The OSCAR Computer Algebra System

Lars Göttgens, RWTH Aachen University

. . . and the OSCAR Development Team

https://oscar-system.org/

June 3, 2025

What is OSCAR?
https://oscar-system.org/

- Open Source Computer Algebra Research system
- A tool for interdisciplinary research and computations in algebra, geometry, and number theory
- Also a toolkit for applications of algebra (coding theory, cryptography, bioinformatics, ...)
- Funded by German Research Council (DFG) from 2017-2028
- Contributions from 108 individuals from all over the world
- Open to contributions from anyone, curated via code reviews
- All parts of OSCAR are open source

OSCAR Is Both Old and New

- OSCAR is “new” ...

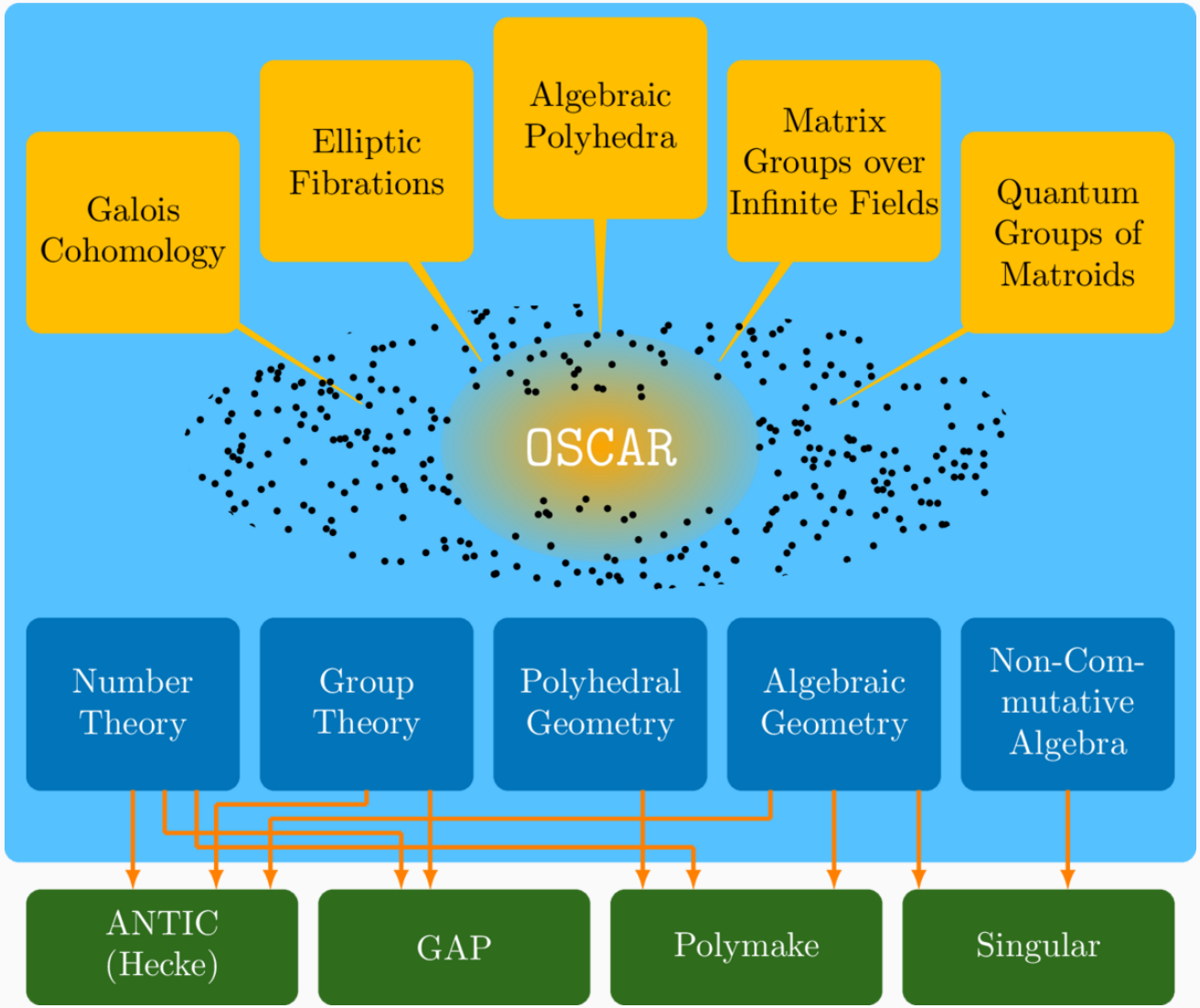
OSCAR Is Both Old and New

- OSCAR is “new” ...
- ...but based on pre-existing systems, the four *cornerstones*:
  - ANTIC (Nemo, Hecke)
  - GAP
  - polymake
  - Singular

OSCAR Is Both Old and New

- OSCAR is "new" ...
- ...but based on pre-existing systems, the four *cornerstones*:
  - ANTIC (Nemo, Hecke)
  - GAP
  - polymake
  - Singular
- constituting ≥ 30 years of work and a few million lines of code
- tied together and extended by code written in the Julia programming language
- development of OSCAR and cornerstones is tightly intertwined
