A Formalization of the C99 Standard in HOL, Isabelle and Coq

Robbert Krebbers
Joint work with Freek Wiedijk

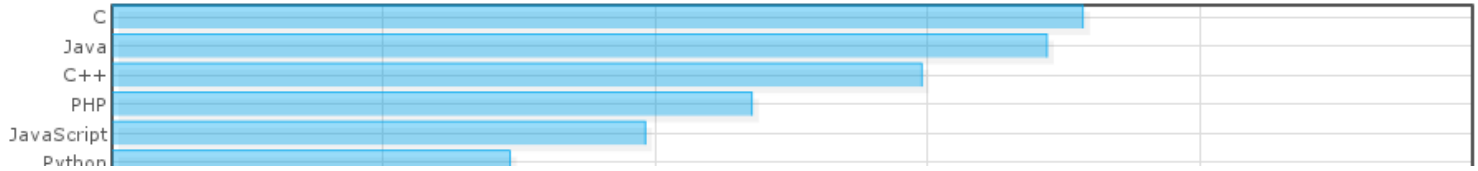
Radboud University Nijmegen

July 19, 2011 @ CICM, Bertinoro, Italy

The C programming language

Among the two currently most used languages:

- LangPop.com - Programming Language Popularity



- TIOBE Software - Programming Community index

Position Jun 2011	Position Jun 2010	Delta in Position	Programming Language	Ratings Jun 2011	Delta Jun 2010	Status
1	2	↑	Java	18.580%	+0.62%	A
2	1	↓	C	16.278%	-1.91%	A
3	3	=	C++	9.830%	-0.55%	A
4	6	↑↑	C#	6.844%	+2.06%	A
5	4	↓	PHP	6.602%	-2.47%	A

Used for the smallest microcontroller to the largest supercomputer.

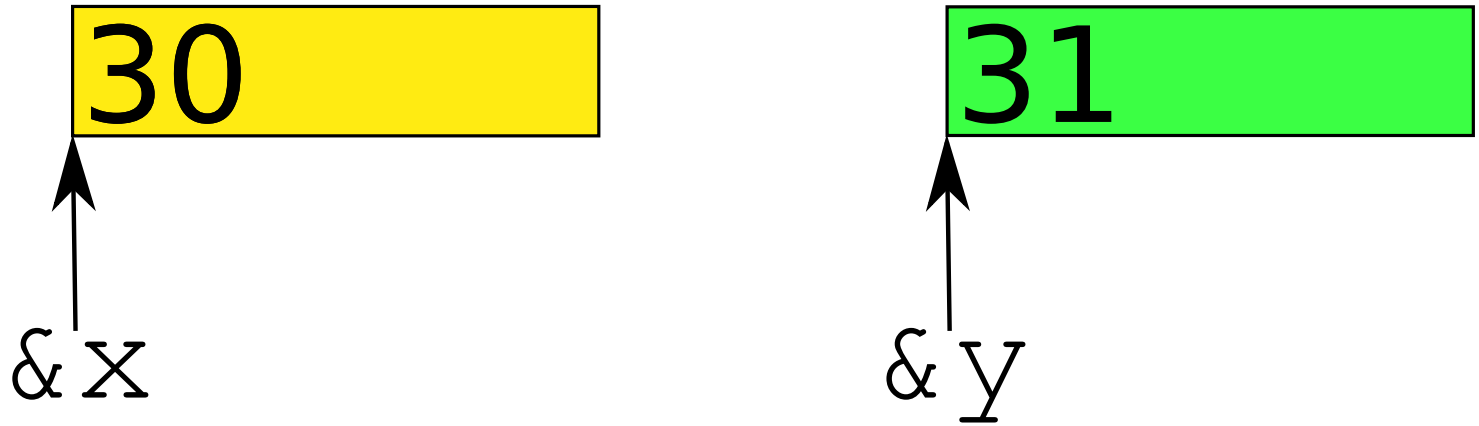
The C99 standard

The official description issued by ANSI and ISO:

- Written in English
- No mathematically precise formalism
- Incomplete and ambiguous

