# TDIU16

## Föreläsning 0

Filip Strömbäck, Klas Arvidsson

LINKÖPING UNIVERSITY

LINKÖPING UNIVERSITY

# Resurser

- Kurshemsida: `http://www.ida.liu.se/~TDIU16/`
- Litteratur: *Operating System Concepts* av Silbertschatz, Galvin, Gagne

| | |
|---|---|
| Examinator | Filip Strömbäck (även KL) |
| Kursledare | Klas Arvidsson |
| Assistenter | Dag Jönsson |
| | Tobias Elfstrand |
| | Malte Nilsson |
| | Mattias Burman |
| | Mattias Karlsson |
| Administratör | Annelie Almquist |

# Mål med kursen

Tillämpa teori från TDIU11:

- Hur ser en ”vanlig” dator ut?
- Hur körs program?
- Vilka viktiga resurser finns för programkörning?
- Varför finns operativsystem?
- Vad gör operativsystemet? När körs det?

# Mål med kursen

Nytt innehåll:

- Multitrådning
- Synkronisering
- Programmering i C

Praktisk erfarenhet:

- Programmera i större system
- Läsa, förstå och modifiera kod som andra har skrivit
- Felsökning

# Mål med kursen

# FÖ, SE, LA

- Föreläsningar
  - Slides är *inte* avsedda att fungera självständigt: Närvara
  - Ställ frågor
  - Svara (gärna fel) på ställda frågor
- Seminarier
  - Bidra till problemlösning
  - Ställ ännu mer frågor
- Lab
  - Arbeta så långt som möjligt på egen hand
  - Ställ frågor/demonstrera på labpassen
  - Det är ofta hög belastning på passen

# Examination

DAT1 Skriftlig tentamen med dator, 1hp (U, 3, 4, 5)

LAB1 Laborationer, 3hp (U, G)

- Ett ofärdigt operativsystem (Pintos) skrivet i C
- 100h skall användas
- Arbeta med labbarna i par
- Redovisas muntligt under labpass
- Mjuka deadlines ger bonus
- En hård deadline (slutet av kursen)
- Eget arbete utöver schemalagd tid krävs

# Labanmälan

- Arbete sker parvis som vanligt.
- Tre klasser efter program:
  1. DI2, EL2, A
  2. DI2, EL2, B
  3. IP2
- Labanmälan öppen tom. 27/3, anmäl er snarast!

# Planering

| Vecka | Fö/Se | Lab |
|---|---|---|
| 13 | Fö: C + Syscall (+påsk) | C[1], intro |
| 14 | - | systemanrop |
| 15 | Fö + Se: Semaforen | Processhantering |
| 16 | Fö: Lås, cond | Processhantering |
| 17 | Fö: Deadlock | Processhantering |
| 18 | - | Synkronisering |
| 19 | Fö: Låsimplementation | Synkronisering, säkerhet |
| 20 | Se: Deadlock + tenta | Säkerhet |
| 21 | - | Säkerhet |

[1] lämpligt att demonstrera första passet

# Tid i kursen

- Kursen är 4hp $\approx$ 100h
- Du kommer att behöva alla 100h. Planera din tid så att du pluggar varje vecka. Lämna inte alla 100h till dagarna före tenta.
- Det finns fristående uppgifter ni kan börja med ifall ni kör fast.

# Kommentarer från tidigare år

- Intressant innehåll, men kursen tar mycket tid
- Mycket information, ibland svårt att veta vad som är ”rätt”
- Får ibland vänta länge på labpassen
- Bra med visualiseringsverktyget

Förändringar:

- Verktyg som hjälper att hitta minnesläckor
- ”Banker’s algorithm” borttaget
- Bättre koppling till TDIU11

LINKÖPING UNIVERSITY

# C och C++ – vad är skillnaden?

I allmänhet är C ett mycket tunnare lager ovanpå hårdvaran än C++ (byggt för att skriva UNIX)

Det mesta i C är giltigt i C++, men det finns små skillnader.

Notera att `struct` och `union` har separata namnrymder i C. Går att komma runt med `typedef`.

Notera: `int foo();` $\not\Longleftrightarrow$ `int foo(void);`

# C och C++ – vad är skillnaden?

| C++ | C |
|---|---|
| `cout` | `printf` |
| klasser | `struct` |
| referenser | pekare |
| `vector`, `string` | arrayer + biblioteksfunktioner |
| `private` | definiera i c-fil, `static` |
| statisk datamedlem | `extern` |
| `new`, `delete` | `malloc`, `free` |
| virtuella funktioner | funktionspekare |
| klassmallar | `void *`, makron (se Pintos lista) |
| exceptions | returvärden |
| överlagring | **nej** |

# Utmatning

```
int main() {
    int a = 1;
    unsigned int b = 2;
    float c = 3.5f;
    double d = 5.8;
    char e = 'A';
    int *f = &a;
    const char *g = "hello";

    cout << a << b << c << d
         << e << f << g << endl;
}
```

# Utmatning

```
int main() {
    int a = 1;
    unsigned int b = 2;
    float c = 3.5f;
    double d = 5.8;
    char e = 'A';
    int *f = &a;
    const char *g = "hello";

    printf("%d %u %f %f %c %p %s\n",
           a, b, c, d, e, f, g);
}
```

# Utmatning

```c
int main() {
    int a = 10;
    const char *b = "hello";

    printf("%6d\n%06d\n%6s\n", a, a, b);
    // Ger:
    //     10
    // 000010
    //  hello
}
```

# Exempel på pekararitmetik

I visualiseringsverktyget: *File* → *Open example...*

- `locals.c`
- `struct.c`
- `array.c`
- `array_malloc.c`
- `struct_array.c`

# Första veckan

Första labpasset

- Installera Pintos
- GDB
- Påbörja associativ container (används senare)

Sedan:

- Systemanrop: `exit`, `halt`

Filip Strömbäck, Klas Arvidsson

www.liu.se