- cornerstone creators & custodians are OSCAR team members

The structure of OSCAR



OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...
- algebraic geometry: schemes, (elliptic) curves, toric varieties, ...

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...
- algebraic geometry: schemes, (elliptic) curves, toric varieties, ...
- polyhedral geometry, tropical geometry

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...
- algebraic geometry: schemes, (elliptic) curves, toric varieties, ...
- polyhedral geometry, tropical geometry
- noncommutative algebra: PBW-algebras, GR-algebras, ...

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...
- algebraic geometry: schemes, (elliptic) curves, toric varieties, ...
- polyhedral geometry, tropical geometry
- noncommutative algebra: PBW-algebras, GR-algebras, ...
- Lie theory: root systems, Weyl groups, Lie algebras, modules, ...

OSCAR Is More Than Group Theory

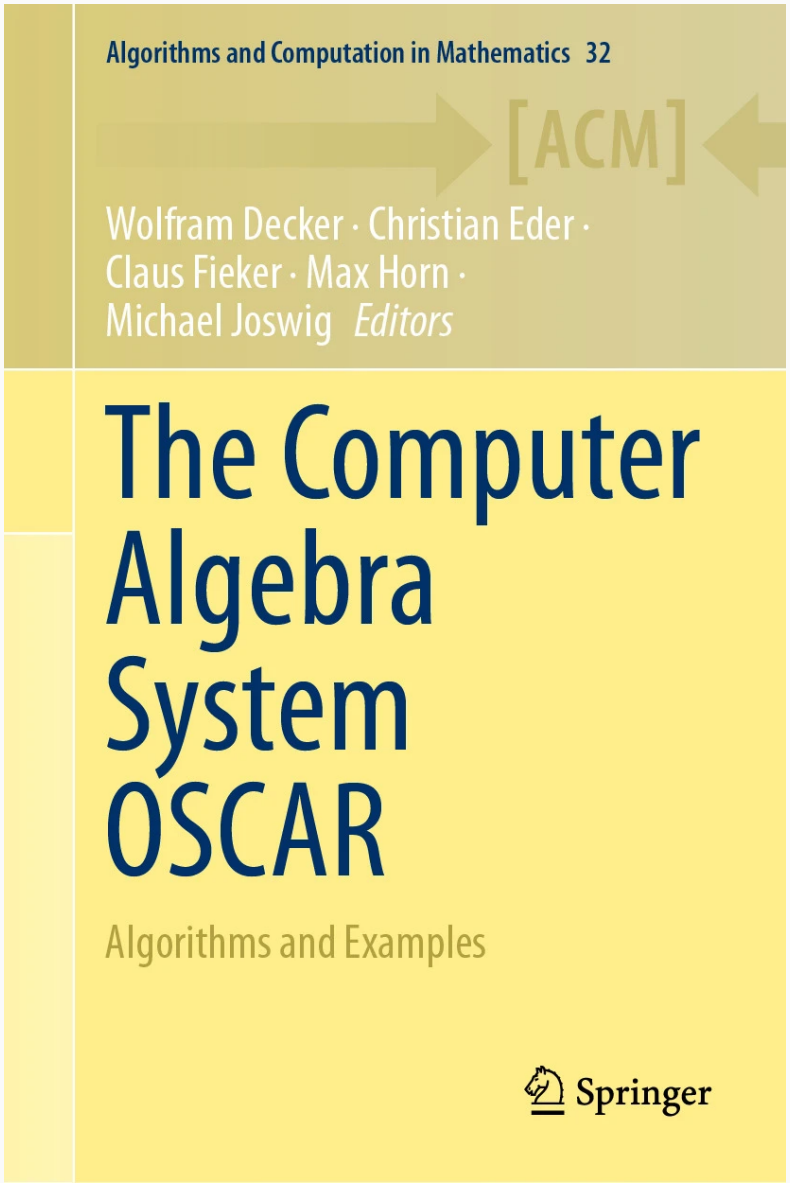
- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...
- algebraic geometry: schemes, (elliptic) curves, toric varieties, ...
- polyhedral geometry, tropical geometry
- noncommutative algebra: PBW-algebras, GR-algebras, ...
- Lie theory: root systems, Weyl groups, Lie algebras, modules, ...
- ...and much more is there, or to come

