Imperativ och objektorienterad programmeringsmetodik

Föreläsning 1 av många

Tobias Wrigstad

Tobias Wrigstad

Professor i datalogi

Programspråk

Programspråksdesign

Typsystem

Exekveringsmiljöer

Concurrency

Parallellprogrammering

Minneshantering

Mastery learning

Grundutbildningsprefekt 2020–

Undervisning under #2020

Alla föreläsningar och labbar sker ”digitalt”

Obs — ingen risk att UU går över till digital undervisning

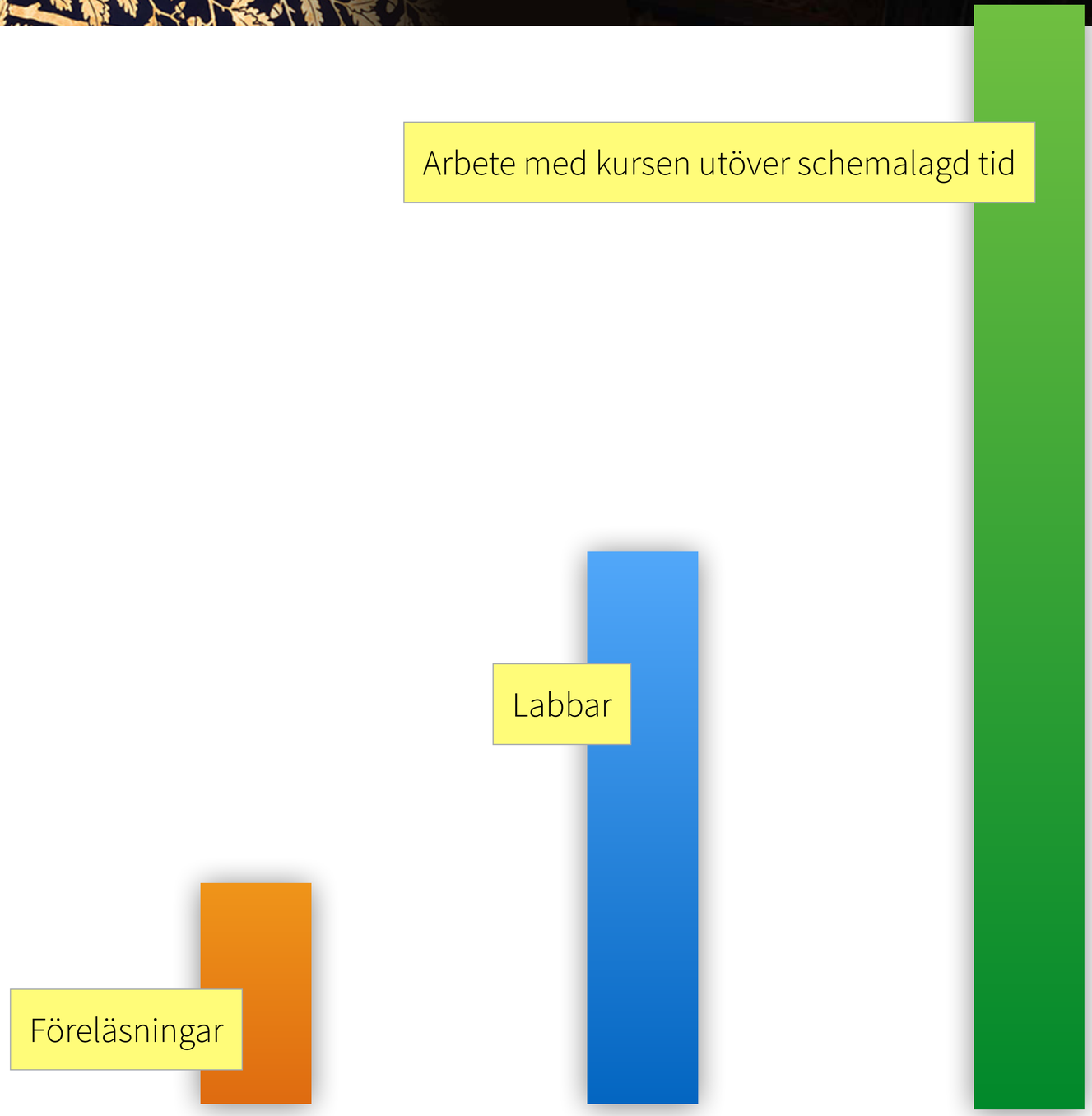
Kodprov kan ev. ske på Campus

Besked kommer i september — minst 3 kodprovstillfällen under kursens gång

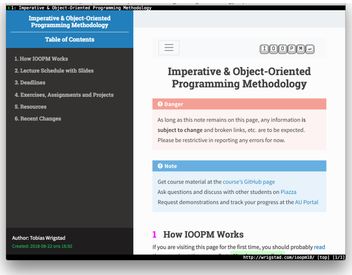
Slutseminarier och slutföreläsning i januari kan ev. ske på Campus

Besked kommer i december

En modell av IOOPM

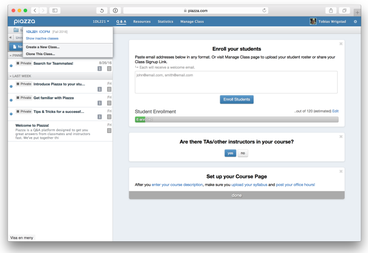


Undervisning under #2020



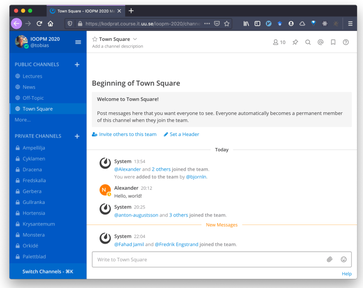
All information, kursmaterial

wrigstad.com/ioopm



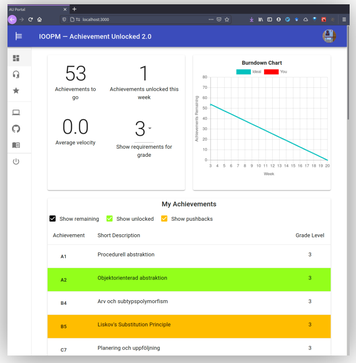
Diskussionsforum, handledning, enkäter, etc.

