Speicherverwaltung und Speicherbereinigung

WS 2009/2010

Themen, betreut von

Univ.-Prof. em. Dr. H.-J. Hoffmann

Stand 18. Oktober 2009

Die Literaturangaben in den einzelnen Themenblättern dienen als Anleitung zur jeweiligen Thematik. Die Bearbeiter müssen sich um einen darüber hinausgehenden Überblick bemühen.

Angaben noch unvollständig !

Alle Internet-Zugriffe September – Oktober 2009 !
Irrtum und Tippfehler vorbehalten !
Alle Wikipedia-Referenzen sind von 2009 !

Leider sind in einigen angegebenen Literaturstellen immer wieder Tippfehler zu finden.

Anleitung zur Vorbereitung Ihres Vortrags bzw. der Ausarbeitung:

- S.I.P. Jones et al.: *How to give a good research talk*; ACM SIGPlan Notices 28 (1993) 11, 9 - 12
- M. Deininger et al.: *Studien-Arbeiten, ein Leitfaden* ...; (u.a.) Teubner, 1992
- M.R Theisen: *Wissenschaftliches Arbeiten* ; Verlag Vahlen, 2008 (und eventl. frühere)

Können in der Bibliothek des FB Informatik eingesehen werden !
Es gibt für dieses Seminar dort auch einen sog. „Semesterapparat"

Inhaltliche Vorbereitung für alle:

NN(): *Basics of computer memory* ; 2006,
www.osdata.com/system/physical/memory.htm

NN (Microsoft):
Back to basic - Series on dynamic memory management ; 2009,
blogs.msdn.com/abhinaba/archive/2009/01/25/back-to-basic-series-on-dynamic-memory-management.aspx

R.E. Jones, R. Lins: *Garbage collection – Algorithms for automatic dynamic memory management* ;
Wiley, 1996, ISBN 0-471-94148-4
(im Semesterapparat ! – hier gekennzeichnet als „siehe auch JL" –)

D.F. Bacon et al: *A Unified Theory of Garbage Collection* ;
ACM SIGPLAN Notices, 29 (2004) 10, 50 - 68

Zur kürzeren Schreibweise von URLs
ist „http://" bzw. „https://" immer weggelassen!

Übersicht

- Einführung
  - #01 Speichermöglichkeiten früherer und heutiger Rechnersysteme
  - #02 Adressierung – Hardware
  - #03 Adressierung – Software
- Programmiertechnisches Umfeld
  - #04 Klassische Programmiersprachen
  - #05 Objektorientierte Programmiersprachen
  - #06 Kontourmodell
- Implementierung
  - #07 Aufrufkonventionen, Aufrufkeller
  - #08 0-Adresskeller
  - #09 Nebenläufige Programme u.ä. *
- Speicherverwaltung und Speicherbereinigung bei objektorientierten Sprachen
  - #10 einfache Fälle
  - #11 Referenzzähler
  - #12 moderne Verfahren
  - #13 ... bei Smalltalk
  - #14 ... bei C++, Java u.ä. *
  - #15 ... bei .NET *
  - #16 ... bei Stöberern u.ä. *

Mindestteilnehmerzahl 12, Höchstteilnehmerzahl 16.

Die mit * markierten Themen stehen am Ende zur Auswahl,

Themen #01 bis #08 und #10 bis #13 werden zuerst vergeben !

HJH#01a
2009/2010

Speichermöglichkeiten früherer und heutiger Rechnersysteme

- NN (Wikipedia): *IBM System/360* ;
  de.wikipedia.org/wiki/System/360 und
  en.wikipedia.org/wiki/System/360
  (siehe auch S. Arun-Kumar:
  www.cse.iitd.ernet.in/~sak/courses/cdp/cards/IBM360.pdf)
- NN (Global Oneness): *Burroughs B5000 - Stack architecture* ; 1961, www.experiencefestival.com/a/
  burroughs b5000 – stack architecture/id/4823150
  – siehe Referat #08 ! –
- NN (Wikipedia): *X86-Prozessor* und *Intel 8086* ;
  de.wikipedia.org/wiki/X86-Prozessor **und**
  de.wikipedia.org/wiki/Intel_8086

