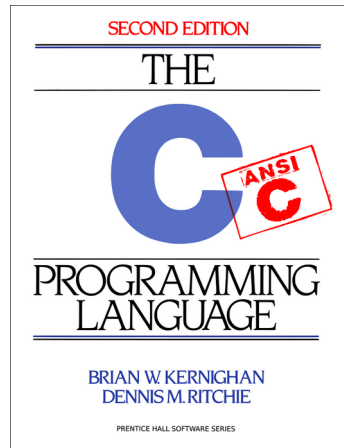
Simula 67



++

Dennis Ritchie

- Creator of C in 1972 at Bell labs
- One of the main developers of the UNIX operating system
- Won the Turing award in 1983
- Wrote *THE BOOK* on C



The C Programming Language

- C is itself one of the most popular programming languages in use today
- Created to be simpler than assembly

Assembly

```
circleArea:
    sub     sp, sp, #16
    str     s0, [sp, #12]
    ldr     s0, [sp, #12]
    fcvtnz  d1, s0
    adrp    x8, .LCPI0_0
    ldr     d2, [x8, :lo12:.LCPI0_0]
    fmul     d1, d2, d1
    ldr     s0, [sp, #12]
    fcvtnz  d2, s0
    fmul     d1, d1, d2
    fcvtnz  s0, d1
    add     sp, sp, #16
    ret
```

C

```
float circleArea(float radius) {
    return M_PI * radius * radius;
}
```


Kristen Nygaard & Ole-Johan Dahl

- Created the Simula programming language in 1962
- Established modern object oriented concepts
- Made Commander of the Order of St. Olav in 2000
- Won the Turing award in 2001



Kristen Nygaard

Ole-Johan Dahl

The Simula Programming Language

- First object-oriented programming language
- Influenced many of the most commonly used languages today (such as C++, Java, and C#)

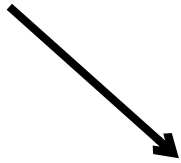
```
Class Rectangle (Width, Height); Real Width, Height;
Begin
    Real Area, Perimeter;

    Procedure Update;
    Begin
        Area := Width * Height;
        Perimeter := 2*(Width + Height)
    End of Update;

    Boolean Procedure IsSquare;
        IsSquare := Width=Height;
    Update;

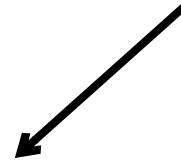
    OutText("Rectangle created: "); OutFix(Width,2,6);
    OutFix(Height,2,6); OutImage
End of Rectangle;
```

C



C

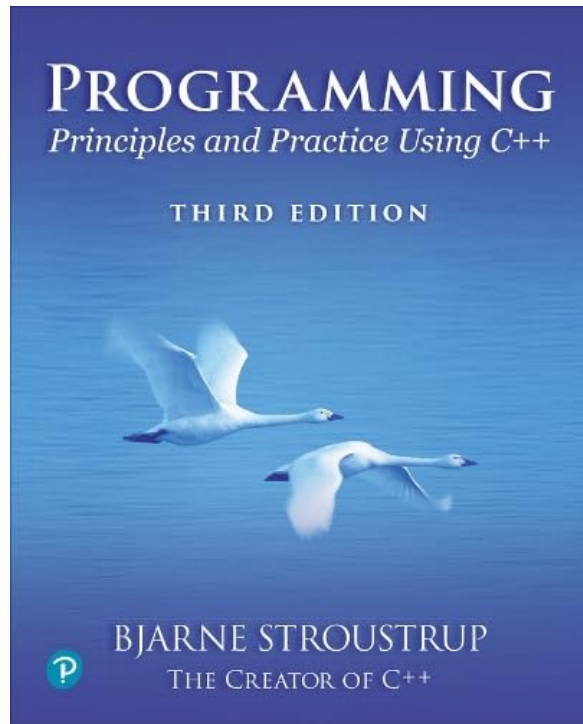
Simula 67



++

Bjarne Stroustrup

- Creator of C++ in 1979
- Author of the course book



What is C++?

- C++ is a compiled programming language

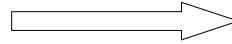
What is C++?

- C++ is a compiled programming language
 - A C++ program is compiled (translated) into machine code
 - The compiler is the program that performs this translation

From C++ source code..

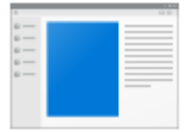
```
float circleArea(float radius) {  
    return M_PI * radius * radius;  
}
```

Compiler



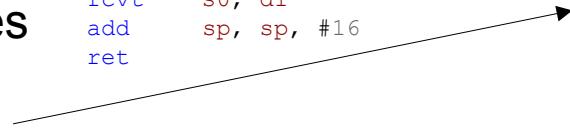
.. To machine code

```
circleArea:  
    sub    sp, sp, #16  
    str    s0, [sp, #12]  
    ldr    s0, [sp, #12]  
    fcvtd  d1, s0  
    adrp   x8, .LCPI0_0  
    ldr    d2, [x8, :lo12:.LCPI0_0]  
    fmul   d1, d2, d1  
    ldr    s0, [sp, #12]  
    fcvtd  d2, s0  
    fmul   d1, d1, d2  
    fcvtd  s0, d1  
    add    sp, sp, #16  
    ret  
  
D10043FF  
BD000FE0  
BD400FE0  
1E22C000  
90000000  
FD443801  
1E610801  
BD400FE0  
1E22C000  
1E600820  
1E624000  
910043FF  
D65F03C0
```



program.exe

Your program becomes a bunch of ones and zeroes that the processor can understand, also known as «a binary»



What is C++?

- C++ is a compiled programming language
 - A C++ program is compiled (translated) into machine code
 - The compiler is the program that performs this translation
- C++ is low-level: programs are executed on «bare metal»



program.exe

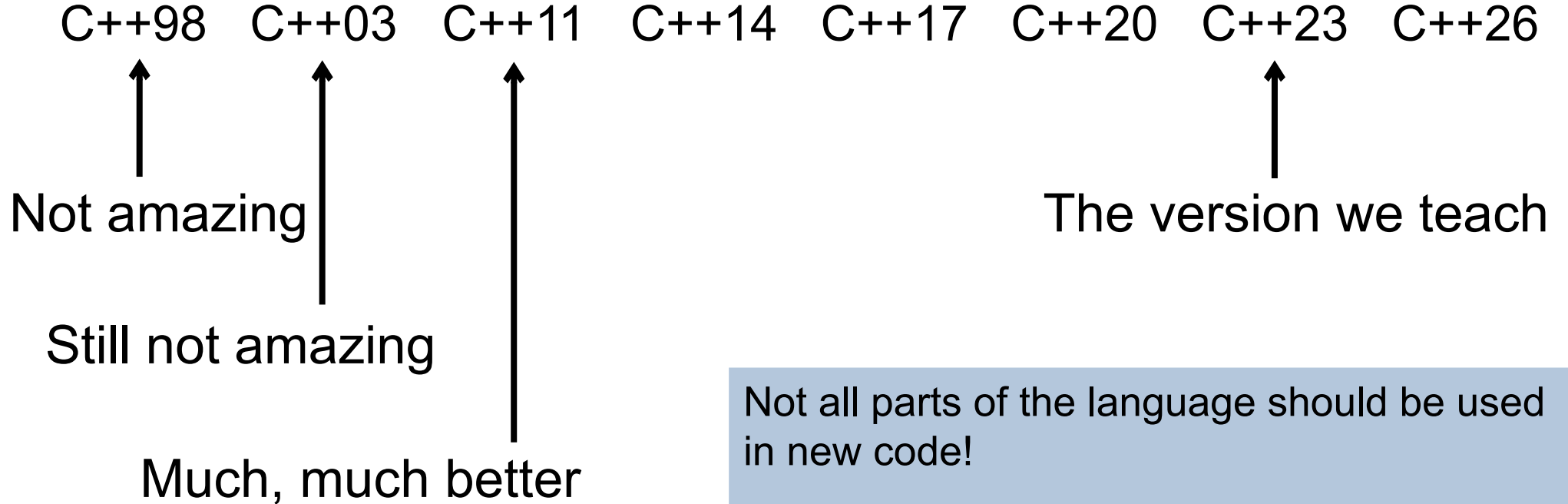
```
circleArea:
    sub    sp, sp, #16
    str    s0, [sp, #12]
    ldr    s0, [sp, #12]
    fcvtn  d1, s0
    adrp   x8, .LCPI0_0
    ldr    d2, [x8, :lo12:.LCPI0_0]
    fmul   d1, d2, d1
    ldr    s0, [sp, #12]
    fcvtn  d2, s0
    fmul   d1, d1, d2
    fcvtn  s0, d1
    add    sp, sp, #16
    ret

D10043FF
BD000FE0
BD400FE0
1E22C000
90000000
FD443801
1E610801
BD400FE0
1E22C000
1E600820
1E624000
910043FF
D65F03C0
```

What is C++?

- C++ is a compiled programming language
 - A C++ program is compiled (translated) into machine code
 - The compiler is the program that performs this translation
- C++ is low-level: programs are executed on «bare metal»
- C++ is lean: you generally don't pay for what you don't use
- C++ is flexible: it allows many different programming styles

Major language revisions



Not all parts of the language should be used in new code!

They only still exist to keep old code working.

Today..

1. Answer questions about the course:

- Who are these people?
- Where do we find information?
- What is the course about?