http://piazza.com/uu.se/fall2020/1dl221/home



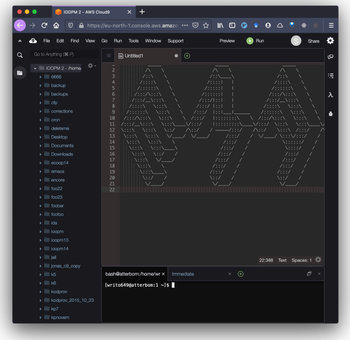
Matter Most

https://kodprat.course.it.uu.se/



AU Portal 2.0

https://ioopm.course.it.uu.se/



Cloud 9

https://ioopm.course.it.uu.se/

Zoom

GitHub

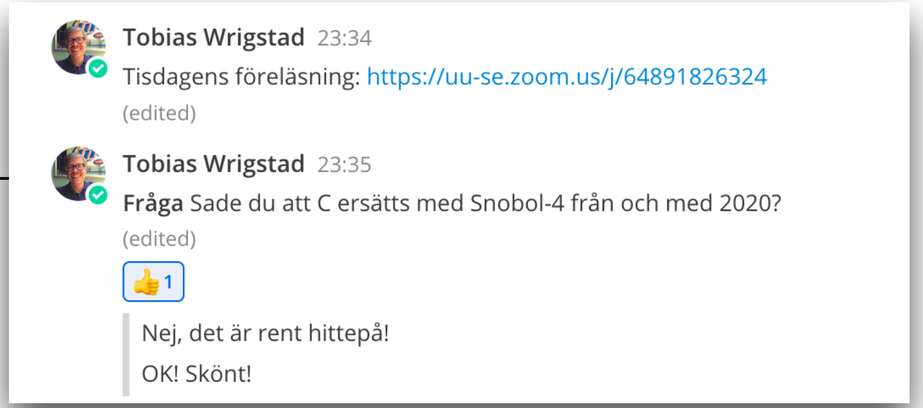
Föreläsningar

~ två gånger i veckan, 2 timmar

Öronmärkt tid för att kunna ta del av ev. inspelat material

Tar gärna emot frågor, både före, under och efter föreläsning

Använd MatterMost-kanalen #Lectures under och efter föreläsning



MatterMost-kanalen #Lectures

Liveström + inspelning är default

Labbar

Labbar är handlednings- och redovisningstillfällen

Förutom första 2 veckor på kursen som är snitslad bana

Handledning

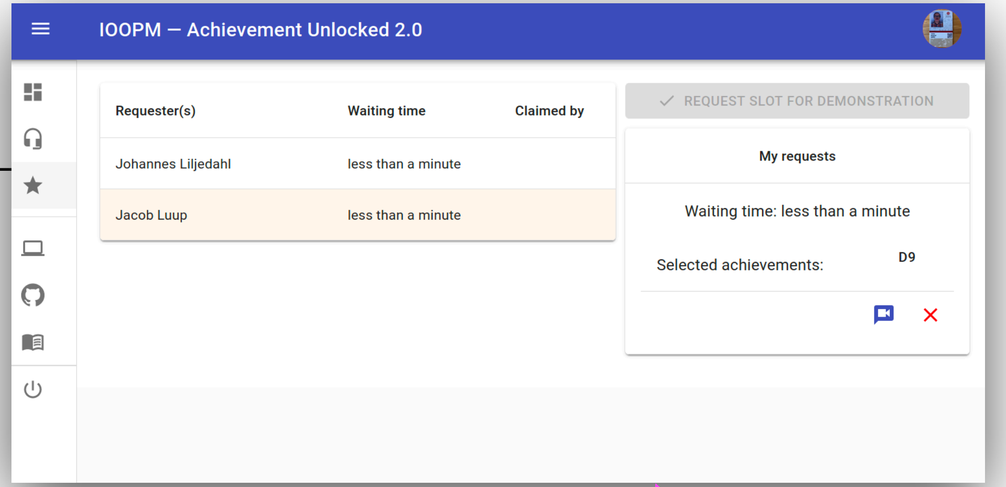
Hjälp med programmeringsproblem

Hjälp med kursen

Redovisning

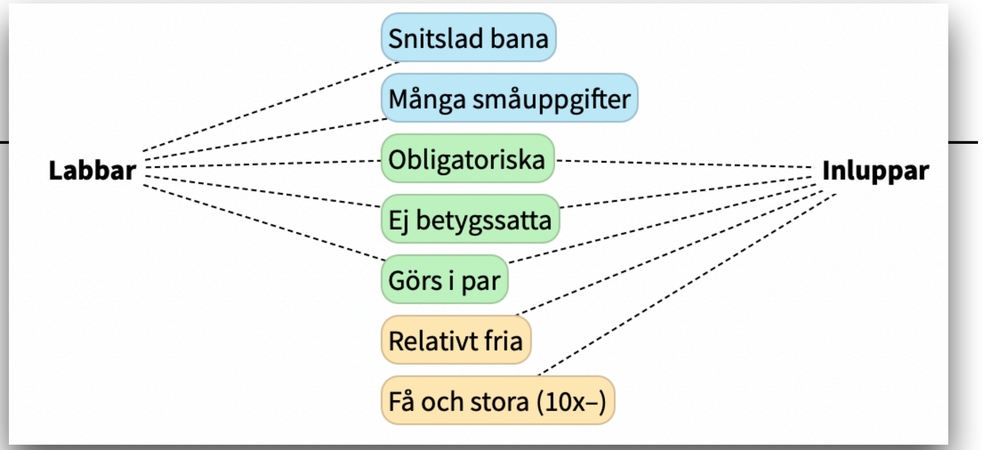
Demonstrera uppfyllelse av kursmålen

När du är redo, i den ordning du vill



AU-portalen är vår vän under labben

Uppgifter under kursen



Imperativ programmering [–mitten av oktober]