Fortsetzung

- NN (Wikipedia): *Reduced instruction set computing* ;
  de.wikipedia.org/wiki/Reduced_Instruction_Set_Computing
  und en.wikipedia.org/wiki/RISC
- NN (Wikipedia): *Complex instruction set computing* ;
  de.wikipedia.org/wiki/Complex_Instruction_Set_Computing
  & en.wikipedia.org/wiki/Complex_Instruction_Set_Computer

HJH#01b
2009/2010

Speichermöglichkeiten früherer und heutiger Rechnersysteme

- M. Karbo: *PC architecture* ;
  `www.karbosguide.com/books/pcarchitecture/start.htm`
- NN (Wikipedia): *PowerPC* ; `de.wikipedia.org/wiki/PowerPC`
- NN (Wikipedia): *OLPC XO-1* :
  `de.wikipedia.org/wiki/One_Laptop_per_Child`

Eine Anmerkung zu den Wikipedia-Referenzen:

Zum Verdeutlichen und Umschreiben eines Seminarthemas sind diese Referenzen m.E. hilfreich. Zur Vorbereitung des Referats und dem Abfassen der Literaturangaben in der Ausarbeitung reichen sie aber in der Regel nicht aus !

Adressierung
Befehlsformate, Registerverwendung u.ä.

Blickwinkel *Hardware*

- NN (Wikipedia): *Instruction* und *Instruction set* ; **en.wikipedia.org/wiki/Instruction_(computer_science) und ... Instruction_set_architecture**
- NN: *Assembly Language - address space and addressing modes* ; 2001, **www.osdata.com/topic/language/asm/address.htm**
- NN (Wikipedia): *Subroutines* ; **en.wikipedia.org/wiki/Subroutine**
- R. Hyde: *Minimal 80x86 instruction set (Online appendix A)* und *Minimal PowerPC instruction set (Online appendix B)* ; 2006, **nostarch.com/greatcode2.htm**
- NN (): *Can ARM code be relocatable/reentrant?* ; 2009, **infocenter.arm.com/help/topic/com.arm.doc.faqs/ka3698.html**

Adressierung
Befehlsformate, Registerverwendung u.ä.

Blickwinkel *Software*

- R. Hyde: *Variables in a high-level language* ; 2006, `nostarch.com/download/greatcode2_ch8.pdf`
- NN (The Ganssle group): *Writing relocatable code* ; 1992, `www.ganssle.com/articles/arelocat.htm`
- NN(): *Crossware 8051 C compiler* ; undatiert, `www.crossware.com/datasheets/c8051nt.htm`
- S. Kawat: *Understanding variable scope and duration* ; 2001, `www.suite101.com/article.cfm/learning_visual_basic/64055`
- NN: *C# variable scopes* ; 2009, `www.blackwasp.co.uk/CSharpVariableScopes.aspx`
- NN (Wikipedia): *Dangling pointer* ; `en.wikipedia.org/wiki/Dangling_pointer`
- A.N. Kumar: *Learning the interaction between pointers and scope in C++* ; ACM SIGCSE Bulletin, 33 (2001) 3, 45 - 48

Klassische Programmiersprachen

- E.D. Reilly: *Activation record* . In: A. Ralston et al.: *Encyclopedia of computer science* , John Wiley and Sons Ltd , 4th edition, 2003, ISSN:0-470-86412-5 , 10 – 12, portal.acm.org/citation.cfm?id=1074106&CFID=47199798&CFTOKEN=26125053 (über Informatikbibliothek erreichbar)
- R. Cartwright: *Understanding run-time environment Representation and Control* ; 2008, www.cs.rice.edu/~javaplt/311/Notes/08/09.html
- J. Göers: *Imperative Sprachen* ; 2008, www-lehre.inf.uos.de/psk/0809/pdf/kapitel6b.pdf
- H. Weber: Laufzeit-Speicherverwaltung ; 2006, *www.informatik.fh-wiesbaden.de/~weber/fachsem/kap7.pdf*
- J. Cain: *The activation record concept* ; 2000, *cse.stanford.edu/class/cs107/handouts/22TheActivationConcept.pdf*

