

# TDP004

Kursupplägg och intro till C++

Eric Ekström

Institutionen för datavetenskap

Handout version

# 1 Kursinformation

## 2 Grunder i C++

Hello World

Kompilering

Utströmmar

Variabler

Inströmmar

Villkor

# TDP004 - Kursmål

- använda grundläggande språkkonstruktioner och språkets standardbibliotek.
- använda designprinciper, metoder och tekniker som används inom objektorienterad programmering för att konstruera program som löser givna problem.

# TDP004 - Kursinnehåll

- 12 Föreläsningar
- 6 Laborationer
- 4 Lektioner
- 3 Dojos
- Dugga
- Tentamen

All information finns på kurshemsidan:

<https://www.ida.liu.se/~TDP004/>

## TDP004 - Examination

- **LAB2** - Laborationer (4hp)  
Alla 6 laborationer godkända + 33p genom kursen.
- **DAT2** - Datortentamen (4hp)  
Get slutbetyget i kursen.

## TDP004 - Laborationer

- Registrera er på WebReg (deadline idag)!
- 6 laborationer (0-5)
- Redovisning, kodinlämning och komplettering
- Laborationerna kräver mycket tid (första mjuka deadline redan imorgon)
- Hård deadline: sista passet
- Ungefär ett labborationspass per dag
- Mjuka deadlines ger bonus på tentan  
+0.5p på del 2

# TDP004 - Lektioner & Dojos

## Lektion:

- Föreberedelseuppgifter (1 poäng per löst uppgift)
- Maximalt 8 poäng per lektion
- Inga poäng om man kommer sent
- Uppgifter redovisas på tavlan under första timmen
- Problemlösning i helgrupp under andra timmen

## Dojo:

- Första timmen som lektion (2 poäng per uppgift)
- Andra timmen löses ett dojoproblem
- Maximalt 11 poäng per dojo

# TDP004 - Tentamen & Dugga

## Tentamen:

- 2 delar, 3 uppgifter på varje
- I del 1 måste alla uppgifter klaras
- I del 2 krävs ett visst antal poäng

## Dugga:

- Del 1 från tentamen + 1 uppgift från del 2
- Går att nå betyg 3 på duggan
- Avklarad del 1 kan tillgodoräknas på tentamen



## TDP004 - Tempo

- Högt tempo i kursen
- Går endast i November
- förväntad arbetstid är 40 timmar i veckan
- C++ kan vara svårt, ställ alltid frågor!
- Var källkritisk mot info på nätet

## TDP004 - Personal

- Examinator - Pontus Haglund (pontus.haglund@liu.se)
- Kursledare - Eric Ekström (eric.ekstrom@liu.se)
- Assistenten - Alexander, Alex, Anton, Aron, Malte, Oliver

Kontaktuppgifter på kurshemsidan

Prata med oss!

# TDP004 - Återkoppling

## Förra årets Evaluate

- Väldigt högt tempo i kursen.
- Svårt att hinna förstå innehållet på djupet
- Ojämn feedback på laborationer
- 4 veckor är för lite!

## TDP004 - Ändringar i kursen

- Bättre placerade föreläsningar
- Uppdaterad lab2  
*Nya instruktioner och krav. I stora drag samma uppgift*
- Ny lab 4  
*Helt ny uppgift. Mindre förvirrande?  
Starkare koppling till TDP005.*
- Template-labb borttagen  
*Mer tid till lab 4 & 5*
- Ny föreläsare!

1 Kursinformation

2 Grunder i C++

Hello World

Kompilering

Utströmmar

Variabler

Inströmmar

Villkor

# Ett enkelt program i C++

## Python

```
print("Hello World")
```

## C++

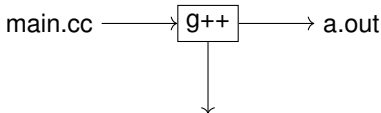
```
#include <iostream>

int main()
{
    std::cout << "Hello World" << std::endl;
    return 0;
}
```

- Instruktioner avslutas med ;
- Alla instruktioner skrivs inom main()
- Indentering i Python == Måsvingar i C++
- #include <iostream> låter oss använda std::cout

# Kompilering

C++ är ett kompilerat språk. Vi kommer använda kompilatorn g++.



```
main.cc: In function 'int main()':  
main.cc:8:9: error: cannot convert 'string' to 'int' in assignment  
      8 |         x = num;  
        |         ^  
        |         string
```

- En kompilator är ett speciellt program
- Den översätter källkodsfiler till exekverbara filer
- En exekverbar fil innehåller maskinkod som datorn kan köra

# Kompilering

Vi kompilerar koden från terminalen

```
$ g++ main.cc  
$ ls  
a.out main.cc
```

Om inget skrivs ut har kompileringen lyckats

Programmet kan då köras med

```
$ ./a.out
```



# Kompilering

Vi vill kompilera med extra kontroller

```
$ g++ -std=c++17 -Wall -Wextra -Wpedantic -Werror main.cc
```

Långt att skriva? Skapa ett alias!