5 introlabbar

2 inlämningsuppgifter

Objektorienterad programmering [mitten oktober–slut november]

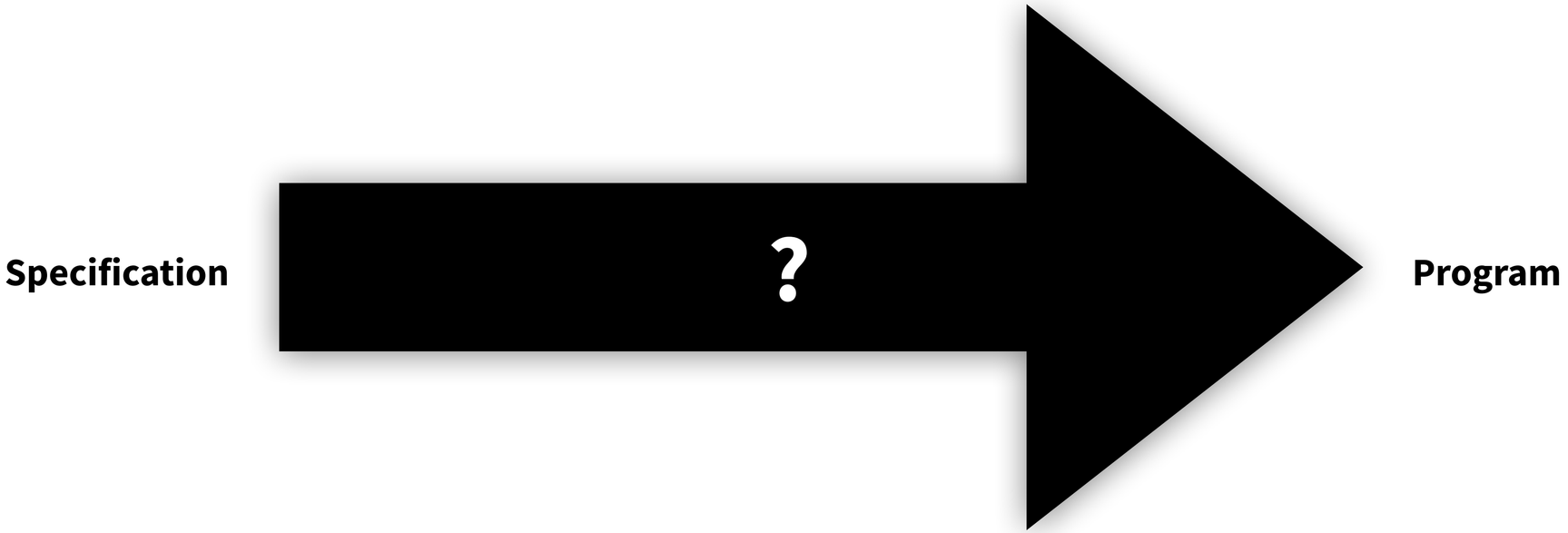
2 introlabbar

2 inlämningsuppgifter

Projektuppgift [december–]