Fortsetzung

Klassische Programmiersprachen

- (wer es bildlich schätzt:) `images.google.de/images?hl=de&q=%22stack+frame%22&um=1&ie=UTF-8&ei=OzCMSqrIH4SKnQPFjvx4&sa=X&oi=image_result_group&ct=title&resnum=4`

- NN (Wikipedia): *Man or boy test* (nach D. Knuth, 1964) ; `en.wikipedia.org/wiki/Man_or_boy_test`
- NN (Wikipedia): *Funarg problem* und *Scope* ; `en.wikipedia.org/wiki/Funarg_problem` bzw. `.../Scope`

(– bzgl. Pascal siehe Referat #08 ! –)

Objektorientierte Programmier-sprachen

Smalltalk

- NN (Wikipedia): *Smalltalk* ; `de.wikipedia.org/wiki/Smalltalk-80_(Programmiersprache)` und `en.wikipedia.org/wiki/Smalltalk`
- A.L. Lovejoy : *Smalltalk, getting the message - The essentials of message-oriented programming with Smalltalk* ; 2007, `www.smalltalk-resources.com/Smalltalk-Getting-the-Message.html`
- R. Lutze: *Die Implementierung von Smalltalk* . In H.-J. Hoffmann: *Smalltalk – verstehen und anwenden* ; Carl Hanser-Verlag, 1987, 129 – 188
- D. Ingalls et al.: *Back to the future, the story of Squeak, ... (Storage Management)* ; ACM SIGPLAN Notes 32 (1997) 10, 318 - 326
- S. Ducasse: *Evaluating message passing control techniques in Smalltalk* ; Journal object-oriented programming (JOOP) , June 1999, `scg.unibe.ch/archive/papers/Duca99aMsgPassingControl.pdf`

Fortsetzung

Objektorientierte Programmier-sprachen

Anderes

- L. Cain: *More detail about activation records - Layout of memory during a function call* ; lecture 10, academicearth.org/lectures/activation-records-layout-of-memory-during-function-call
- NN (Wikipedia) : *Message passing* ; en.wikipedia.org/wiki/Message_passing
- NN: *Smalltalk-like message passing in C++* : 2003, www.objectvalue.com/articles/CppMessagePassingV04.html
- B. Eckel, C. Allison: *Thinking in C++ ... (exceptions)* ; www.linuxtopia.org/online_books/programming_books/c%2B%2B_practical_programming/c++_practical_programming_038.html
- D. Schmidt: *The activation-record stack (Java)* ; 2005, people.cis.ksu.edu/~schmidt/300s05/Lectures/Lecture3.html
- A.O. Ramirez: *Programming bits - C# data types* ; linuxgazette.net/issue85/ortiz.html

Kontourmodell

- J.B. Johnston: *The contour model of block structured processes* ; ACM SIGPLAN Notices, 6 (1971) 2, 55 – 82
- D.M. Berry: *Introduction to Oregano* ; ACM SIGPLAN Notices, 6 (1971) 2, 171 - 190
- D.M. Berry: *Block structure - retention or deletion?* ; *Proc. 3rd Annual ACM Symp. Theory of Computing*, 1971, 86 - 100
- P. Wegner: *Programming languages, information structures, and machine organization* ; McGraw-Hill, 1968
- E.I. Organick et al.: *Programming language structures* ; Academic Press, 1978
- L.L. Deneen: *The contour model as an instructional tool in elementary computer science* ;
  ACM SIGCSE Bulletin, 19 (1987) 1, 170 - 178