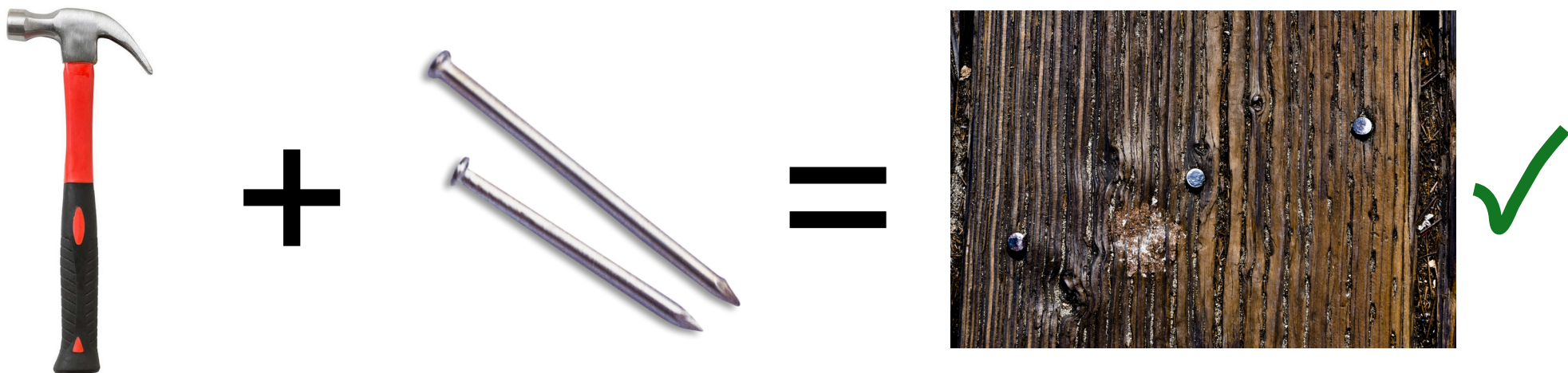
- **Why do we care?**

May? Yes of course you may! <3

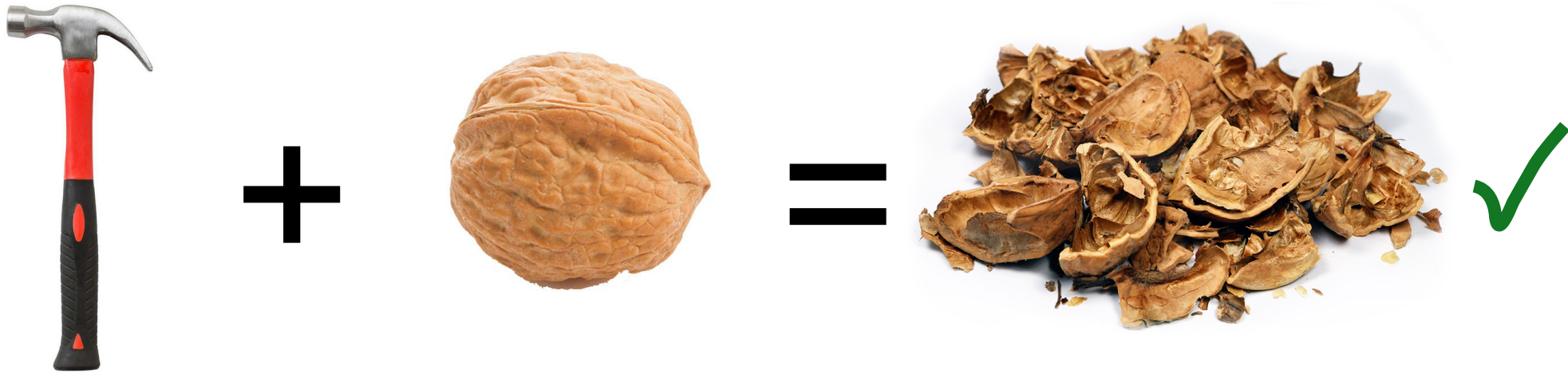
Meanwhile, C++ is a tool..



C++ is a tool..

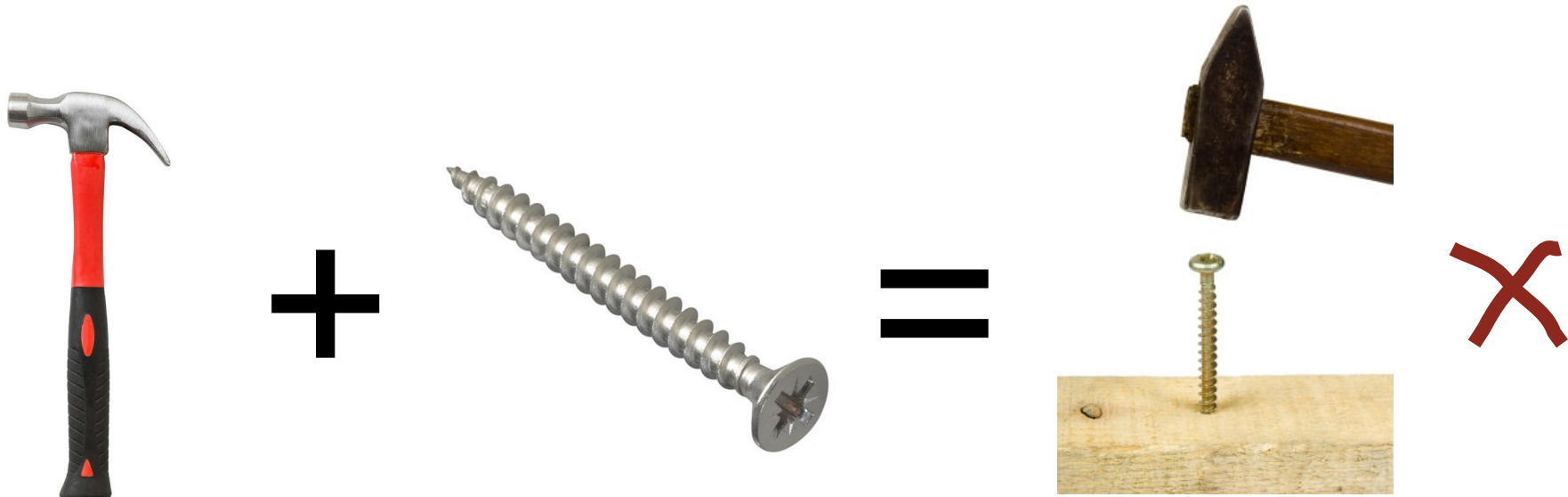


C++ is a tool..



.. Which works well for many applications

C++ is a tool..



.. Which works well for many applications,
but not for others.

Any tool has advantages and disadvantages

C++	Python
<ul style="list-style-type: none">• ???	<ul style="list-style-type: none">• ???

Let's do a speed test..

$$\sum_{i=1}^{100,000,000} i = 1 + 2 + 3 + (\dots) + 99,999,999 + 100,000,000$$

Python

C++

Let's do a speed test..

$$\sum_{i=1}^{100,000,000} i = 1 + 2 + 3 + (\dots) + 99,999,999 + 100,000,000$$

Python: 6.001s

C++

Let's do a speed test..

$$\sum_{i=1}^{100,000,000} i = 1 + 2 + 3 + (\dots) + 99,999,999 + 100,000,000$$

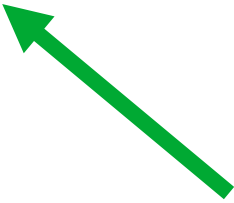
Python: 6.001s

C++: 0.149s (40.3x faster!)

Any tool has advantages and disadvantages

C++	Python
<ul style="list-style-type: none">• Code runs at the speed the processor is capable of	<ul style="list-style-type: none">• Code runs pretty slow

Let's do a speed test..

$$\sum_{i=1}^{100,000,000} i = 1 + 2 + 3 + (\dots) + 99,999,999 + 100,000,000$$


Python: 6.001s

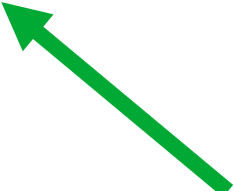
The result of this sum
is always the same!

C++: 0.149s (40.3x faster!)

Let's do a speed test..

