object oriented programming
and
mathematics

by Marc Conrad, Southampton Institute.

“Classical” Reasons for using OOP

- Reusability, Understandability, Data Abstraction, Modularity, Extendability.

OOP is a concept to help solve the “software crisis”, the difficulty to handle more and more complex real world tasks.

OOP has not been developed in a mathematical context.
Why is OOP useful for Mathematicians?

- OOP techniques are already widely used in Software Design. Therefore a lot of tools are available (books, courses, UML, CASE, ...).

and

- There is a strong correspondence between mathematical principles and OOP principles!

Objects and Methods

An object is an encapsulation of data values and methods. It consists of an interface and the implementation:

- The interface provides a collection of methods which can be invoked by other objects. To those, an object is only known by its methods.

- The implementation contains data and algorithms which are used in order to implement the methods.
Objects and Methods (example)

<table>
<thead>
<tr>
<th>algebraic number field $\mathbb{Q}(\alpha)$</th>
<th>implementation</th>
<th>interface (methods)</th>
</tr>
</thead>
<tbody>
<tr>
<td>data and algorithms</td>
<td>$\text{initialize}$ (e.g. with a polynomial)</td>
<td>$\text{compute} \ z = x + y$ for $x, y \in \mathbb{Q}(\alpha)$</td>
</tr>
<tr>
<td></td>
<td>$\text{return}$ minimal polynomial of $\alpha$</td>
<td>$\text{return} \ [\mathbb{Q}(\alpha) : \mathbb{Q}]$</td>
</tr>
</tbody>
</table>

Figure 1: An object algebraic number field.

Inheritance

“An object B inherits from an object A” means that the methods of object A are also available in B. A is called a parent class of B, B is a child class of A.

- Inheritance can be described as a “is a” relationship: B is an A.
- A collection of objects which are connected by inheritance is called an object hierarchy.
Inheritance (example)

Slide 7

- algebraic number field $\mathbb{Q}(\alpha)$
- abelian number field $\mathbb{Q}(\alpha)^{ab}$
  - cyclotomic number field $\mathbb{Q}(\zeta_n)$
  - quadratic number field $\mathbb{Q}(\sqrt{D})$

Figure 2: An object hierarchy of algebraic number fields.

Overriding

Slide 8

In an object hierarchy, *overriding a method* means that a method of a parent class A is *reimplemented* in the child class B.

There are two important applications of overriding:

- Implementation of deferred methods.
- Performance improving of existing methods.
Overriding (example)

- abstract field \( F \)
- \( \mathbb{Q}(\alpha) \)
- \( \text{GF}(2^n) \)
- \( \mathbb{C} \)

- deferred: 0, 1, +, *, -x, 1/x.
- implemented: \( x - y \), \( x/y \), \( x^2 \), \( x^n \), ...

Deferred methods of \( F \) are implemented

Figure 3: Deferred and implemented methods.

- Deferred methods (+, *, ...) are overridden in the child classes.
- Squaring may be overridden (e.g. in \( \text{GF}(2^n) \)) in order to increase performance.

Polymorphism

“A polymorphic object is any entity such as a variable or a function argument that is permitted to hold values of different types during the course of execution.”

- Polymorphic references allow the construction of structures which are related to a collection of objects with similar properties.
- Of special interest is the situation where a polymorphic reference represents an arbitrary object of a given hierarchy.
Polymorphism (example): Matrices

![Diagram showing the relationship between matrices and abstract fields.](image)

Figure 4: Matrices, defined over an arbitrary field.

Polymorphism (example): Polynomials

![Diagram showing the relationship between polynomial rings and abstract fields.](image)

Figure 5: Multivariate polynomials by univariate polynomials.
Elliptic curves (definition and properties)

- An elliptic curve over a field \( F \) is an isomorphism class of models of the form \( y^2 + a_1 y + a_3 y = x^3 + a_2 x^2 + a_4 x + a_6 \) with \( a_i \in F \).
- Two models are isomorphic, if there is a birational transformation of the form \( x' = u^2 x + r \) \( y' = u^3 y + u^2 s x + t \) with \( r, s, t \in F \) and \( u \in F^* \).
- The points on the curve form a group (Mordell-Weil group) where the group law can be given explicitly by rational functions.

Object hierarchy of models of an elliptic curve

1. abstract model
2. polymorphic field
3. model
4. over finite fields
5. model
6. over number fields
7. model
8. over other fields
9. ...
Methods in the model hierarchy

abstract: construction by \( a_1, \ldots, a_6 \) or by a birational transformation • computation of model dependent constants (Tate’s values, discriminant) • addition of points

finite fields: computation of special points, e.g. points of large order • algorithm for solving the discrete logarithm problem

number fields: computation of torsion points, generators, integral and \( S \)-integral points • estimates concerning heights

Object hierarchy of elliptic curves

abstract elliptic curve ➔ list of models (polymorphic)

elliptic curve over finite fields

elliptic curve over number fields

elliptic curve over other fields ➔ ...
Methods in the elliptic curve hierarchy

abstract: construction by a model, which is then labeled the actual model • facilities to manage a list of models and birational transformations between these models • computation of model independent constants (e.g. j-invariant) • computation of the structure of the point group (as deferred method)

finite fields: special models (depending on the characteristic of the field) • computation of the number of points of the group

number fields: special models (SWNF, minimal models) • computation of regulator, conductor, rank, ...

Object oriented elliptic curves (summary)

In order to implement elliptic curves we use an object hierarchy for elliptic curves and an object hierarchy of models. The methods in these hierarchies are:

- field independent → abstract level
- model independent → curve hierarchy

or

- field dependent → concrete level
- model dependent → model hierarchy
Concrete programming languages

**C++:**
- widely used, extension of C (⇒ fast implementations).
- no a priory implementation of polymorphism, overriding must be explicitly allowed by the parent class ("virtual methods").

**Java:**
- provides polymorphism and automatic garbage collection, commonly used (but not in mathematical context).
- slower then C++, no multiple inheritance.

**Objective-C, Object Pascale, Smalltalk, ...:**
- provide various facilities concerning oop.
- uncommon in computer algebra, no efficient arithmetics.

There is a strong correspondence between object oriented concepts and mathematical structures:

<table>
<thead>
<tr>
<th>OOP</th>
<th>Math</th>
</tr>
</thead>
</table>
| Inheritance | Specification of structures.  
|           | "A field is a ring with ..."          |
| Overriding | Abstract definitions for concrete applications.  
|           | "x^2 := xx ⇒ 3^2 = 9"               |
| Polymorphism | Use of abstract structures in other structures.  
|           | "A matrix over a ring."          |

In computational mathematics object oriented concepts are suited not only for reasons of software design but for principal reasons.