Löses i mindre grupper

Syftet med inlämningsuppgifter

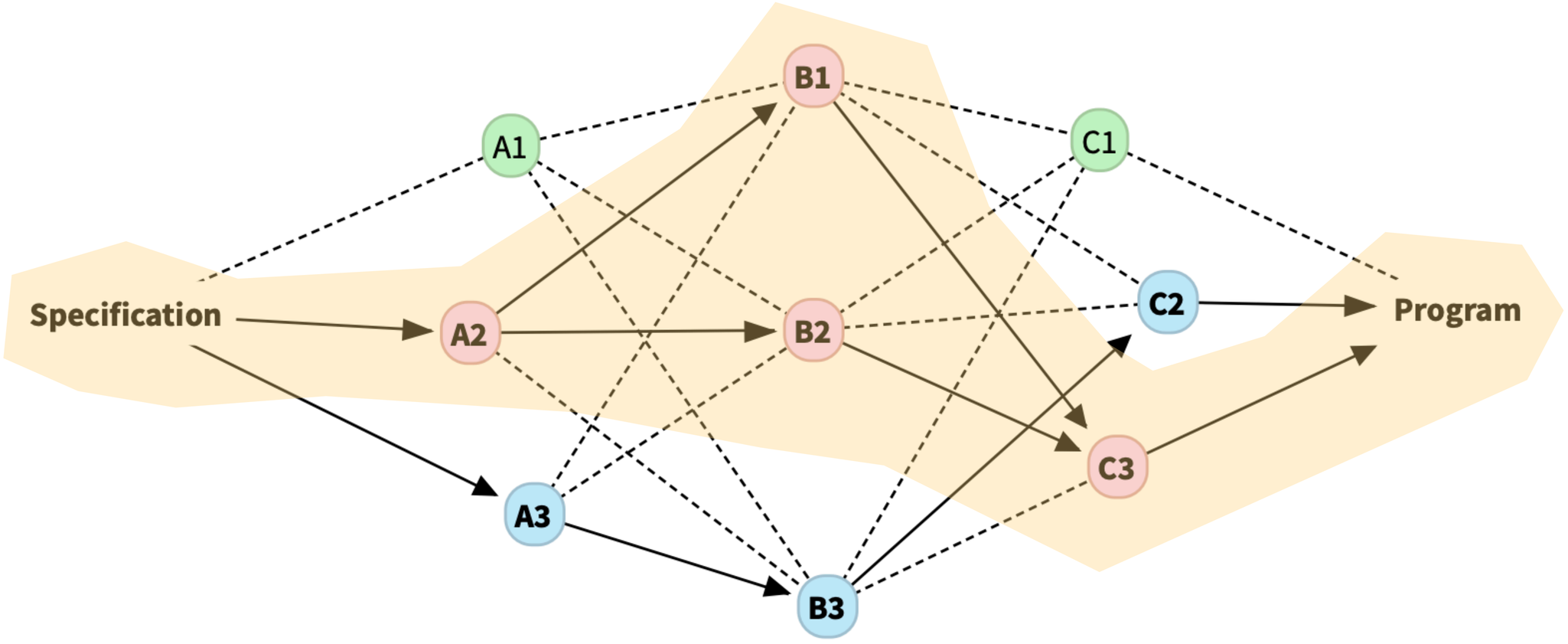