$$\sum_{i=1}^{100,000,000} i = 1 + 2 + 3 + (\dots) + 99,999,999 + 100,000,000$$

Python: 6.001s



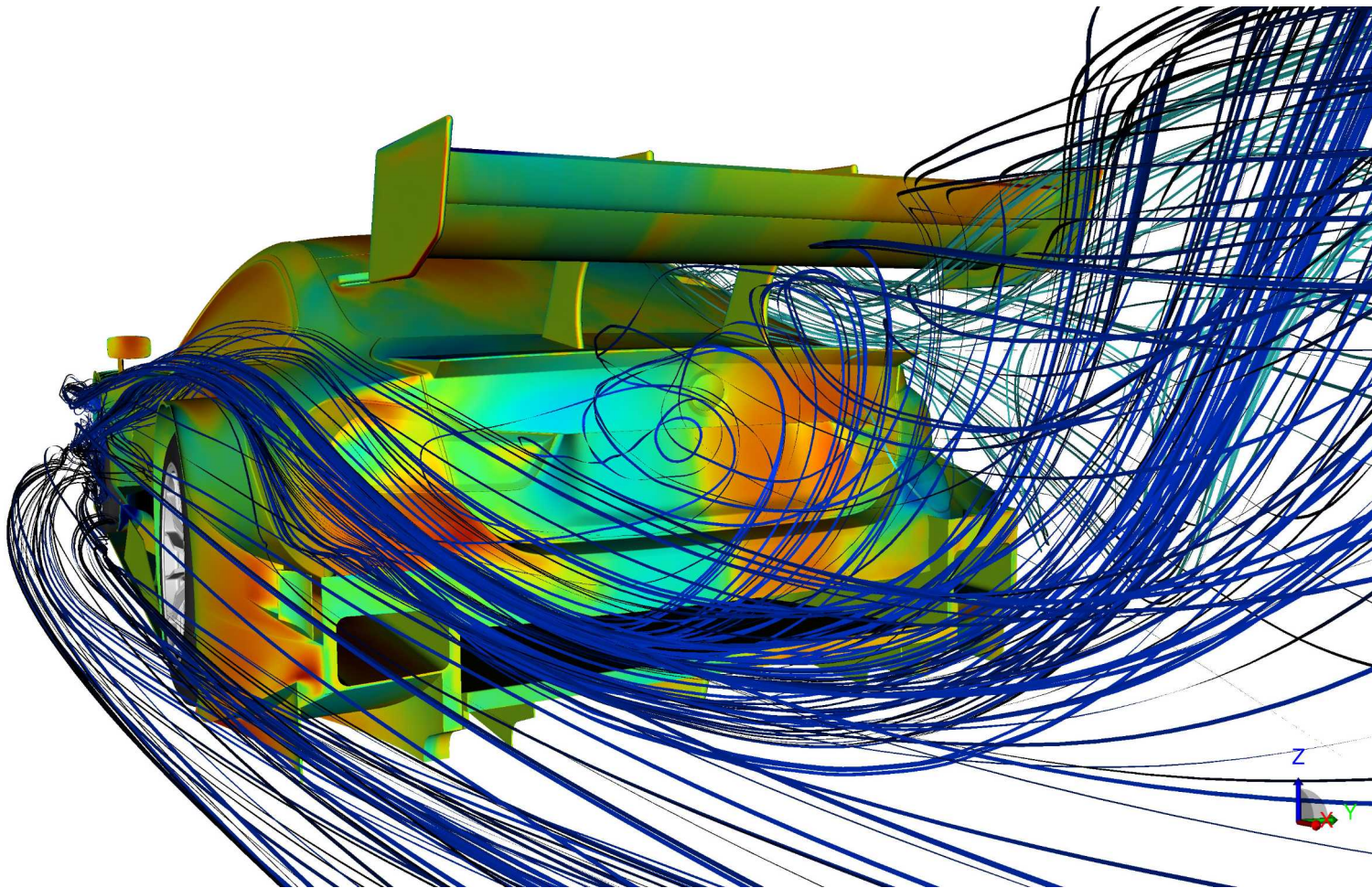
The result of this sum
is always the same!

C++: 0.149s (40.3x faster!)

C++ (optimised): 0.002s (3000x faster!)

Any tool has advantages and disadvantages

C++	Python
<ul style="list-style-type: none">• Code runs at the speed the processor is capable of• Code can be automatically optimised	<ul style="list-style-type: none">• Code runs pretty slow• Options for optimising code are limited *

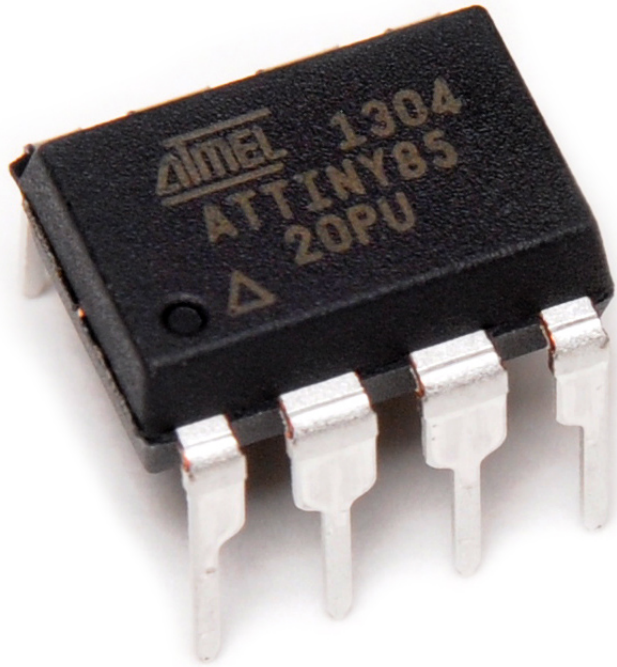


*

- Certain Python libraries (e.g. numpy) have portions of their code implemented in C or C++
- If you need something simple, this is likely easier than using C++
- But: tiny changes can reduce your performance to Python speeds

Any tool has advantages and disadvantages

C++	Python
<ul style="list-style-type: none">• Code runs at the speed the processor is capable of• Code can be automatically optimised• A number of common mistakes are caught by the compiler	<ul style="list-style-type: none">• Code runs pretty slow• Options for optimising code are limited (or require using C++)• Some common errors may not be caught until the program is run



Specifications:

- Memory: 512 bytes
- Storage: 8KB
- 20M instructions / second

Can't run Python

C is most popular, but possible to use many features of C++

<https://www.atomsindustries.com/p1005>

Any tool has advantages and disadvantages

C++	Python
<ul style="list-style-type: none">• Code runs at the speed the processor is capable of• Code can be automatically optimised• A number of common mistakes are caught by the compiler• Can be used on microcontrollers	<ul style="list-style-type: none">• Code runs pretty slow• Options for optimising code are limited (or require using C++)• Some common errors may not be caught until the program is run• Can't be used on microcontrollers

Any tool has advantages and disadvantages

C++	Python
<ul style="list-style-type: none">• Code runs at the speed the processor is capable of• Code can be automatically optimised• A number of common mistakes are caught by the compiler• Can be used on microcontrollers• Massive number of libraries available	<ul style="list-style-type: none">• Code runs pretty slow• Options for optimising code are limited (or require using C++)• Some common errors may not be caught until the program is run• Can't be used on microcontrollers• Massive number of libraries available

Any tool has advantages and disadvantages

C++	Python
<ul style="list-style-type: none">• Code runs at the speed the processor is capable of• Code can be automatically optimised• A number of common mistakes are caught by the compiler• Can be used on microcontrollers• Massive number of libraries available• No single go-to package manager	<ul style="list-style-type: none">• Code runs pretty slow• Options for optimising code are limited (or require using C++)• Some common errors may not be caught until the program is run• Can't be used on microcontrollers• Massive number of libraries available• Has a package manager (pip) for easily installing libraries

Any tool has advantages and disadvantages

C++	Python
<ul style="list-style-type: none">• Code runs at the speed the processor is capable of• Code can be automatically optimised• A number of common mistakes are caught by the compiler• Can be used on microcontrollers• Massive number of libraries available• No single go-to package manager• Compilation is time consuming	<ul style="list-style-type: none">• Code runs pretty slow• Options for optimising code are limited (or require using C++)• Some common errors may not be caught until the program is run• Can't be used on microcontrollers• Massive number of libraries available• Has a package manager (pip) for easily installing libraries• Program runs instantly

More generally:

Programming is everywhere.

Programming is making its inroads
into every field of study.

The ability to write code well in addition to
your own field of study is highly valuable and
sought after in industry.

A look at the future: hardware trends

Early computers

IBM 704



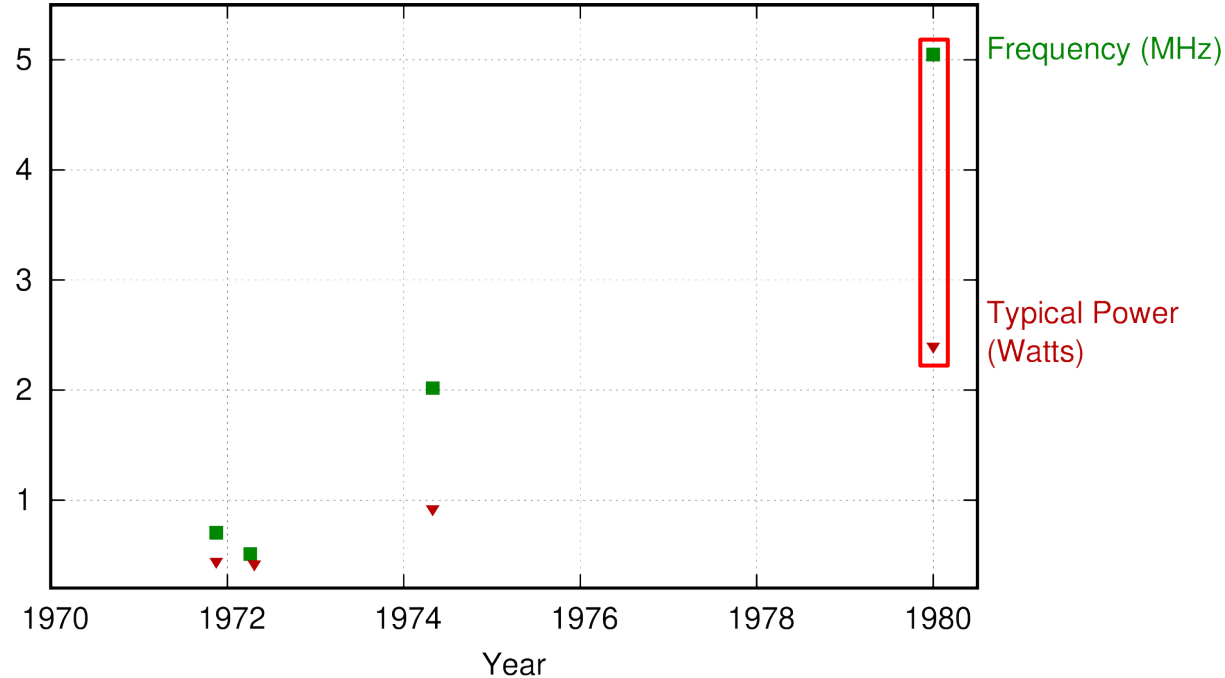
- Launched in 1954
- ~12 000 calculations per second (FLOPS)
- Likely consumed more than 100 kW (!!)