The OSCAR Book

Get the OSCAR book for many worked out examples

19 chapters with topics ranging from basics to current research.

See book.oscar-system.org



Why Julia?

Performance

- Solves the 2-language problem:
  - easy to write
  - near C performance due to JIT compilation
- supports parallel computing

Why Julia?

Performance

- Solves the 2-language problem:
  - easy to write
  - near C performance due to JIT compilation
- supports parallel computing

Community and Ecosystem

- Open Source (MIT License)
- supports multiple platforms
- designed by mathematically minded people
- large, growing ecosystem

Why Julia?

Julia features

- easy C interoperability; good C++ support
- good support for interactive use in the REPL:
  - searchable history, tab-completion, help mode
- first-class Jupyter notebook support
- dynamically typed
- multiple dispatch
- automatic memory management (garbage collector)

Example computations

Topics:

1. Algegraic Lie Theory: Root systems, Weyl groups, Lie algebras
2. Group Theory: Permutation groups, G-sets
3. Number Theory: Number fields, Class groups

Number Theory

in OSCAR

Lars Göttgens, RWTH Aachen University

Examples by C. Fieker and T. Hofmann

https://oscar-system.org/

June 3, 2025

Feature overview

- Number fields,
- Orders and fractional ideals,
- Class groups, unit groups,
- Lattices,
- Class fields,
- Elliptic curves,
- Galois theory,
- G-modules,
- ...

Feature overview (this talk)

- **Number fields,**
- **Orders and fractional ideals,**
- **Class groups**, unit groups,
- Lattices,
- Class fields,
- Elliptic curves,
- Galois theory,
- G-modules,
- ...

Number fields: construction

Simple number fields are realized as $K \cong \mathbb{Q}[x]/(f)$, i.e. via the minimal polynomial of some primitive element.

Number fields: construction

Simple number fields are realized as $K \cong \mathbb{Q}[x]/(f)$, i.e. via the minimal polynomial of some primitive element.

```
julia> Qx, x = QQ[:x]; # suppress the output

julia> K, a = number_field(x^2 - 235, :a)
(Number field of degree 2 over QQ, a)

julia> a^2 - 235 # confirm that a is a root of x^2 - 235
0
```

Number fields: construction

Simple number fields are realized as $K \cong \mathbb{Q}[x]/(f)$, i.e. via the minimal polynomial of some primitive element.

```
julia> Qx, x = QQ[:x]; # suppress the output

julia> K, a = number_field(x^2 - 235, :a)
(Number field of degree 2 over QQ, a)

julia> a^2 - 235 # confirm that a is a root of x^2 - 235
0
```