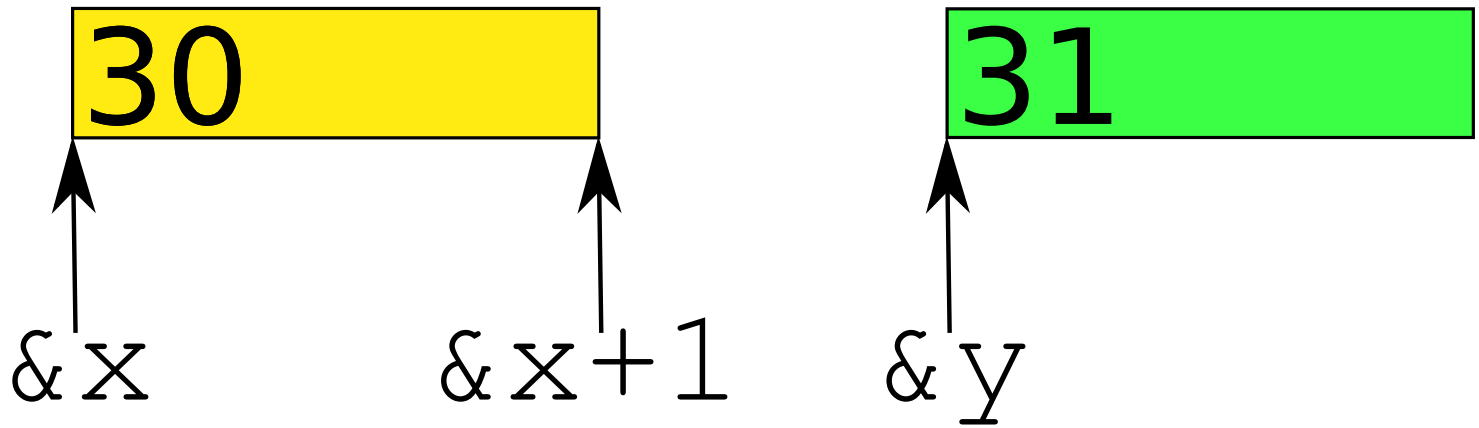
Example

```
int x = 30, y = 31;
```



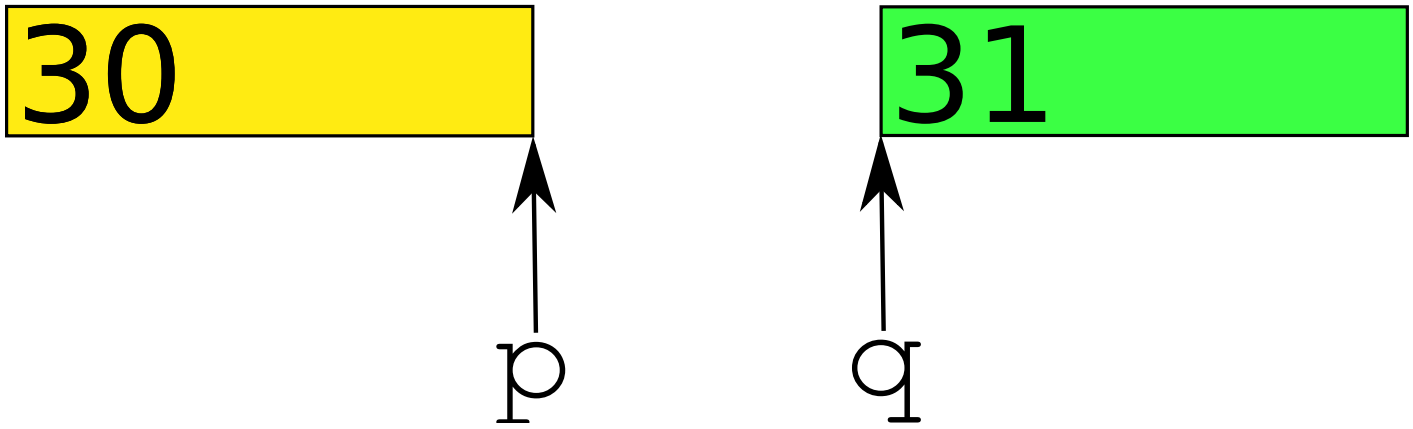
Example

```
int x = 30, y = 31;
```



Example

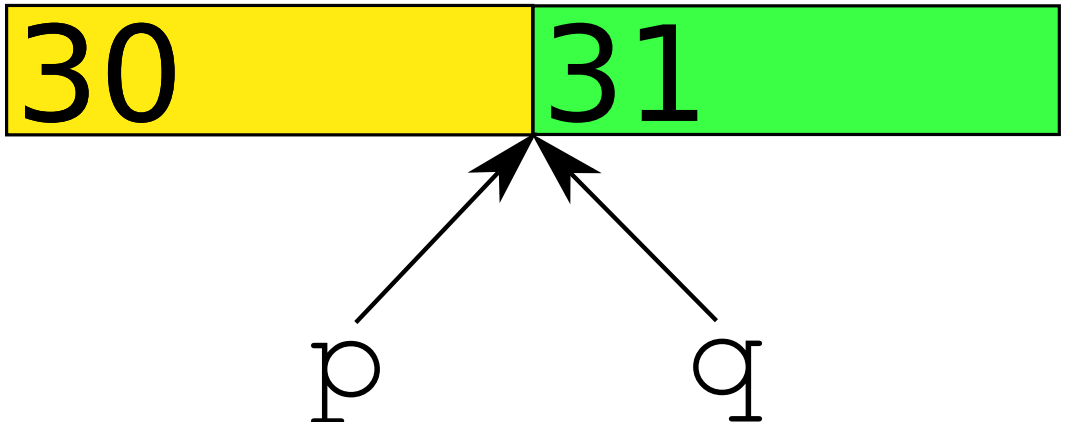
```
int x = 30, y = 31;
int *p = &x + 1, *q = &y;
```



