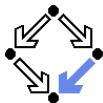


# Object-Oriented Programming in C++ (Course “Computer Systems”)

Wolfgang Schreiner  
Wolfgang.Schreiner@risc.uni-linz.ac.at

Research Institute for Symbolic Computation (RISC)  
Johannes Kepler University, Linz, Austria  
<http://www.risc.uni-linz.ac.at>



# Overview

---

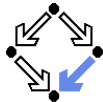


A continuation of the course “Programming” in the last semester.

- **Last semester:** procedural (“imperative”) programming in C++.
  - Focus on organization of control flow.
  - Passive entities (“data”) processed by active entities (“functions”).
  - Programs organized as sets of functions.
- **This semester:** object-oriented programming in C++.
  - Focus on organization of data.
  - “Classes” combine data and functions to “objects”.
  - Programs organized as sets of classes.

Modern approach to “programming in the large”.

# Example



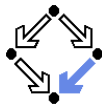
```
struct Date {
    int day;
    int month;
}

static void print(Date d)
{
    cout << d.day << "."
         << d.month << ".";
}

Date d;
d.day = 24;
d.month = 12;
print(d);
```

```
class Date {
private:
    int day;
    int month;
public:
    Date(int d, int m) {
        day = d;
        month = m;
    }
    void print() {
        cout << day << "."
             << month << ".";
    }
}

Date d = Date(24, 12);
d.print();
```



- **Classes:** combining data and functions.
  - Classes versus records (“structs”).
  - Construction, destruction, assignments.
  - Static members versus non-static members.
- **Inheritance:** building class hierarchies.
  - Classes and subclasses.
  - Virtual functions and overriding.
  - Abstract classes, interfaces, frameworks.
- **Templates:** type-generic (“polymorphic”) programming.
  - Function templates.
  - Class templates.
- **The C++ Standard Library:** reusing existing functionality.
  - Basic features (I/O, numerics, etc).
  - Containers, iterators, and algorithms.

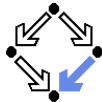
# Organization



- **Lecture:** concepts and examples.
  - Slides, blackboard, online demonstrations.
- **Assignments:** moderate programming exercises.
  - 6 assignments are given.
    - Results to be submitted within 2 weeks.
  - Tutors: Andreas Müller, Mikhael Yuditsky.
    - See the course site for the dates of the meetings offered every week.
- **Grades:** based on final exam and assignments.
  - Final exam (50%): concepts and very small programming tasks.
  - Assignments (50%): best 5 results are evaluated.
  - Both exam and assignments have to be positive.
- **Literature:** any good C++ book will do for this course.
  - Ray Lischner: “C++ in a Nutshell”, O’Reilly, 2003.

# Moodle Course

---



Central point for electronic communication.

- Register as a participant in the Moodle course.
  - All forum messages will be sent as emails to registered participants.
- Submit assignments via Moodle.
  - No email submissions are accepted.
- Post general questions in the “course forum”.
  - Answered by Wolfgang Schreiner.
- Post assignment-specific questions in “assignment forum”.
  - Answered by one of the tutors.

<http://www.risc.uni-linz.ac.at/people/schreine/courses/ss2010/compsys>

# C++ Software

---



Any C++ compiler and program editor will do for this course.

- **Eclipse IDE for C/C++ Developers**

  - `http://www.eclipse.org/downloads`

- **GNU/Linux:** The GNU C++ Compiler.

  - Debian: `apt-get install g++`

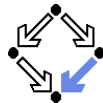
- **Windows:** MinGW and MSYS.

  - `http://www.mingw.org`

  - `http://max.berger.name/howto/cdt`

A decent IDE makes program development much more productive.

# Eclipse IDE for C/C++ Developers



```
/*
 * HelloWorld.cpp
 * Created on: 03.09.2008
 * Author: schreine
 */

#include <iostream>

using namespace std;

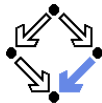
int main()
{
    char message[] = "Hello, World!";
    cout << message;
    cout.flush();
}
```

C-Build [HelloWorld]

Building target: HelloWorld  
Invoking: GCC C++ Linker  
g++ -o"HelloWorld" ./Date.o ./HelloWorld.o  
Finished building target: HelloWorld



# Eclipse Debug View



The screenshot displays the Eclipse IDE's Debug View for a C++ application. The main window is titled "Debug - HelloWorld/HelloWorld.cpp - Eclipse Platform". The interface is divided into several panes:

- Debug Console:** Shows the execution state of the application. It is currently suspended at a breakpoint. The thread list shows "Thread [0] (Suspended: Breakpoint hit)" and "1 main() /usr2/schreine/workspace-cpp/HelloWorld/HelloWorld.cpp".
- Variable View:** Displays the current state of variables. The "message" variable is expanded, showing its value as "0xbfdb4cc6" and its content as "H". Below it, the memory address "0xbfdb4cc6" is shown with the value "H".
- Source Editor:** Shows the source code of the "HelloWorld.cpp" file. The current line of execution is highlighted in blue, corresponding to the breakpoint hit. The code is:

```
int main()
{
    char message[] = "Hello, World!";
    cout << message;
    cout.flush();
}
```
- Outline View:** Shows the project structure, including "iostream", "std", and "main(): int".
- Console:** Shows the output of the application, which is "Hello, World!".