Convenience construtors for special cases like:

```
julia> K, a = quadratic_field(235)
(Real quadratic field defined by x^2 - 235, sqrt(235))

julia> a^2 - 235
0
```

Number fields: constructing elements

a is a primitive element $\Rightarrow \{1, a\}$ is a $\mathbb{Q}$-basis of K

```
julia> b = 2 + 1//3 * a
1//3*a + 2

julia> coordinates(b)
2-element Vector{QQFieldElem}:
 2
 1//3
```

Number fields: constructing elements

a is a primitive element $\Rightarrow \{1, a\}$ is a $\mathbb{Q}$-basis of K

```
julia> b = 2 + 1//3 * a
1//3*a + 2

julia> coordinates(b)
2-element Vector{QQFieldElem}:
 2
 1//3
```

Alternative: map polynomials under $\mathbb{Q}[x] \to \mathbb{Q}[x]/(x^2 - 235) = K$

```
julia> K(x^3 + x + 2)
236*a + 2
```

Number fields: element properties

```
julia> b = 2 + 1//3 * a
1//3*a + 2

julia> representation_matrix(b)
[     2   1//3]
[235//3      2]

julia> trace(b)
4

julia> norm(b)
-199//9

julia> is_integral(b)
false

julia> minimal_polynomial(b)
x^2 - 4*x - 199//9
```

Ring of integers

$\mathcal{O}_K$ is the subring of K comprised of all elements with integral minimal polynomial.

```
julia> OK = ring_of_integers(K)
Maximal order
  of number field with defining polynomial x^2 - 235
    over rational field
with Z-basis [1, a]

julia> basis(OK)
2-element Vector{AbsSimpleNumFieldOrderElem}:
 1
 a
```

OSCAR determines the integral basis $\omega = \{1, a\}$, which means $\mathcal{O}_K = \mathbb{Z} \cdot 1 \oplus \mathbb{Z} \cdot a$.

Ring of integers

$\mathcal{O}_K$ is the subring of K comprised of all elements with integral minimal polynomial.

```
julia> OK = ring_of_integers(K)
Maximal order
  of number field with defining polynomial x^2 - 235
    over rational field
with Z-basis [1, a]

julia> basis(OK)
2-element Vector{AbsSimpleNumFieldOrderElem}:
 1
 a
```

OSCAR determines the integral basis $\omega = \{1, a\}$, which means $\mathcal{O}_K = \mathbb{Z} \cdot 1 \oplus \mathbb{Z} \cdot a$.

$$\operatorname{disc}(\mathcal{O}_K) = \det((\operatorname{Tr}_{K/\mathbb{Q}}(\omega_i \omega_j))_{i,j})$$

```
julia> discriminant(OK)
940
```

Unique factorization of ideals

$\mathcal{O}_K$ is an integral extension of $\mathbb{Z}$

$\Rightarrow$ the canonical map $\mathrm{Spec}(\mathcal{O}_K) \to \mathrm{Spec}(\mathbb{Z})$ is surjective

```
julia> prime_ideals_over(OK, 7)
2-element Vector{AbsSimpleNumFieldOrderIdeal}:
 <7, a + 5>
 <7, a + 2>
```

These two prime ideals lie above the prime 7.

Unique factorization of ideals

$\mathcal{O}_K$ is an integral extension of $\mathbb{Z}$
$\Rightarrow$ the canonical map $\mathrm{Spec}(\mathcal{O}_K) \to \mathrm{Spec}(\mathbb{Z})$ is surjective

```
julia> prime_ideals_over(OK, 7)
2-element Vector{AbsSimpleNumFieldOrderIdeal}:
 <7, a + 5>
 <7, a + 2>
```

These two prime ideals lie above the prime 7.