1970 - 1980

Intel 8087



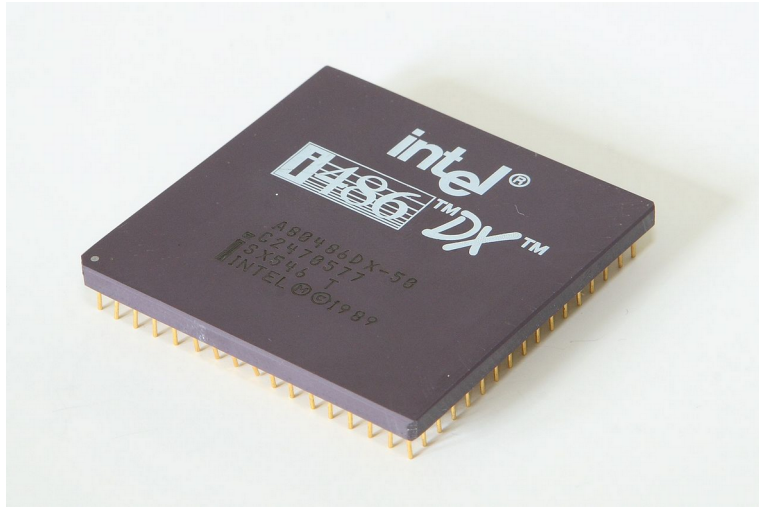
- Launched in 1979
- ~5 000 000 instructions per second
- Consumed 2.4W of power



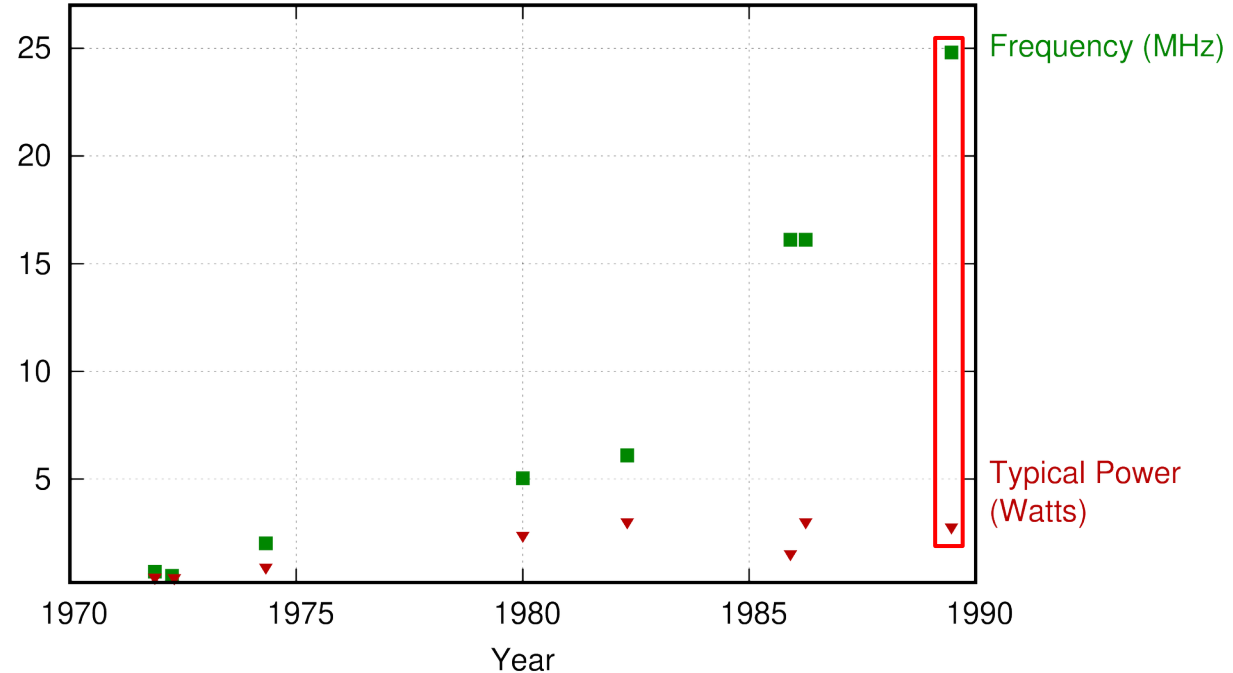
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2021 by K. Rupp

1980 - 1990

Intel i486

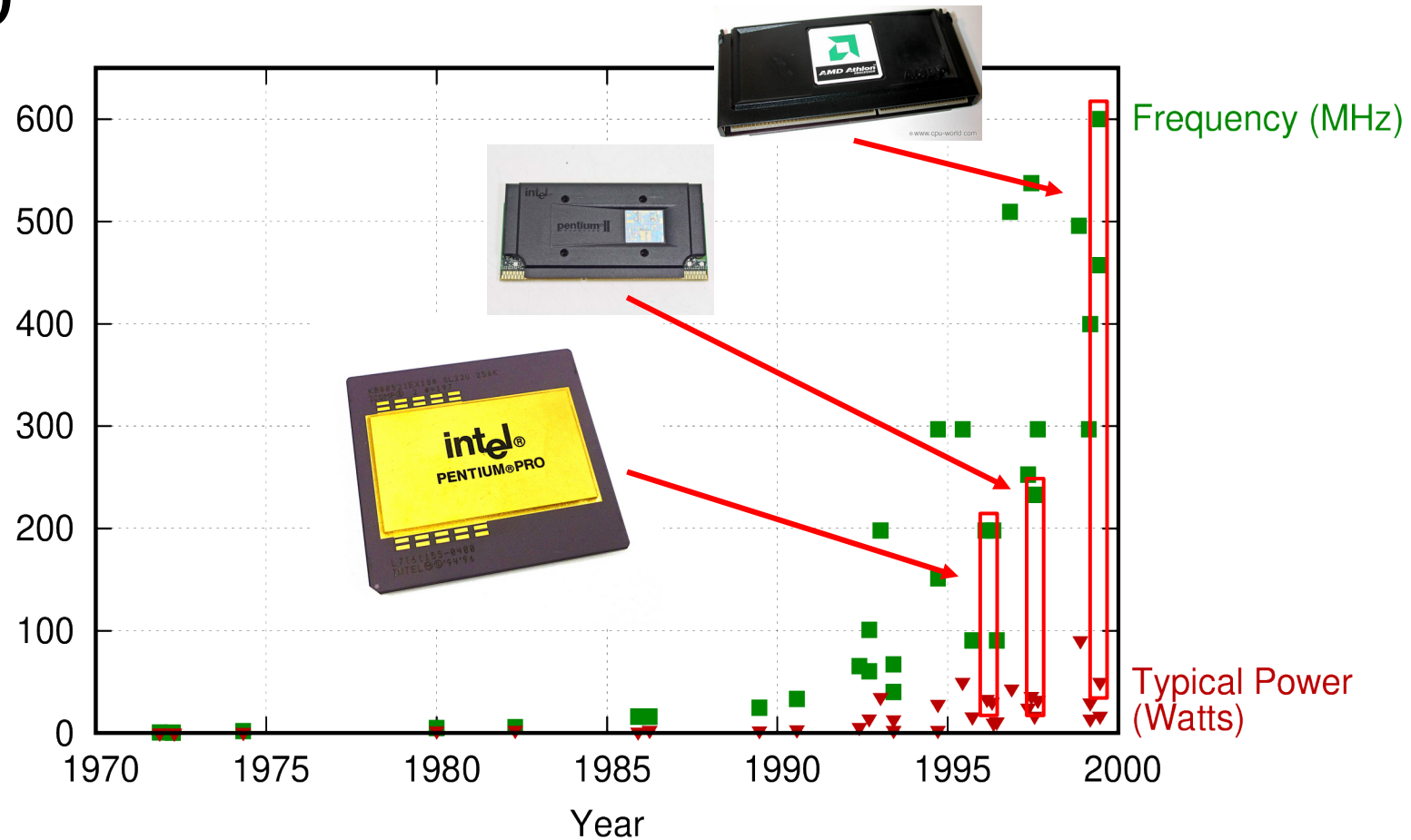


- Launched in 1989
- ~25 000 000 instructions per second
- Consumed 2.8W of power