Example

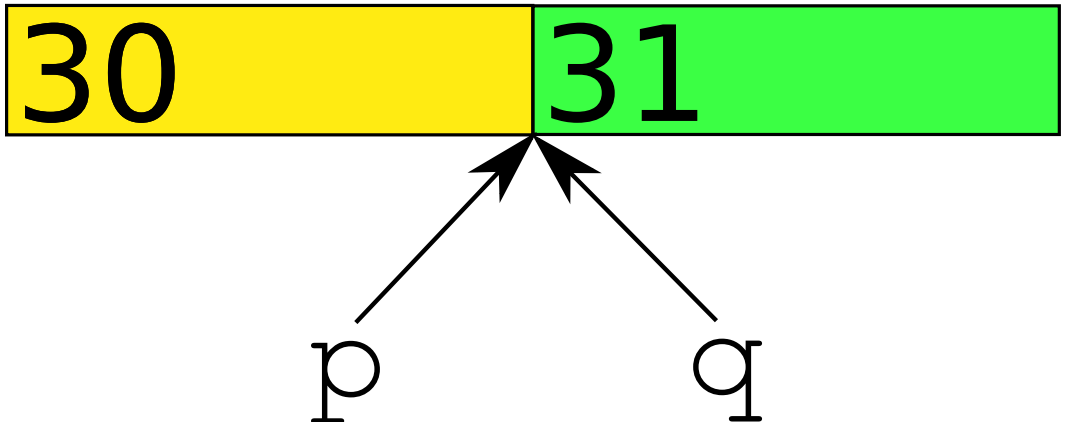
```
int x = 30, y = 31;
int *p = &x + 1, *q = &y;
if (memcmp(&p, &q, sizeof(p)) == 0) {

}
```



Example

```
int x = 30, y = 31;
int *p = &x + 1, *q = &y;
if (memcmp(&p, &q, sizeof(p)) == 0) {
  printf("%d\n", *p);
}
```



Example

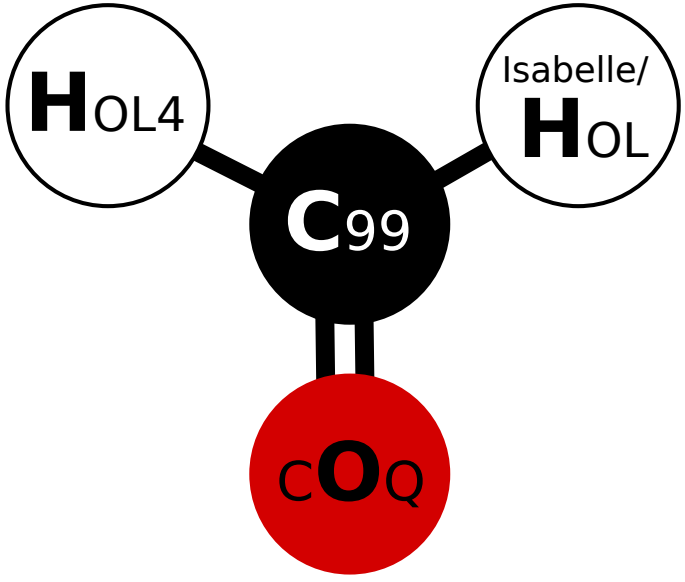
```
int x = 30, y = 31;
int *p = &x + 1, *q = &y;
if (memcmp(&p, &q, sizeof(p)) == 0) {
  printf("%d\n", *p);
}
```

Defect report #260:

> *The implementation is permitted to use the derivation of a pointer value in determining whether or not access through that pointer is undefined behaviour, . . .*

The Formalin project

- May 2011 to May 2015
- Create a formalization of the **complete** C99 standard
- In the theorem provers HOL4, Isabelle/HOL and Coq
- Which follow the standard closely
- All derived from a common master formalization (e.g. in Ott)



Features

- C preprocessor
- C standard library
- Floating point arithmetic
- Casts
- Non-determinism
- Sequence points
- Alignment requirements
- Non-local control flow (`goto`, `setjmp`/`longjmp`, signal handling)
- `volatile`, `restrict` and `const` variables
- Programs in a 'freestanding environment'

Purposes

- Utterly precise version of the standard
- Validate correctness of formal versions of subsets of C (e.g. Compcert)
- Verify correctness of verification conditions generated by tools (e.g. VCC or Frama-C)

Research team

Robbert Krebbers

PhD student
RU, The Netherlands

Freek Wiedijk

Project leader
RU, The Netherlands

Herman Geuvers

Promotor
RU, The Netherlands

James McKinna

Advisor
RU, The Netherlands

Erik Poll

Advisor
RU, The Netherlands

Michael Norrish

HOL advisor
NICTA, Australia

Andreas Lochbihler

Isabelle advisor
KIT, Germany

Jean-Christophe Filliâtre

Coq advisor
CNRS, France

Related projects

- Michael Norrish. C and C++ semantics (L4.verified)
- Xavier Leroy *et al.* Verified C compiler in Coq (Compcert)
- Chucky Ellison and Grigore Rosu. Executable C semantics in Maude (KCC)

More information

http://ch2o.cs.ru.nl/