Fortsetzung

Kontourmodell

Eine Anwendung bei ADA

- L.K. Dillon: *A visual execution model for Ada tasking* ; ACM Trans. Software Engineering & Methodology, 2 (1993) 4, 311 – 345

Eine Anwendung für Java-Programme

- B. Jayaraman, C.M. Baltus: *Visualizing program execution* ; IEEE Proc. Symp. Visual Languages, 1996, 30 - 37
- P. Gestwicki, B. Jayaraman: Interactive visualization of Java programs ; IEEE Symp. Human-Centric Computing, Languages, and Environments, 2002, 226–235
- P. Gestwicki, B. Jayaraman: Methodology and architecture of JIVE ; Proc. ACM Symp Software Visualization, 2005, 95 - 104

Aufrufkonventionen, Aufrufkeller

- NN (Wikipedia): *Calling convention* ; **en.wikipedia.org/wiki/Calling_convention**
- NN (Wikipedia): *Stack-based memory allocation* ; **en.wikipedia.org/wiki/Stack-based_memory_allocation**
- NN (Wikipedia): *Dynamic memory allocation* ; **en.wikipedia.org/wiki/Dynamic_memory_allocation**
- NN (Wikipedia): *Call stack* ; **en.wikipedia.org/wiki/Call_stack**
- NN (Wikipedia): *Automatic variable* ; **en.wikipedia.org/wiki/Automatic_variable**
- NN (Wikipedia): *Thread* ; **en.wikipedia.org/wiki/Thread_(computer_science)**

0-Adresskeller

- NN (Wikipedia): *Kellerautomat* ;
  `de.wikipedia.org/wiki/Kellerautomat` und
  `en.wikipedia.org/wiki/Pushdown_automaton`
- NN (Wikipedia): *Stack-oriented programming language* ;
  `en.wikipedia.org/wiki/Stack-oriented_programming_language`
  und `en.wikipedia.org/wiki/Stack_machine`
- NN (Global Oneness): *Burroughs B5000 - Stack architecture* ;
  1961, `www.experiencefestival.com/a/burroughs b5000 - stack architecture/id/4823150`
  (Zitat von B. Randell: *„The machine that everyone loves, and nobody buys„*)
- NN (Wikipedia): *Burroughs large systems* ;
  `en.wikipedia.org/wiki/Burroughs_large_systems`

- NN: *Bytecode* ; `www.economicexpert.com/a/Bytecode.htm`

Fortsetzung

0-Adresskeller

- NN (Wikipedia); *P-code machine* ;
  `en.wikipedia.org/wiki/P-code_machine`
- S.A. Moore: *Pascal-P: The portable Pascal compiler* ;
  `www.moorecad.com/standardpascal/PascalP.html`
- A. Goldberg, D. Robson: *Smalltalk-80 – The language* ; Addison-Wesley, chapter 21 (verschiedene Auflagen!)
  – siehe Referat #13 ! –
- NN (Wikipedia): *Java virtual machine (JVM)* ;
  `en.wikipedia.org/wiki/Java_Virtual_Machine`
- B. Venners: *Bytecode basics – A first look at the bytecodes of the JVM* ; 1996, `www.artima.com/underthehood/bytecode.html`
- NN: *Common language infrastructure CLI* (u.a.);
  `www.economicexpert.com/a/Common:Language:Infrastructure.htm` bzw.
  `www.economicexpert.com/a/Common:Intermediate:Language.htm`
  – siehe Referat #15 ! –

Nebenläufige Programme u.ä.