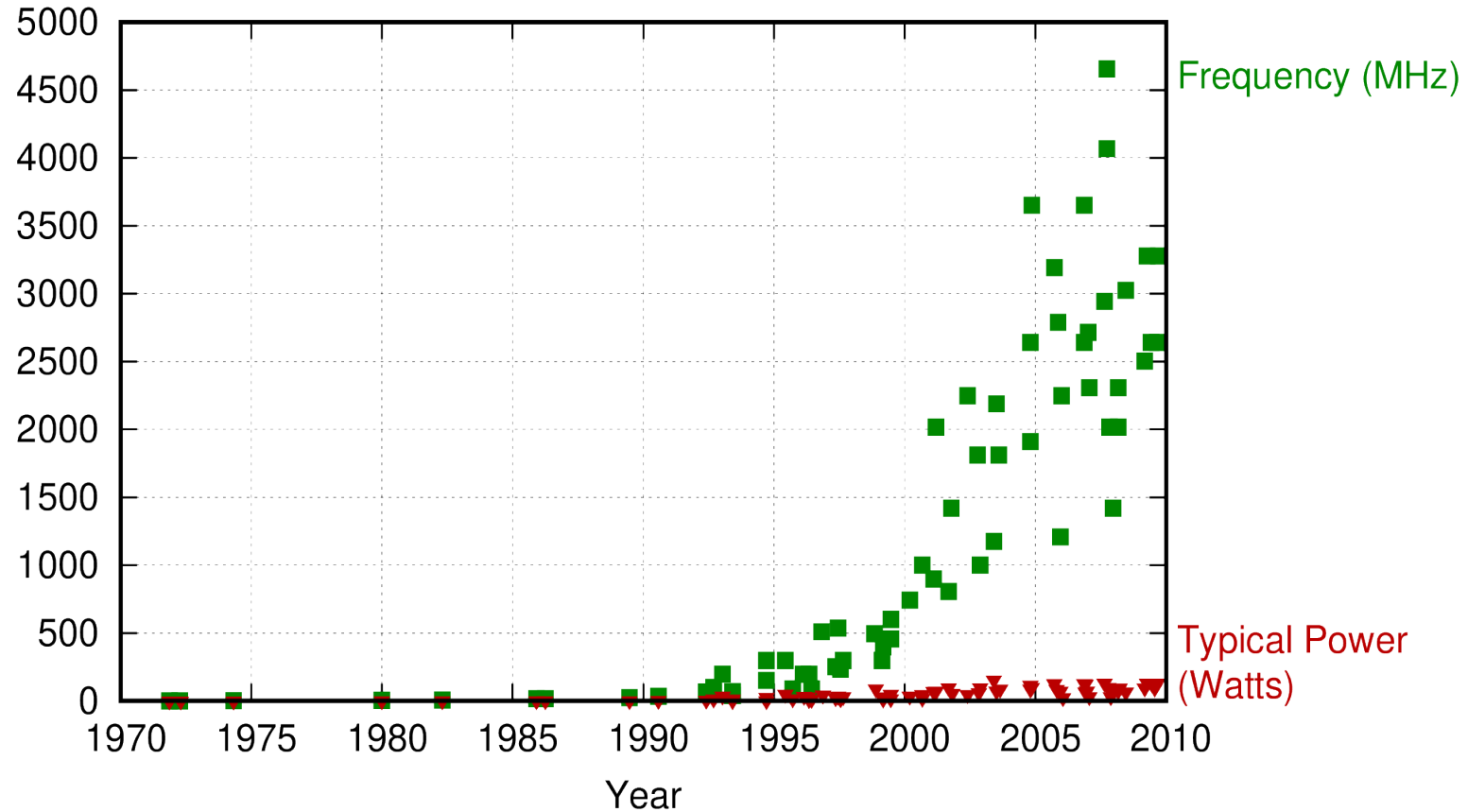
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2021 by K. Rupp

1990 - 2000



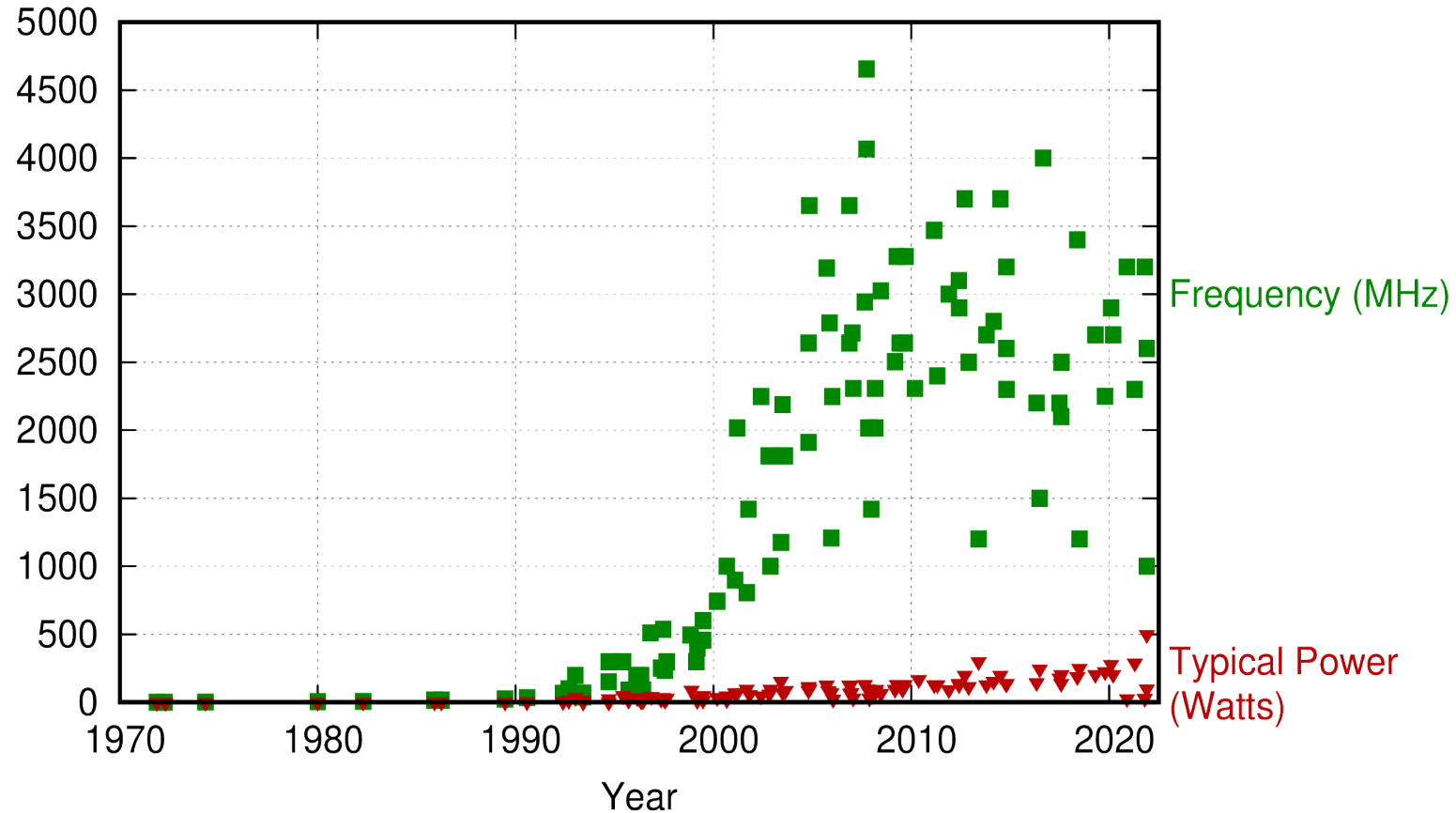
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2021 by K. Rupp

1990 - 2010



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2021 by K. Rupp

2010 - 2023



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2021 by K. Rupp

Today..

1. Answer questions about the course:

- Who are these people?
- Where do we find information?
- What is the course about?
- Why do we care?
- **When are the lectures and deadlines?**

May? Yes of course you may! <3

Forelesningsplan (tentativ)				
Uke	Tirsdag 8:15 – 10, R1	Torsdag 14:15 – 16, R1	Fredag 10:15 – 12, R1	Lever
2	Teknisk støtte	Forelesning 1	Teknisk støtte	Øving 0
3	Øvingsforelesning 1	Forelesning 2	Forelesning 3	Øving 1
4	Øvingsforelesning 2	Forelesning 4	Forelesning 5	Øving 2
5	Øvingsforelesning 3	Forelesning 6	Øvingsforelesning 4	Øving 3
6		Forelesning 7	Øvingsforelesning 5	Øving 4
7		Forelesning 8	Øvingsforelesning 6	Øving 5
8		Forelesning 9	Øvingsforelesning 7	Øving 6
9		Forelesning 10	Øvingsforelesning 8	Øving 7
10		Forelesning 11	Øvingsforelesning 9	Øving 8
11		Forelesning 12	Prosjekt	Øving 9
12		Forelesning 13		
13		Forelesning 14		
14	Påskeferie			
15	Insperaøving			Prosjekt
16		Oppsummering 1/2	Eksamenssett	Demonstrer prosjekt
17		Oppsummering 2/2 Prosjektene	Eksamenssett	

Eksamenskrav

For å gå på eksamen stilles følgende krav:

- Krav 1: 8 / 12 øvinger må være godkjent.
Prosjektet teller som 3 øvinger
- Krav 2: Inspiraøvingen må være godkjent

Eksamen: 4. mai

- Pulje 1: kl. 9:00
- Pulje 2: kl. 15:00

Din pulje vises på studentweb kort før eksamen

Project

- Counts as 0 - 3 assignments
- Write a program in C++ from the ground up
 - Can be anything you feel like
 - You may work alone or in pairs
 - Some requirements that should not be difficult to satisfy
- More information will follow closer to the kick-off date

Project

- What kind of projects have people done in the past?