Studentens mål: ett körande program

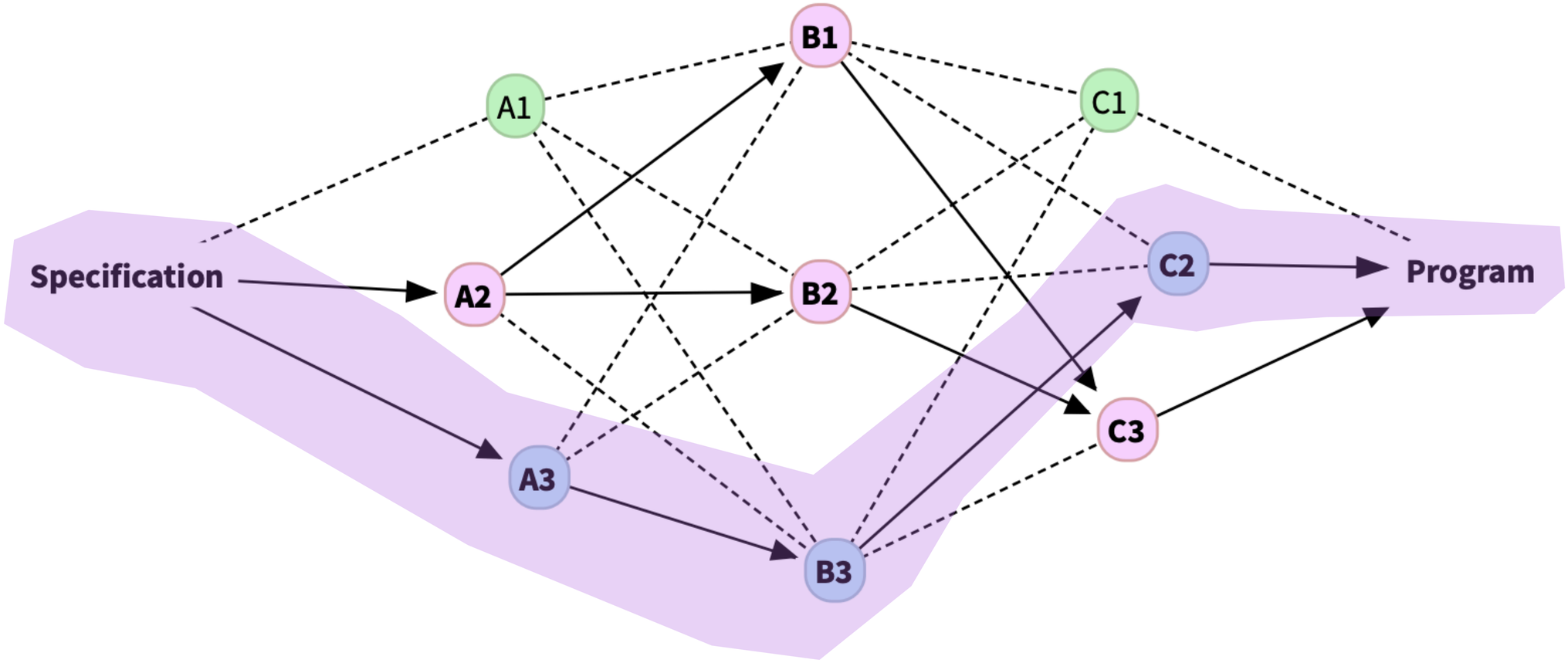
Lärarens mål:

- ✓ Polymorphism
- ✓ Memory management
- ✓ Defensive programming
- ✓ …

Ett program — olika lärdomar



Ett program — olika lärdomar



Achievements / Mastery Learning

Utgå från avsedda läromål

Studentens och lärarens mål är samma

Programmet är ett medel, inte ett mål

Utforska läromålen

Skaffa erfarenhet

Muskelminne

ATT är viktigare än NÄR, eller antal försök

Kursens två första veckor

Kom igång med C
(loopar, strängar, tal, arrayer)

Äntligen händer det något!
(I/O och funktioner)

Haskell, fast C
(Rekursion, Generalisering, Funktionspekare)

	Mån	Tis	Ons	Tors	Fre
V35		Föreläsning Labb	Labb		Föreläsning Labb
V36	Föreläsning Labb		Labb	Föreläsning Labb	

Data och strukturer
(pekare och struktar)

Miniprojekt
(Använd allt tillsammans)

Buffert
(Svårt att planera utan en!)

Intro till kursen!

Demo: Mattermost

https://kodprat.course.it.uu.se

Demo: AU-portalen

https://ioopm.course.it.uu.se

Demo: Cloud 9

Via AWS

Programmeringsparadigm

Funktionell

”Beräkning” sker genom exekvering av funktioner

Deklarativ

Fokus på *vad* som skall utföras

Imperativ

Fokus på de enskilda stegen

Objektorienterad

Fokus på de olika objekten i domänen och deras interaktion

Ortogonalt mot övriga paradigm ovan

Lästips: https://en.wikipedia.org/wiki/Comparison_of_programming_paradigms

Imperativ och objektorienterad programmering

”Mainstream”

Funktionell programmering används ”inbäddat” i imperativa språk

Imperativ programmering

Exemplifieras i kursen av C och Java

Objektorienterad programmering

Exemplifieras i kursen av Java

Aug 2020	Aug 2019	Change	Programming Language	Ratings	Change
1	2	⌃	C	16.98%	+1.83%
2	1	⌄	Java	14.43%	-1.60%
3	3		Python	9.69%	-0.33%
4	4		C++	6.84%	+0.78%
5	5		C#	4.68%	+0.83%
6	6		Visual Basic	4.66%	+0.97%
7	7		JavaScript	2.87%	+0.62%
8	20	⌃⌃	R	2.79%	+1.97%
9	8	⌄	PHP	2.24%	+0.17%
10	10		SQL	1.46%	-0.17%

https://www.tiobe.com/tiobe-index/

Kursens första del: imperativ programmering i C

C är ett maskinnära språk (för någon maskin iaf)

Resurskritiska applikationer, hårdvarunära programmering, effektivitet

Skapades ca 1969, användes för att implementera UNIX

Språk som kan ersätta

C: C++, D, Go, Java, Rust

På denna kurs använder vi C för att det inte gömmer komplexitet

(Och för att det är underbart och fantastiskt!)