- (many contributors): *Is there a separate stack for each thread in Java ?* ; www.geekinterview.com/question_details/12828
- G. Lavender: *Concurrent programming using threads (Java)* ; 1999, www.cs.utexas.edu/~lavender/courses/tutorial/java-08.pdf
- R. James: *(einiges BlaBla, dann aber über)* *Iterators* ; 2004, www.thinkingms.com/pensieve/CommentView,guid,e5126078-543f-49bf-a83a-325c11ec7682.aspx
- A. Shankar: *Implementing coroutines for .NET by wrapping the unmanaged fiber API* ; msdn.microsoft.com/de-de/magazine/cc164086(en-us).aspx
- T. Brunklaus, L. Kornstaedt: *A virtual machine for multi-language execution* ; 2002, www.ps.uni-sb.de/Papers/abstracts/multivm.pdf
- N. Williams: *Morphable objects in Smalltalk* (in verteilten Anwendungen); ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=730912&isnumber=15791

Einführung zur Speicherbereinigung und alte („tracing")Verfahren

- NN (Wikipedia): *Garbage collection* ; en.wikipedia.org/wiki/Garbage_collection_(computer_science)
- B.R. Preiss, P. Eng: *Mark-and-sweep garbage collection* ; 1998, www.brpreiss.com/books/opus5/html/page424.html
- B. Zorn: *Comparing mark-and sweep and stop-and-copy garbage collection* , 1990, Proc. ACM Conf. LISP and Functional Programming, 1990, 87 - 98
- L. Fegaras: *Copying garbage collection* ; 2005, lambda.uta.edu/cse5317/notes/node48.html
- H. Azatschi et al.: *An on-the-fly mark and sweep garbage collector based on sliding views* ; 2003, ACM SIGPLAN Notices, 38 (2003) 11, 269 – 281
- NN (namenloser Blogger): *The fully upturned bin - Managing memory and efficient disposal of waste in Ruby* ; 2005, whytheluckystiff.net/articles/theFullyUpturnedBin.html
- C. Höglinger: *Conservative garbage collection (für C)* ; 2006, www.ssw.uni-linz.ac.at/Teaching/Lectures/Sem/2005/Slides/Hoeglinger.pdf

siehe auch JL auf Folie 4

Referenzzähler

Allgemein

siehe auch JL
auf Folie 4

- NN (Wikipedia): *Reference counting* ; `en.wikipedia.org/wiki/Reference_counting`
- L.P. Deutsch, D.G. Bobrow: *An efficient, incremental, automatic garbage collector*; Comm. ACM, 19 (1976) 9, 522 – 526
- D.P. Friedman, D.S. Wise: *The one-bit reference count* ; BIT, 1977 (17) 3, 351 – 359
- D.P. Friedman, D.S. Wise: *Reference counting can manage the circular environments of mutual recursion* ; Information Processing Letters, 8 (1979) 1, 41 – 45

Fortsetzung

Referenzzähler

Python

- NN (Python Software Foundation): Objects, types and reference counts ; 2009, docs.python.org/c-api/intro.html#reference-counts
- NN: *Reference counts* (bei Python); 2008, www.python.org/doc/2.5.2/ext/refcounts.html
- N. Schemenauer: *Garbage collection for Python* ; 2000 arctrix.com/nas/python/gc/

Perl

- D.A.P. Mitchell: *Two-phased garbage collection* (bei Perl) ; search.cpan.org/~dapm/perl-5.10.1/pod/perlobj.pod#Two-Phased_Garbage_Collection
- NN: *Finding circular reference leaks in Perl* ; 2009, letsgetdugg.com/2009/04/19/finding-circular-reference-leaks-in-perl/

moderne Verfahren

- NN (DseWiki): *Garbage collection* ; 2006, **www.wikiservice.at/dse/wiki.cgi?GarbageCollection**
- G. Goos: *Speicherbereinigung* ; 2008, **pp.info.uni-karlsruhe.de/lehre/ws200809/compiler/06-Speicherbereinigung_v1.pdf**
- H.-J. Boehm et al: *Mostly parallel garbage collector* ; ACM SIGPLAN Notices, (1991) 6, 157 – 164
- NN: *Garbage collection auf Multiprozessoren* ; 2001 (etwas veraltet), **www.virtualmachine.de/2000-ernst-schneider/node55.html**
- NN (TLB): *Prozessor mit integrierter, echtzeitfähiger Garbage Collection* (und so ein denglischer Titel kommt von Stuttgart, kaum zu glauben); 2008, ***www.uni-stuttgart.de/forschung/uploads/285.pdf***