We can also factorize ideals into prime ideal:

```
julia> factor(7 * OK)
Dict{AbsSimpleNumFieldOrderIdeal, Int64} with 2 entries:
  <7, a + 5> => 1
  <7, a + 2> => 1

julia> factor(a * OK)
Dict{AbsSimpleNumFieldOrderIdeal, Int64} with 2 entries:
  <47, a> => 1
  <5, a>  => 1
```

Class group

Some notation:

- I_K set of invertible fractional ideals of $\mathcal{O}_K$
- $P_K \trianglelefteq I_K$ subgroup of principal factional ideals
- $\mathrm{Cl}(K) = I_K / P_K$ class group

Class group

Some notation:

- I_K set of invertible fractional ideals of $\mathcal{O}_K$
- $P_K \trianglelefteq I_K$ subgroup of principal factional ideals
- $\mathrm{Cl}(K) = I_K/P_K$ class group

```
julia> A, m = class_group(OK);

julia> A
Z/6
```

This tells us that $\mathrm{Cl}(K) \cong \mathbb{Z}/6\mathbb{Z}$ (as groups).

Class group: interpretation map

The second return value is an *interpretation map*.
Here: $m : A \to I_K$ such that $A \xrightarrow{m} I_K \to I_K/P_K = \mathrm{Cl}(K)$ is an isomorphism

Class group: interpretation map

The second return value is an *interpretation map*.
Here: $m : A \to I_K$ such that $A \xrightarrow{m} I_K \to I_K/P_K = \mathrm{Cl}(K)$ is an isomorphism

```
julia> m(zero(A))
<1>

julia> p = prime_ideals_over(OK, 2)[1]
<2, a+1>

julia> preimage(m, p)
Abelian group element [3]
```

$\mathfrak{p}$ corresponds to $\overline{3} \in \mathbb{Z}/6\mathbb{Z}$, i.e. $\mathfrak{p}^2 \in P_K$

Class group: interpretation map

The second return value is an *interpretation map*.
Here: $m : A \to I_K$ such that $A \xrightarrow{m} I_K \to I_K/P_K = \mathrm{Cl}(K)$ is an isomorphism

```
julia> m(zero(A))
<1>

julia> p = prime_ideals_over(OK, 2)[1]
<2, a+1>

julia> preimage(m, p)
Abelian group element [3]
```

$\mathfrak{p}$ corresponds to $\overline{3} \in \mathbb{Z}/6\mathbb{Z}$, i.e. $\mathfrak{p}^2 \in P_K$

```
julia> is_principal_with_data(p^2)
(true, 2)

julia> p^2 == 2*OK
true
```

Relative/non-simple number fields

In practice, not every number field is given as a simple extension of $\mathbb{Q}$ by one generator.

Options in OSCAR:

- a simple or non-simple extension
- extension of $\mathbb{Q}$ (*absolute field*) or of another number field (*relative field*)

Relative/non-simple number fields

In practice, not every number field is given as a simple extension of $\mathbb{Q}$ by one generator.

Options in OSCAR:

- a simple or non-simple extension
- extension of $\mathbb{Q}$ (*absolute field*) or of another number field (*relative field*)

Example: Construct $\mathbb{Q}(\sqrt{2}, \sqrt{3})$ two different ways

```
julia> K, a = number_field([x^2 - 2, x^2 - 3], :a)
(Non-simple number field of degree 4 over QQ,
↪  AbsNonSimpleNumFieldElem[a1, a2])

julia> a[1]^2 == 2 && a[2]^2 == 3
true
```

Relative/non-simple number fields (2)

Example: Construct $\mathbb{Q}(\sqrt{2}, \sqrt{3})$ two different ways

```
julia> K, a = number_field([x^2 - 2, x^2 - 3], :a) # first way
(Non-simple number field of degree 4 over QQ,
↪  AbsNonSimpleNumFieldElem[a1, a2])

julia> L1, b = quadratic_field(3);
julia> L1t, t = L1[:t];
julia> L, c = number_field(t^2 - 2, :c) # second way
(Relative number field of degree 2 over L1, c)
```

Relative/non-simple number fields (2)

Example: Construct $\mathbb{Q}(\sqrt{2}, \sqrt{3})$ two different ways

```
julia> K, a = number_field([x^2 - 2, x^2 - 3], :a) # first way
(Non-simple number field of degree 4 over QQ,
↪  AbsNonSimpleNumFieldElem[a1, a2])

julia> L1, b = quadratic_field(3);
julia> L1t, t = L1[:t];
julia> L, c = number_field(t^2 - 2, :c) # second way
(Relative number field of degree 2 over L1, c)
```