BEAR

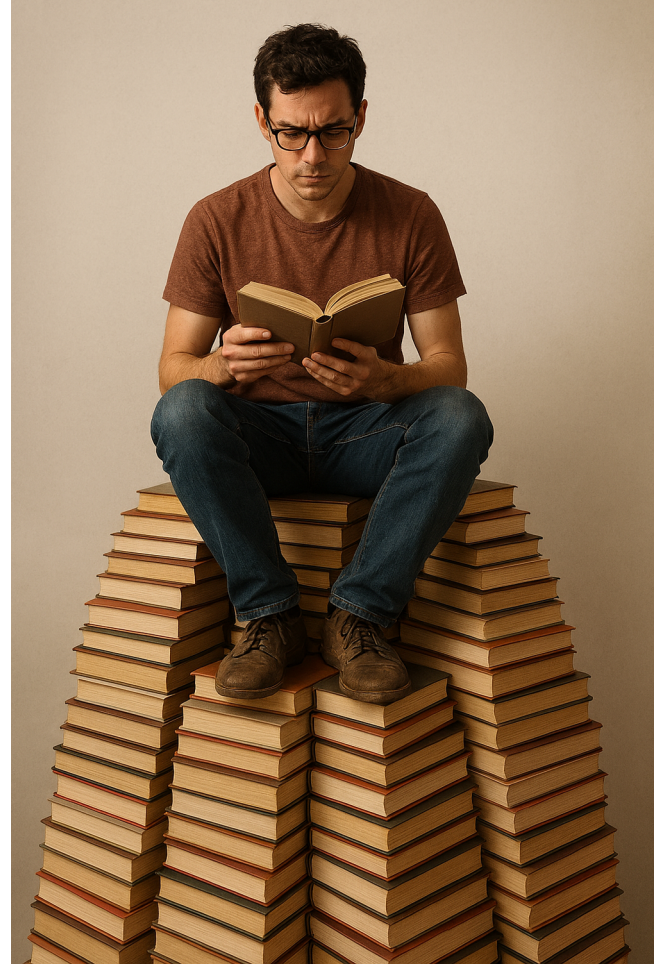


WITH ME

May? Yes of course you may! <3



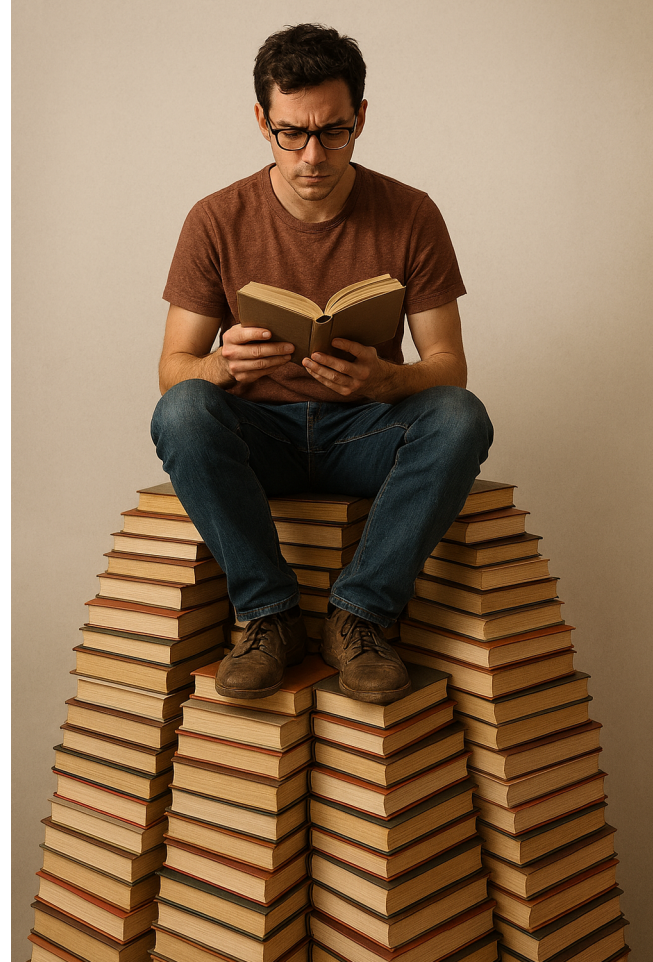
System 1



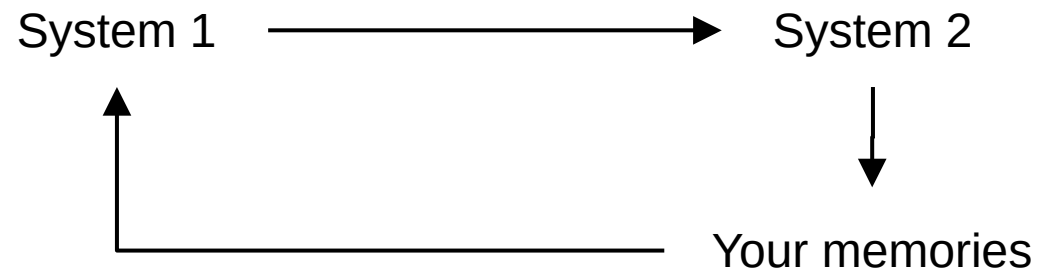
System 2



$$5 \times 4 = ?$$



$$17 \times 24 = ?$$

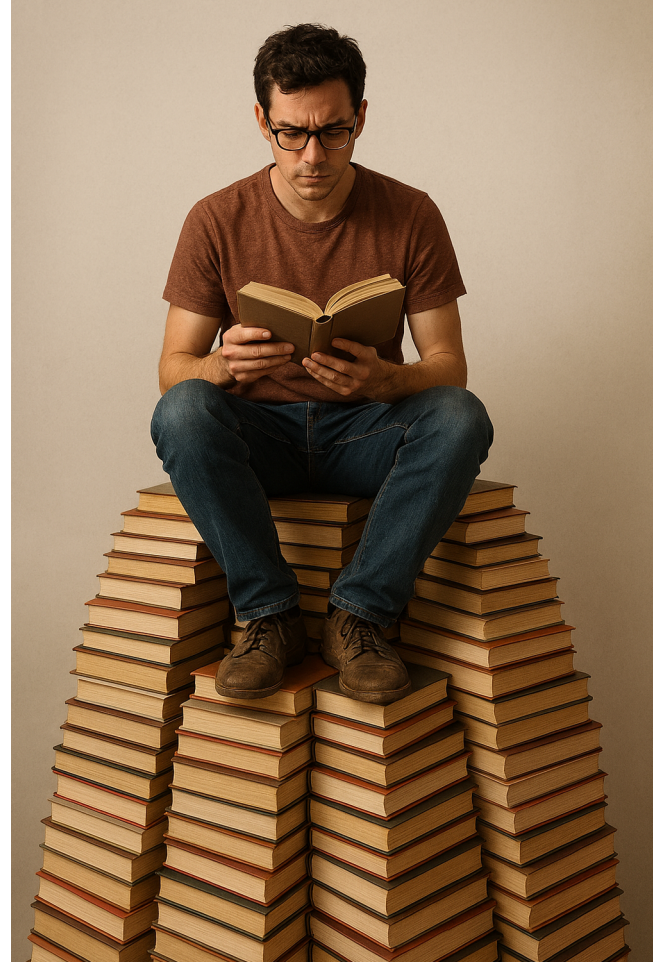




60 MINUTES Overtime

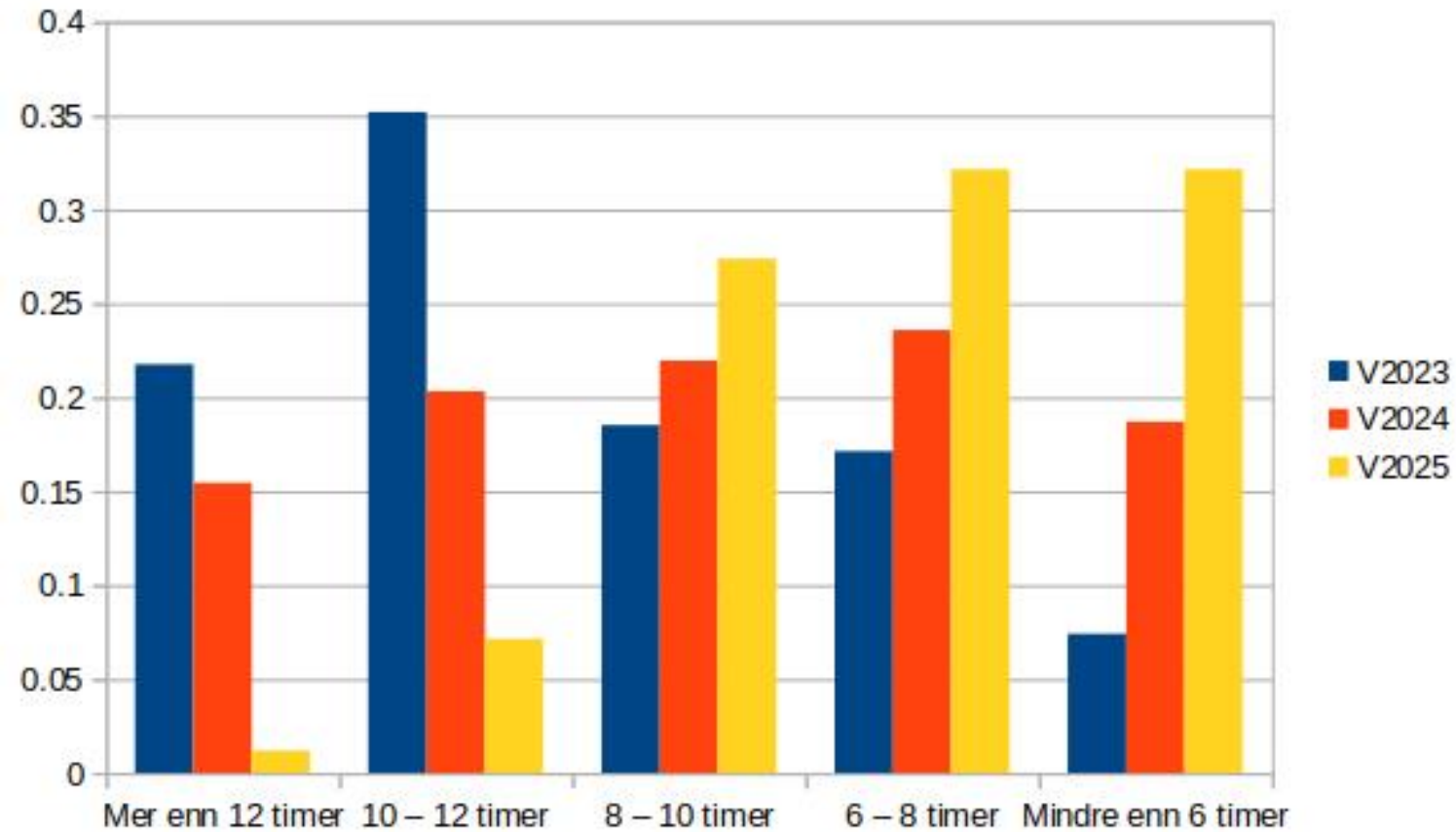


$$5 \times 4 = ?$$



$$17 \times 24 = ?$$





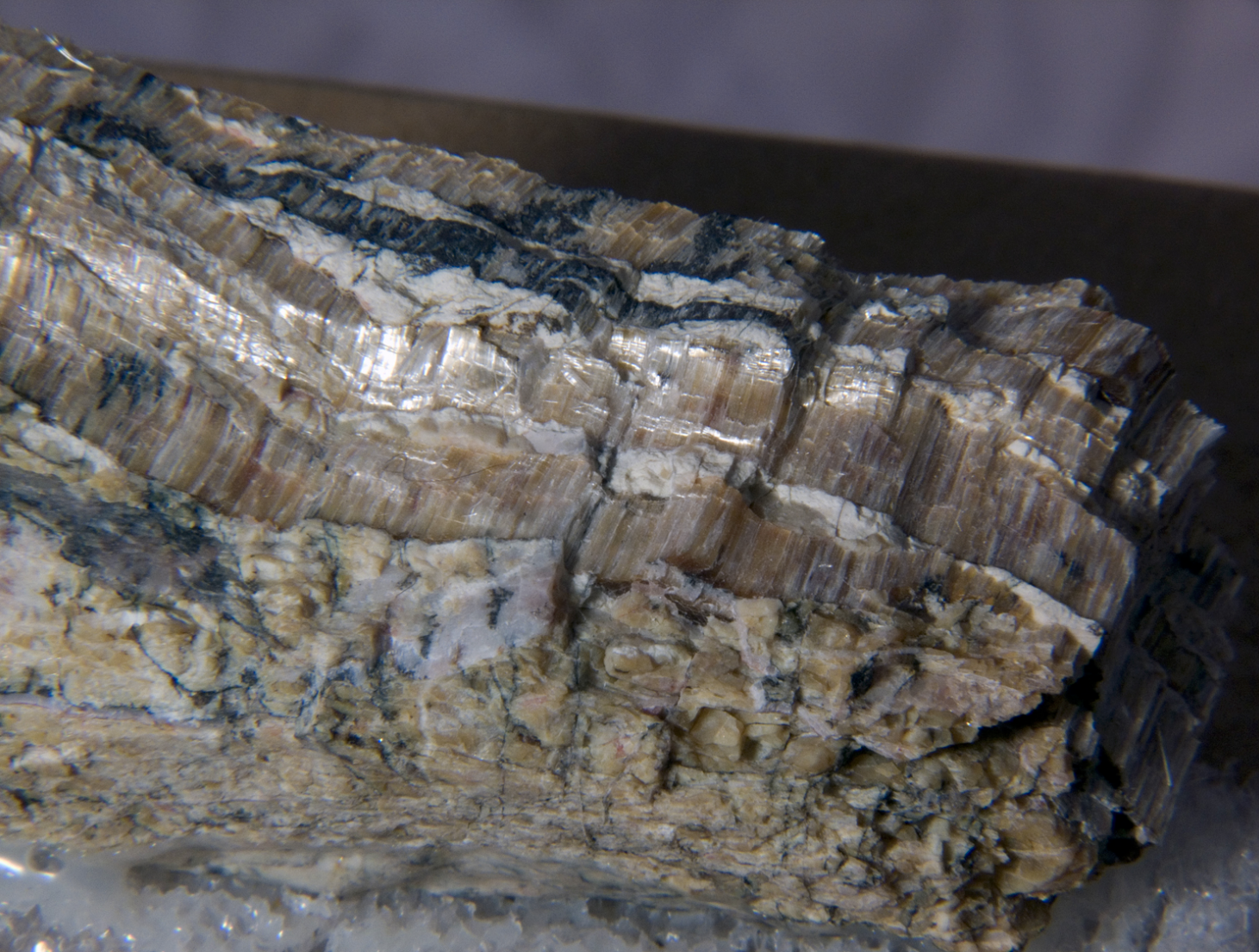
Our observations:

- Far less time spent per assignment
- Censors have noticed that ability shown on the exam has noticeably degraded these past two years

- Most of your body functions in a «use it or leave it» kind of way
- The same is true for your head
