

# Hva vi gjennomgår i dag

- Første time: vandring gjennom læreboken
- Andre time: hva kreves utover å lese læreboken
- Hva vil det si å kunne programmere?
- Løsning og diskusjon av to konkrete eksempler på oppgaver
- Gode råd for eksamen

# Programmering er en ferdighet!

- Målet er å kunne anvende programmering til å løse problemer
- Å forstå de ulike begrepene (som if og while) er en forutsetning, men ikke tilstrekkelig
- Å lese boka er ikke nok - man må også trene på å løse mange ulike problemer
- Alle skriftlige læremidler kan tas med på eksamen - ferdigheten må man ha opparbeidet selv

# Hva vil det si å kunne programmere?

## Oppsummering av

## INF1000

Programmering som en ferdighet

- Mer spesifikt:
- Hva er det dere forventes å kunne etter INF1000?
- Eller sagt på annen måte:
- Hva er det dere må kunne for å gjøre det godt på eksamen?

# Eksempel på oppgave

(eksamen INF1000, høst 2013)

## Et annet mulig svar

- Du skal nå skrive en metode som har tre parametre av typen *double* og som returnerer en verdi av typen *double*. Metoden skal finne den minste av de tre parameterverdiene og returnere denne. Hvis metoden heter *minst*, så skal f.eks. setningen `double v = minst(3, 1.3, 2.6);` føre til at variabelen *v* blir tilordnet verdien 1.3.
- Prøv selv å skrive et komplett svar på denne (5 min)!

```
double minst(double a, double b, double c){
    double svar=0;
    if (a <= b && a <= c){
        svar = a;
    }
    if (b <= a && b <= c){
        svar = b;
    }
    if (c <= a && c <= b){
        svar = c;
    }
    return(svar);
}
```

## Programmeringens natur

## Et mulig svar

- Fra første time:
- Software development happens in your head, not in an editor" (Andy Hunt)
- "Programming is all about problem solving. It requires creativity, ingenuity, and invention"

```
double minst(double a, double b, double c){
    double svar=a;
    if (b < svar){
        svar = b;
    }
    if (c < svar){
        svar = c;
    }
    return(svar);
}
```

# En noe vanskelige oppgave

- Vi har gitt en fil med navn "bokstaver.txt", som har én bokstav per linje. Skriv et program som leser gjennom denne filen og
  - a) teller antall A-er i filen
  - b) regner ut andelen av bokstavene i filen som er A (antall A dividert med antall bokstaver totalt).
- Prøv å skrive ned koden for dette på papir (5 minutt)

## Et tredje mulig svar

```
double minst(double a, double b, double c){
    double svar;
    if (a <= b && a <= c){
        svar = a;
    }
    else if (b <= c){
        svar = b;
    }
    else {
        svar = c;
    }
    return(svar);
}
```

# Hva som krevdes for å løse oppgaven

- Se behovet for en løkke (while) for å lese bokstaver
- Kunne lese fra fil (her kan du slå opp i notat)
- Se behovet for å sjekke hva en bokstav er (if)
  - Equals er en mindre detalj
- Se behovet for en A-teller som begynner på 0
- Oppg b: se at man trenger en ekstra teller for alle bokstaver
- *(Man kunne også løst ved å lese inn i array. Det ville vært mer arbeid, men ville også løst problemet.)*

## Et mulig svar

```
import java.util.Scanner;
import java.io.File;

public class U9Eksempel_ferdig{
    public static void main(String[] args) throws Exception {
        String bokstav;
        int antallA = 0;
        int antallBokstaver=0;
        String filNavn = "bokstaver.txt";
        Scanner fil = new Scanner(new File(filNavn));

        while (fil.hasNextLine()){
            bokstav = fil.nextLine();
            antallBokstaver +=1;
            if (bokstav.equals("a")){
                antallA += 1;
            }
        }
        System.out.println("Antall A: " + antallA );
        System.out.println("Andel A: " + 1.0*antallA/antallBokstaver);
    }
}
```