Några skillnader mellan C och Haskell

C != Haskell (som de flesta stiftat bekantskap med under PKD)

C är imperativt, procedurellt och eager (”ivrigt”), Haskell är funktionellt och lazy

Språken tillhör olika syntaxfamiljer men den semantiska skillnaden är enorm!

C är manifest typat — alla variabler måste ges en explicit typ av programmeraren

C är svagt typat — vissa typomvandlingar görs automatiskt och okontrollerade brutala typomvandlingar tillåts

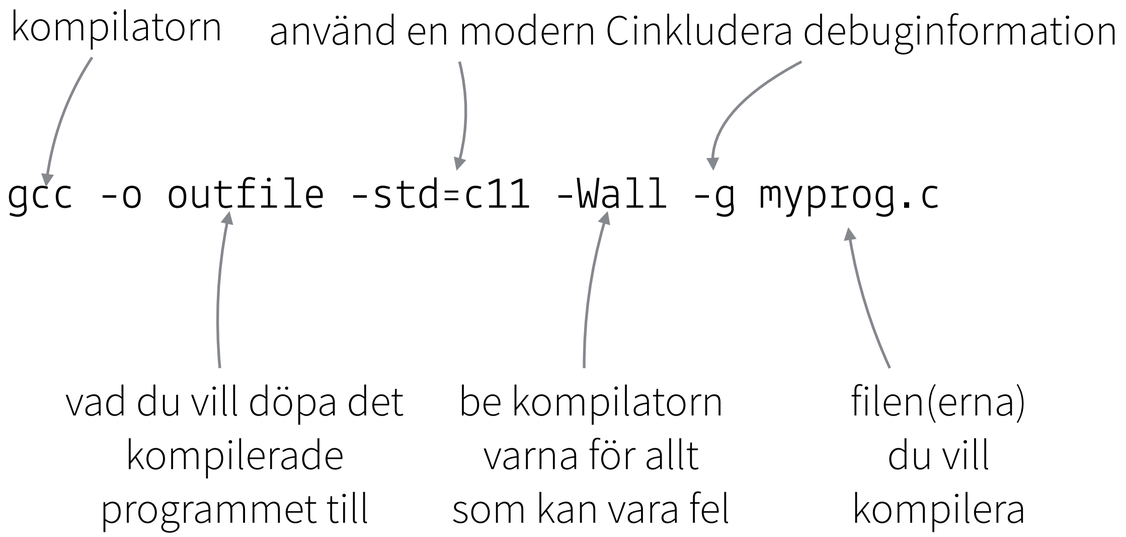
C är betydligt mer låg-nivå:

Du kan arbeta direkt med minnesadresser (pekare)

Minneshanteringen i C måste ofta göras explicit

Det görs vanligen ingen runtime-kontroll när C-program exekverar (vild adressering, arraygränser, odefinierade variabelvärden . . .)

Att kompilera ditt program



Kör ditt program: `./outfile`

Variabeldeklaration

Variabler i imperativa programspråk är i regel föränderliga

```
int x = 42;
printf("x: %d\n", x);
x += 42; // Från och med denna rad har x värdet 82
printf("x: %d\n", x);
```

Anatomi

```
variabeltyp variabelnamn = initialt värde;
```

Gör ALDRIG så här, och särskilt inte i C:

```
variabeltyp variabelnamn;
```

Variabelnamn är viktiga!

```
int foo = 42; // vad betyder foo?

int svaretPåMeningenMedLivet = 42;
```

Tumregler:

Om du inte tror att du kommer att förstå vad variabeln håller om 6 månader — döp den till något bättre

Konsekvens är bra

Ju tajtare scope, desto kortare namn

Väl mött på labben!