Siehe auch JL
auf Folie 4

... bei Smalltalk

Allgemeines und Speicherverwaltung

- NN (Wikipedia): *Smalltalk* ; `de.wikipedia.org/wiki/Smalltalk-80_(Programmiersprache)`
- G. Heeg: *Was ist Smalltalk?* ; undatiert, `www.heeg.ch/smalltalk/main_smalltalk.html`
- NN: *Smalltalk* ; `www.economicexpert.com/a/Smalltalk:programming:language.htm`
- NN (Cincom): *Memory management* ; `www.cincomsmalltalk.com/CincomSmalltalkWiki/DOWNLOAD/DOCS/memory-VW3.x.pdf`
- T. Rowledge: *A tour of the Squeak object engine* ; undatiert, `stephane.ducasse.free.fr/FreeBooks/CollectiveNBlueBook/Rowledge-Final.pdf`

Fortsetzung

... bei Smalltalk

Speicherbereinigung

J.M. McIntosh: *Intro to garbage collection in Smalltalk* ; 1997/2001, `www.smalltalkconsulting.com/papers/GCPaper/GCTalk%202001.htm`
~: *Smalltalk GC theory* (drei ältere Artikel); 1996, `www.smalltalkconsulting.com/papers/papersandpresentations.html`

- D. Ungar: *Generation scavenging - A non-disruptive high performance storage reclamation algorithm* ; ACM SIGPLAN Notices, 19 (1984) 5, 157 - 167 (auch an anderer Stelle)
- ~, F. Jackson: *Tenuring policies for generation-based storage reclamation* ; ACM SIGPLAN Notices, 23 (1988) 11, 1 - 17

... bei C++, Java u.ä.

C++

- B. Zorn: *The measured cost of conservative garbage collection* ; Software – Practice and Experience, 23 (1993) 7, 733 - 756
- D. Detlefs et al.: *Memory allocation costs in large C and C++ programs* ; Software – Practice and Experience, 24 (1994) 6, 527 - 542
- Attardi et al.: *A customisable memory management framework for C++* ; Software – Practice and Experience, 28 (1998) 11, 1143 – 1252
- H. Boehm: *A garbage collector for C and C++* ; undatiert, **www.hpl.hp.com/personal/Hans_Boehm/gc/#details**
- NN (Wikipedia): Boehm garbage collector ; (auch für Thema #15) **en.wikipedia.org/wiki/Boehm_garbage_collector**
- NN: Put garbage collection in C++ ; undatiert, **www.ipetitions.com/petition/garbage_collection_for_cpp/**

Fortsetzung

... bei C++, Java u.ä.

Java

B. Goetz: *Allocation is faster than you think, and getting faster* (Vergleich C++/Java); 2005, `www.ibm.com/developerworks/java/library/j-jtp09275.html`

- M. Persson (& H. Cummins): *Java technology, IBM style - Garbage collection policies* (Part 1 & Part 2) ; 2006, `www.ibm.com/developerwork/java/library/j-imbjava2/index.html?ca=drs-` und `.../j-ibmjava3/...`
- NN: *Garbage collection doesn't mean no memory management* ; 2009, `www.coderfriendly.com/2009/06/14/garbage-collection-doesnt-mean-no-memory-management/`
- NN (SAP): *Memory management (Garbage Collection)* ; undatiert, `help.sap.com/saphelp_nwce10/helpdata/en/2a/1cf5a6737c42158048819a3621d205/content.htm`
- C.E. McDowell: *Reducing garbage in Java* ; ACM SIGPLAN Notices, 33 (1998) 9, 84 - 86