Important: all field theoretic properties of L are relative to L_1!

```
julia> base_field(K) == QQ, base_field(L) == L1
(true, true)

julia> degree(K), degree(L)
(4, 2)
```

Relative/non-simple number fields (2)

Example: Construct $\mathbb{Q}(\sqrt{2}, \sqrt{3})$ two different ways

```
julia> K, a = number_field([x^2 - 2, x^2 - 3], :a) # first way
(Non-simple number field of degree 4 over QQ,
↪  AbsNonSimpleNumFieldElem[a1, a2])

julia> L1, b = quadratic_field(3);
julia> L1t, t = L1[:t];
julia> L, c = number_field(t^2 - 2, :c) # second way
(Relative number field of degree 2 over L1, c)
```

Important: all field theoretic properties of L are relative to L_1!

```
julia> base_field(K) == QQ, base_field(L) == L1
(true, true)

julia> degree(K), degree(L)
(4, 2)
```

```
julia> absolute_degree(L)
4
```

Number field morphisms

A morphism gets constructed via images of generators, and an optional map of the base field.

Number field morphisms

A morphism gets constructed via images of generators, and an optional map of the base field.

Example: construct map with $\sqrt{2} \mapsto -\sqrt{2}$ and $\sqrt{3} \mapsto -\sqrt{3}$ for absolute non-simple extension $K = \mathbb{Q}(\sqrt{2}, \sqrt{3})$:

```
julia> f = hom(K, K, [-a[1], -a[2]]);

julia> f(a[1]) == -a[1] && f(a[2]) == -a[2]
true
```

Number field morphisms

A morphism gets constructed via images of generators, and an optional map of the base field.

Example: construct map with $\sqrt{2} \mapsto -\sqrt{2}$ and $\sqrt{3} \mapsto -\sqrt{3}$ for absolute non-simple extension $K = \mathbb{Q}(\sqrt{2}, \sqrt{3})$:

```
julia> f = hom(K, K, [-a[1], -a[2]]);

julia> f(a[1]) == -a[1] && f(a[2]) == -a[2]
true
```

for relative simple extension $L = L_1(\sqrt{2}) = \mathbb{Q}(\sqrt{3})(\sqrt{2})$:

```
julia> g1 = hom(L1, L1, -b);

julia> g = hom(L, L, g1, -c);

julia> g(c) == -c && g(L(b)) == L(-b)
true
```

Collapsing number fields

Some functionality is only available for some number field types
$\Rightarrow$ functions converting a number field to an isomorphic one

Collapsing number fields

Some functionality is only available for some number field types
$\Rightarrow$ functions converting a number field to an isomorphic one

- convert to an absoute simple extension

```
julia> Lprime, LprimetoL = absolute_simple_field(L)
(Number field of degree 4 over QQ, Map: Lprime -> L)

julia> defining_polynomial(Lprime)
x^4 - 10*x^2 + 1
```

Collapsing number fields

Some functionality is only available for some number field types
$\Rightarrow$ functions converting a number field to an isomorphic one

- convert to an absoute simple extension

```
julia> Lprime, LprimetoL = absolute_simple_field(L)
(Number field of degree 4 over QQ, Map: Lprime -> L)

julia> defining_polynomial(Lprime)
x^4 - 10*x^2 + 1
```

- convert to a simple extension

```
julia> Kprime, KprimetoK = simple_extension(K)
(Number field of degree 4 over QQ, Map: Kprime -> K)
```

Collapsing number fields

Some functionality is only available for some number field types
$\Rightarrow$ functions converting a number field to an isomorphic one

- convert to an absoute simple extension

```
julia> Lprime, LprimetoL = absolute_simple_field(L)
(Number field of degree 4 over QQ, Map: Lprime -> L)

julia> defining_polynomial(Lprime)
x^4 - 10*x^2 + 1
```

