Some success stories of **automated reasoning in industry**:

- **Microsoft**: static driver verification;
- **Intel**: hardware verification;
- **Airbus, Dassault Aviation**: safety and runtime verification.

```
void upd( int *s, int *d, int n )
{
  int i=0;

  while (i<n) {

     d[ i ] = s[ i ] + 1;
     i++;}
}
```

```
void upd( int *s, int *d, int n )
{
  int i=0;

  while (i<n) {

     d[ i ] = s[ i ] + 1;
     i++;}
}
```

- uses **array s**

- updates **array d**

- each element of **d** is greater than the corresponding element in **s**

@requires n>0

```
void upd( int *s, int *d, int n )
{
  int i=0;

  while (i<n) {

     d[ i ] = s[ i ] + 1;
     i++;}
}
```

@ensures **(for each p) (0≤p<i ⇒ d[p]>s[p])**

- uses **array s**

- updates **array d**

- each element of **d** is greater than the corresponding element in **s**

How can one prove such program properties?

@requires n>0

```
void upd( int *s, int *d, int n )
{
  int i=0;

  while (i<n) {

     d[ i ] = s[ i ] + 1;
     i++;}
}
```

@ensures **(for each p) (0≤p<i ⇒ d[p]>s[p])**

- uses **array s**

- updates **array d**

- each element of **d** is greater than the corresponding element in **s**

How can one prove such program properties?

@requires n>0

```
void upd( int *s, int *d, int n )
{
  int i=0;

  while (i<n) {

     d[ i ] = s[ i ] + 1;
     i++;}
}
```

@ensures **(for each p) (0≤p<i ⇒ d[p]>s[p])**

- prove properties of **integers**, including those about + and >

- prove properties of **arrays**

- prove **quantified** properties

Course on theory and practice of:

1. SAT solving: propositional logic (only bits)
   Tool: MiniSAT
2. Theory reasoning: arrays, integers, uninterpreted functions
   Tool: Z3 and Vampire
3. Reasoning in combination of theories
   Tool: Z3

http://www.cse.chalmers.se/~laurako/links/ARV.html

- First ARV lecture: October 28, 10:15am-12:00, EDIT 3364
- ARV lectures (Mondays and Thursdays) and exercises (Fridays)
- 7.5 ECTS
- sign-up to the course: send email to laura.kovacs@chalmers.se