- Learning takes time



Language models have amazing potential

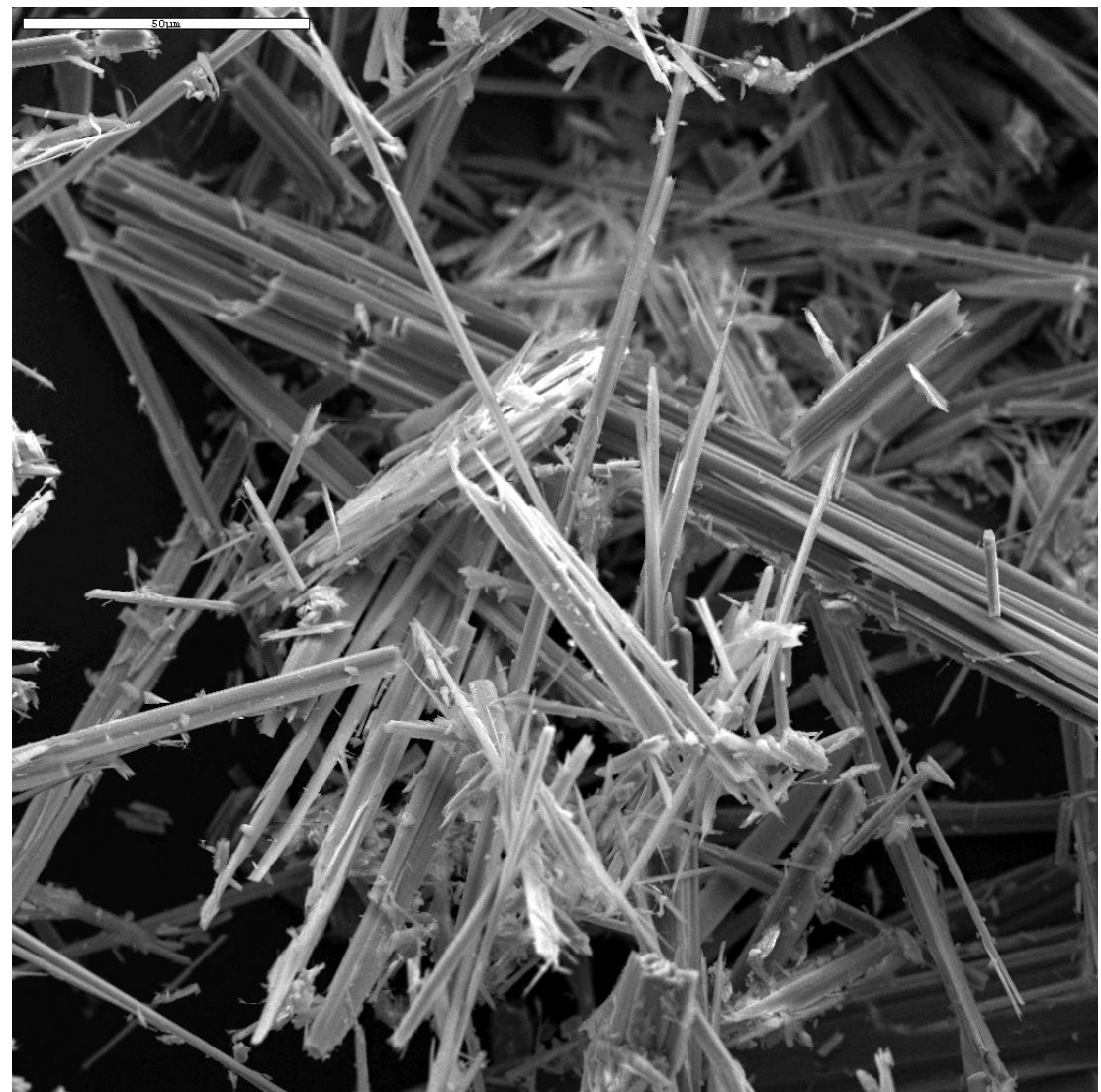


- Highly fire resistant
- Thermal insulator
- Electrical insulator
- Flexible
- Cheap



Asbestos

- There were signs of its dangers already by the 1930's
- Took until the 1980's before the material was finally banned



Language models have amazing **potential**



Reading code is harder than writing it.

English

C++ is a powerful, high-performance programming language that supports both procedural and object-oriented programming paradigms, widely used for system/software development, game development, and real-time simulations due to its efficiency and control over system resources.