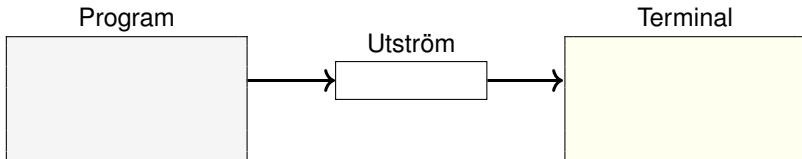
```
alias w++17='-std=c++17 -Wall -Wextra -Wpedantic'  
alias e++17='-std=c++17 -Wall -Wextra -Wpedantic -Werror'
```

Dessa två kommer finnas på tentan.

- `-Wall -Wextra` lägger till fler varningar från kompilatorn, vilket hjälper oss att programmera bättre.
- `-Werror` ändrar alla varningar till fel.
- `-std=c++17` låter oss välja versionen av C++. C++17 är den vi använder

# Utströmmar

Det finns en utström från programmet till terminalen som den kan kommunicera genom

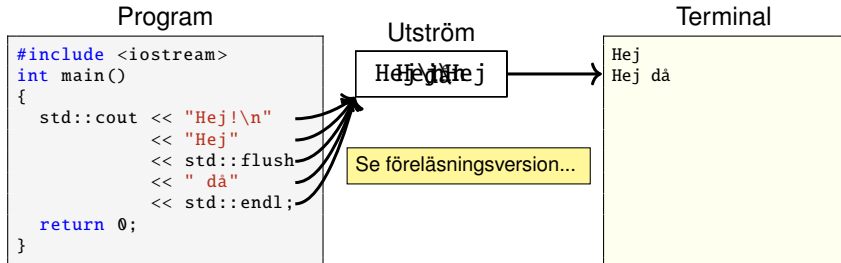


# Utströmmar

- `std::cout` används för att kommunicera med utströmmen.
- `operator<<` skickar data till strömmen.
- `std::flush` tömmer strömmen till terminalen.
- `std::endl` lägger en nyrad i strömmen och gör en `std::flush`.

Ex. `std::cout << "Hello World" << std::endl;`

# Utströmmar



# Utströmmar

Vi kan formatera utskriften m.h.a funktioner från `std::ostream` och `std::omanip`.

- `fixed` - Använd decimalpunkt och N=6 decimaler
- `setprecision(N)` - Använd N decimaler
- `setw(N)` - Använd minst N tecken för nästa utskrift
- `setfill(C)` - Använd tecknet C som utfyllnadstecken
- `left` - Vänsterjustera utskriften (inom `setw`)
- `right` - Högerjustera utskriften (inom `setw`)

# Utströmmar

Vi kan formatera utskriften m.h.a funktioner från `std::iostream` och `std::iomanip`.

```
#include <iostream>
#include <iomanip>

int main()
{
    std::cout << std::setw(6)
               << std::setfill('.')
               << 10 << std::endl;

    return 0;
}
```

```
....10
```

# Ett \*lite\* mer komplicerat program i C++

## Python

```
number = input("Vilket heltal är bäst? ")

if number == 76:
    print("Korrekt")
else:
    print("Fel")
```

## C++

```
#include <iostream>
int main()
{
    std::cout << "Vilket heltal är bäst? "
    << std::flush;
    int number {};
    std::cin >> number;

    if (number == 76)
    {
        std::cout << "Korrekt" << std::endl;
    }
    else
    {
        std::cout << "Fel" << std::endl;
    }
    return 0;
}
```

# Datatyper och Variabler

För att spara data under programkörning behöver vi lista ut ett par saker:

- Var i minnet placerar vi data?
- Hur hittar vi den datan senare?
- Hur mycket plats behövs i minnet?
- Hur tolkas datan?



# Datatyper och Variabler

Variabler i C++ har alltid

- ett unikt namn (a-Z, 0-9, \_)
- ett värde (även om vi inte har angett något!)
- en specifik datatyp. De vanligaste är

|               |               |                 |            |
|---------------|---------------|-----------------|------------|
| int           | Heltal        | -7              | 2147483647 |
| float, double | Flyttal       | 14.7            | 0.99f      |
| char          | Tecken        | 'j'             | '\n'       |
| std::string   | Text          | "Hej jag heter" |            |
| bool          | Sanningsvärde | true            | false      |

En variabel i C++ kan aldrig byta typ

# Datatyper och Variabler

```
#include <string>

int main()
{
    int x { 4 };
    char c { 'e' };
    float const pi { 3.14 };
    std::string erics_passwd { "c++ärbäst" };

    x = 10;
    pi = 9;

    return 0;
}
```

**KOMPILERAR EJ!**

- Datatypen måste alltid anges när vi skapar variabler.
- Vi initialiserar variabler med {}

int  
x: 10

char  
c: 'e'

float  
pi: 3.14

# Datayper och Variabler - std::string

Strängar i C++ är inte en inbyggd datatyp.

- `std::string` är definierad i `<string>` som måste inkluderas.
- Strängar har trevlig inbyggd funktionalitet vi kan utnyttja.