- convert to a simple extension

```
julia> Kprime, KprimetoK = simple_extension(K)
(Number field of degree 4 over QQ, Map: Kprime -> K)
```

- collapse a relative extension $L/K/k$ into L/k with `collapse_top_layer`

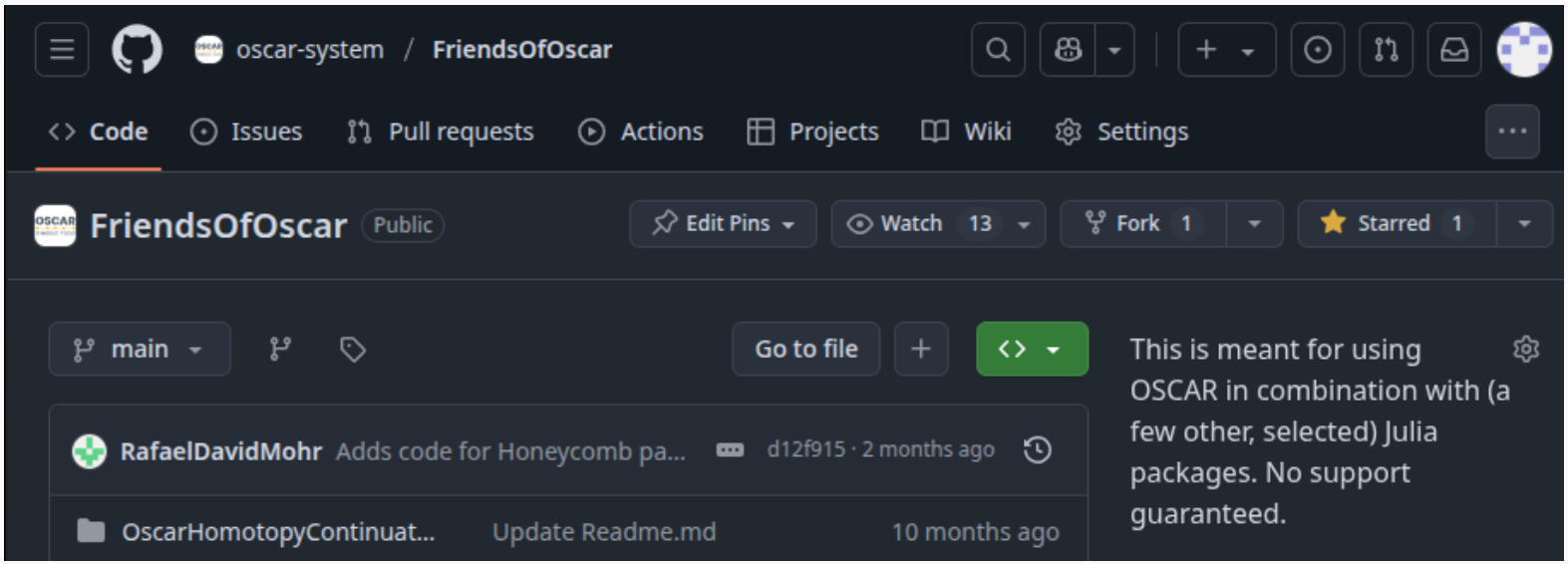
Give it a try yourself!

There is a lot more functionality that does not all fit in this talk.

Have a look at the documentation, talk to us, ...

Friends of OSCAR

A place where we collect julia packages (and julia code in general) that builds upon OSCAR



github.com/oscar-system/FriendsOfOscar/

Please add your existing and future work there!

Try it yourself!

On the website you can find:

- installation instructions
- tutorials
- documentation
- links to slack and github

oscar-system.org

Try it yourself!

On the website you can find:

- installation instructions
- tutorials
- documentation
- links to slack and github

oscar-system.org

We are looking forward to your comments, bug reports, feature requests, code contributions, ...

Try it yourself!

On the website you can find:

- installation instructions
- tutorials
- documentation
- links to slack and github

oscar-system.org

We are looking forward to your comments, bug reports, feature requests, code contributions, ...

oscar-system.org/newsletter

Thank you!