Japanese

ブラウンチーズ（ブリュノスト）の作り方は、まずヤギの乳を煮詰めて水分を飛ばし、濃縮させます。次に、乳糖がキャラメル化するまでさらに煮詰め、濃厚な茶色のペースト状になるまで加熱します。最後に、冷やして固めると、独特の甘みと風味を持つブラウンチーズが完成します。

AI is a tool



We currently do not properly understand AI

It has some good use cases, but our own observations do not show a benefit for learning (if anything more the opposite)

It is easy to get into a cycle of relying too much on AI and avoid activating your brain

My recommendation:

Avoid AI as much as possible.

My recommendation:

Avoid AI as much as possible.

And also:

AI will not be available on the exam.

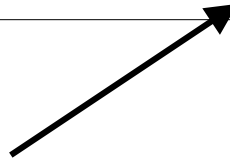
Tips

Tips

- **Find a friend**
- Use all available resources (Piazza and student assistants)
- Handwriting has been shown to force the brain to process incoming information
 - Must be handwriting, typing is far less effective
 - <https://www.forskning.no/barn-og-ungdom-hjernen-ntnu/barna-ma-fra-forste-klasse-forst-laere-a-skrive-for-hand/2414148>

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Assignment published			Regular Lecture	Assignment lecture		
				Assignment deadline		

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Assignment published			Regular Lecture	Assignment lecture		
				Assignment deadline		



We expect a lot of server load on fridays.

Make sure to start early!

That also gives you time to get help.

Tips

- **Find a friend**
- Use all available resources (Piazza and student assistants)
- Handwriting has been shown to force the brain to process incoming information
 - Must be handwriting, typing is far less effective
 - <https://www.forskning.no/barn-og-ungdom-hjernen-ntnu/barna-ma-fra-forste-klasse-forst-laere-a-skrive-for-hand/2414148> '
- Start early
- Learning takes time, and requires lots of effort. Make sure to reward yourself for that!!

See also



Veritasium: What
everyone gets
wrong about AI
and learning

May? Yes of course you may! <3

Today..

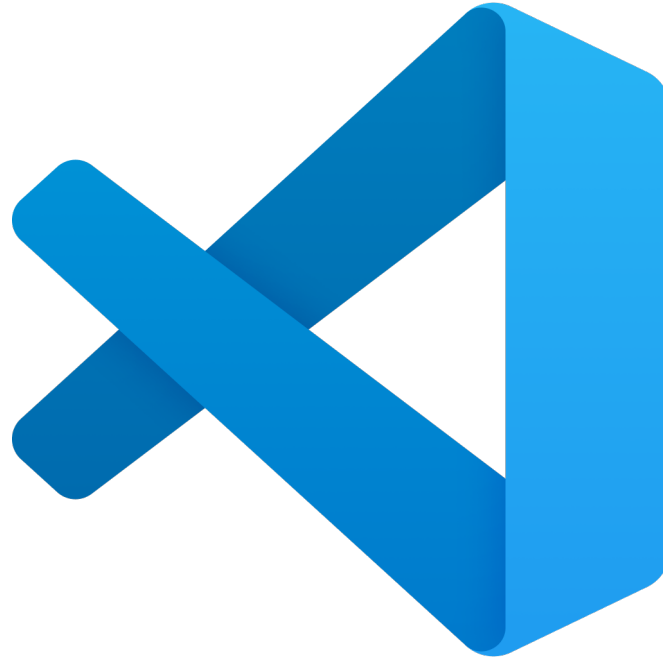
1. Answer questions about the course:

- Who are these people?
- Where do we find information?
- When are the lectures and deadlines?
- What is the course about?
- Why do we care?
- **How do we do the coursework?**

2. First steps in C++

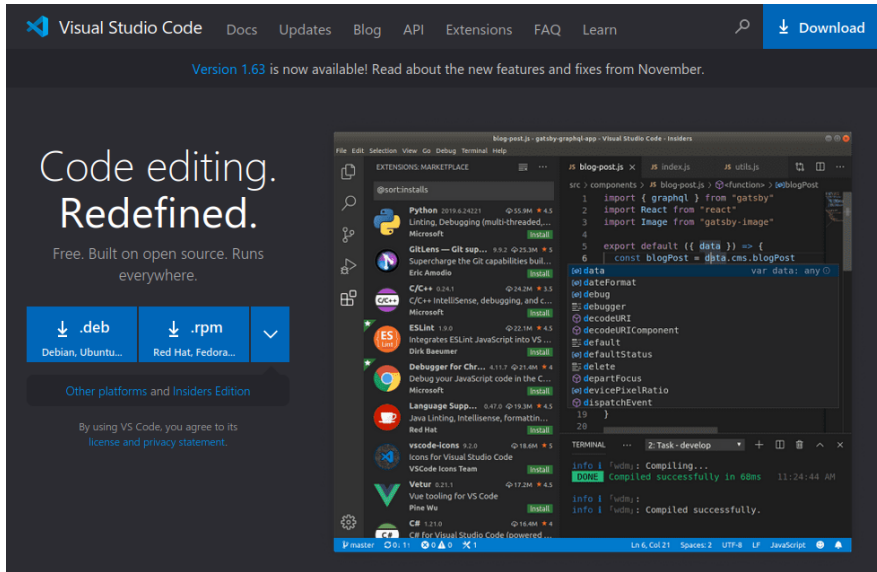
May? Yes of course you may! <3

We will use the Visual Studio Code editor



Assignment 0 (Abridged Edition)

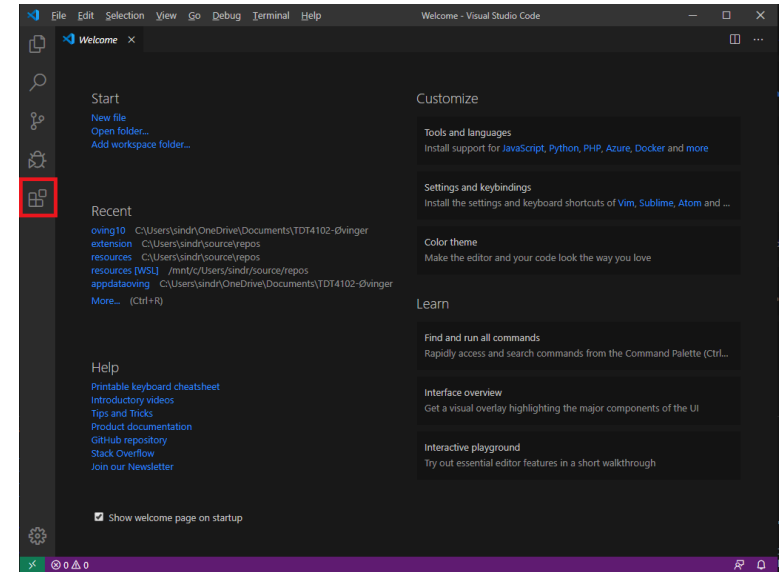
1. Go to <https://code.visualstudio.com>



2. Download the installer
3. Run the installer
(refer to assignment 0 for which settings to use)

4. Open VS Code

5. Install the TDT4102 extension

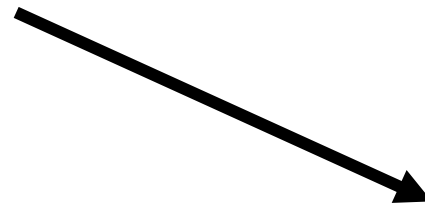


6. Use Ctrl+Shift+P, then
«install required tools»

How to run the code examples

Each example we discuss during the lectures is available in VS Code.

1. Press Ctrl+Shift+P (or Cmd+Shift+P on Mac)
2. Use the command «Create Project from TDT4102 Template»
3. Select «Lectures»
4. Select the lecture number
5. Select the example number shown on the slide



Example 1

Today..

1. Answer questions about the course:

- Who are these people?
- Where do we find information?
- When are the lectures and deadlines?
- What is the course about?
- Why do we care?
- How do we do the coursework?

2. First steps in C++

May? Yes of course you may! <3

Demonstration:

Basics of Compiling and Running C++ Programs

Example 1

Summary of everything shown:

- It is necessary to configure a project before running it. Configuring a project sets up the C++ compiler and ensures everything will compile without any problems.
- Run a program by clicking on Run > Start Debugging
- Code must be placed inside `main()`
- Each “sentence” must be terminated using a `;`
- The equivalent of Python’s `print()` function is `cout << “message” << endl;`
- Unlike Python, whitespace and indentation has no meaning in C++
- The equivalent of Python’s `import` is `#include “”`
- The debugger allows you to run your program one line at a time.

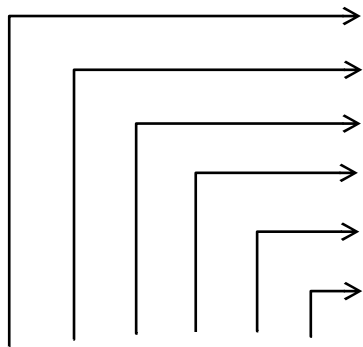
Example 1

```
3  int main() {  
4  |      cout << "A very Happy New Year!" << endl;  
  |
```



Breakpoint: when using the “Start Debugging” option to run your program, you can click in the left hand margin in VS code to set a breakpoint. The program will pause execution and let you inspect what is going on when it encounters a line with one active.

When it does so, the following panel will show up:



“Continue”:

Stop at the next breakpoint

“Step over”:

Stop at the next line in this function

“Step into”:

Stop at the next line of code

“Step out”:

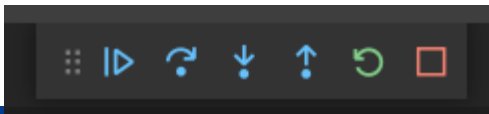
Stop at the next line of code after the function exits

“Restart”:

Restart the program

“Stop”:

Stop the program



Example 2