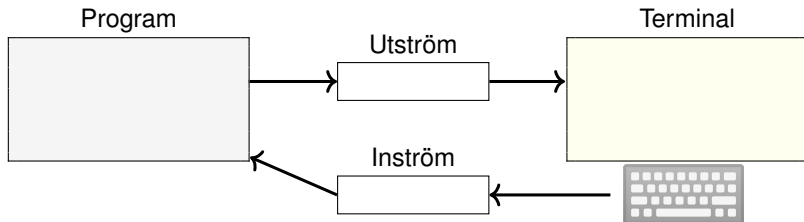
```
#include <iostream>
#include <string>

int main()
{
    std::string str {"hello"};
    std::cout << str      << '\n'
              << str.size() << '\n'
              << str.front() << std::endl;

    return 0;
}
```

# Inströmmar

Det finns även en inström från terminalen till programmet som vi kan kommunicera genom.



# Inströmmar

- `std::cin` kan vi använda för att kommunicera med inströmmen.
- med operator `>>` läser vi data från strömmen.
- Vad kan stå på höger sida om `>>`?

Ex. `std::cin >> x;`

# Inströmmar

temp: double  
24.5      stl: int  
20

## Program

```
#include <iostream>
int main()
{
    std::cout << "Mata in
    rumstemperatur
    och rumstl:"
    << std::flush;
    double temp {};
    int stl {};
    std::cin >> temp;
    std::cin >> stl;
    return 0;
}
```

## Utström

Mata in rumstemperatur  
och rumstl:

## Terminal

Mata in rumstemperatur  
och rumstl: 24.5 20

## Inström

24.5 20\n\n



# Villkor

```
int main()
{
    std::cout << "Vad är temperaturen? "; // Ingen flush??
    int temp {};
    std::cin >> temp;
    if (temp >= 40)
    {
        std::cout << "Är du i en bastu?" << std::endl;
    }
    return 0;
}
```

# Villkor

```
int main()
{
    std::cout << "Vad är temperaturen? "; // Ingen flush??
    int temp {};
    std::cin >> temp;
    if (temp >= 40)
    {
        std::cout << "Är du i en bastu?" << std::endl;
    }
    else
    {
        std::cout << "Låter alldeles lagom" << std::endl;
    }
    return 0;
}
```



# Villkor

```
int main()
{
    std::cout << "Vad är temperaturen? "; // Ingen flush??
    int temp {};
    std::cin >> temp;
    if (temp >= 40)
    {
        std::cout << "Är du i en bastu?" << std::endl;
    }
    else if (temp < 15)
    {
        std::cout << "Maila studentbostäder" << std::endl;
    }
    else
    {
        std::cout << "Låter alldeles lagom" << std::endl;
    }
    return 0;
}
```

# Misslyckade inläsningar

Kan vi kontrollera om en inläsning gått bra?

```
int temp {};  
if ( std::cin >> temp )  
{  
    std::cout << "Du matade in temperaturen " << temp << std::endl;  
}  
else  
{  
    std::cout << "Tyvärr, du matade inte in ett giltigt heltal"  
               << std::endl;  
}
```

Vad gör vi nu...

## Mer om inläsning

- `std::cin.clear();`  
Nollställer felflaggan så vi kan fortsätta läsa.
- `std::cin.ignore(5, '\n');`  
Rensar strömmen på max 5 tecken, eller stannar vid första nyradstecknet.
- `std::getline(std::cin, str, '\n');`  
Läser en hel rad tecken (fram till nyradstecknet) och sparar dessa i variabeln `str`.

## Vad har vi gått igenom?

- Kompilering
- Utmatning
- Variabler
- Inmatning
- Villkor

# Resurser och Informationssökning

Hur hittar man information om C++?

- Givet material i kursen (se kurshemsidan)
- Rekommenderad bok om C++
- Fråga kurspersonal (i sal eller på mail)
- Sök på `cppreference.com`

Vanskliga alternativ (ditt mål är att *lära* dig C++, *inte* att producera lablösning)

- Fråga kompisar
- Google? chatGPT?

Om du ställer frågan “varför...?” inte “hur...?”

# Finn fem fel!

```
#include <string>
int main() {
    std::cout << "Skriv in två heltal: " << std::endl;
    int x; int y {};
    std::cin << x << y;
    if ( std::cin )
    {
        if x < y
            std::cout << x << " är mindre än" << y;
    }
    else
    {
        std::cout << "inläsningen misslyckades" << std::endl;
    }
}
```

## Finn \*minst nio\* fel!

- string inkluderas istället för iostream
- x deklarerar utan måsvingar
- pilarna i strömoperatörn för std::cin går åt fel håll
- if-satsen saknar parenteser och måsvingar
- det saknas en std::flush på andra utskriften
- två satser ligger på samma rad
- det saknas en return 0 i slutet av main
- koden är för kompakt skriven
- med mera...

## Avslutande tankar

- Anmäl er i WebReg direkt
- Första deadline (med redovisning) redan imorgon
- Kursen har ett högt tempo
- Se labbpass som tid för frågor och diskussion
- Kom förberedd till alla pass (även labb)



[www.ida.liu.se/~TDP004/](http://www.ida.liu.se/~TDP004/)