... bei .NET

- NN: *Standard ECMA-335 - Common language infrastructure (CLI)* ; www.ecma-international.org/publications/standards/Ecma-335.htm
- M. Cochran: *C# heap(ing) vs stack(ing) in .NET* (Part I – IV); 2006, www.c-sharpcorner.com/UploadFile/rmcochran/csharp_memory01122006130034PM/csharp_memory.aspx
- J. Puvvala, A. Pota: .NET For Java developers migrating to C# ; 2007, book.javanb.com/NET-For-Java-Developers-Migrating-To-Csharp/0672324024_ch01lev1sec4.html
- NN (Microsoft): *Garbage collection* ; 2007, msdn.microsoft.com/de-de/library/0xy59wtx.aspx
- H. Hunter: *Dangers of the large object heap* ; 2009, www.simple-talk.com/dotnet/.net-framework/the-dangers-of-the-large-object-heap/
- J. Richter: *Garbage collection - Automatic memory management in the Microsoft .NET framework* ; 2002, msdn.microsoft.com/en-us/magazine/bb985010.aspx

HJH#16
2009/2010

... bei Stöberern u.ä.

- E. Lippert: *How do the script garbage collectors work?* ; 2003, `blogs.msdn.com/ericlippert/archive/2003/09/17/53038.aspx`
- Anonymus (Pavlov): *Firefox 3 memory usage* ; 2008, `blog.pavlov.net/2008/03/11/firefox-3-memory-usage/`
- D. Almaer: *Garbage collection in IE6* ; 2007, `ajaxian.com/archives/garbage-collection-in-ie6`
- C. Stockwell: *IE8 performance* ; 2008, `blogs.msdn.com/ie/archive/2008/08/26/ie8-performance.aspx`
- G. Skinner: *Understanding garbage collection in Flash Player 9* ; 2007, `www.adobe.com/devnet/flashplayer/articles/garbage_collection.html`
- Alex Harui: *Garbage collection and memory leaks* (in Flash); 2007, `blogs.adobe.com/aharui/2007/03/garbage_collection_and_memory.html`
- A. Roehrl et al.: *Garbage collection* (bei Ruby); 2002, `http://www.approximity.com/rubybuch2/node189.html`

HJH#??
2009/2010

Zur Vertiefung

unter anderem:

! im Deutschen ohne Bindestrich

- International Symposium on Memory Management, ISSM ab 1998 über ACM SIGPLAN Notices (in Bibliothek vorhanden!), www.sigplan.org/ismm.htm
- Conference on Object-Oriented Programming Systems, OOPSLA, über ACM SIGPLAN Notices (in Bibliothek vorhanden!), u.a.: J. Gil, P.F. Sweeney: Space- and time-efficient memory layout for multiple inheritance ; SNOT 34 (1999) 10, 256 - 275
- andere Konferenzserien, publiziert bei den ACM SIGPLAN Notices (in Bibliothek vorhanden!)
- ACM Transactions on Programming Languages and Systems, TOPLAS (in Bibliothek vorhanden!), toplas.acm.org/
- IEEE Software, www2.computer.org/portal/web/software

- R. Jones: *The garbage collection bibliography* ; 1996-2009, www.cs.kent.ac.uk/people/staff/rej/gcbib/gcbibG.html

“Allgemeine Geschäftsbedingungen”

- Referat maximal 40 Minuten
- Datenprojektor („*beamer*") ist vorhanden
- Anschließend Diskussion
  - inhaltlich
  - Vortragsstil
- Auf Anforderung, mindestens fünf Tage vorher, kann PC für Präsentation mit PowerPoint bereitgestellt werden
- Ausarbeitung maximal acht bis zehn Seiten DIN A4;
  (in der Regel) nach einer Woche abzugeben (Papier);
  Literaturangaben nicht vergessen!
- Nach Bestätigung (eventl. Korrekturwünsche) durch Betreuer Abgabe der endgültigen Ausarbeitung innerhalb einer Woche als doc-/pdf-Datei
  (maximal 2 Mbyte groß, mit Namen “WS09-10-#xx-version-ii.<ext>”, “xx” elem {01 ÷ 15}, “ii” elem {1 ...})

Ein Hinweis des Dekans:

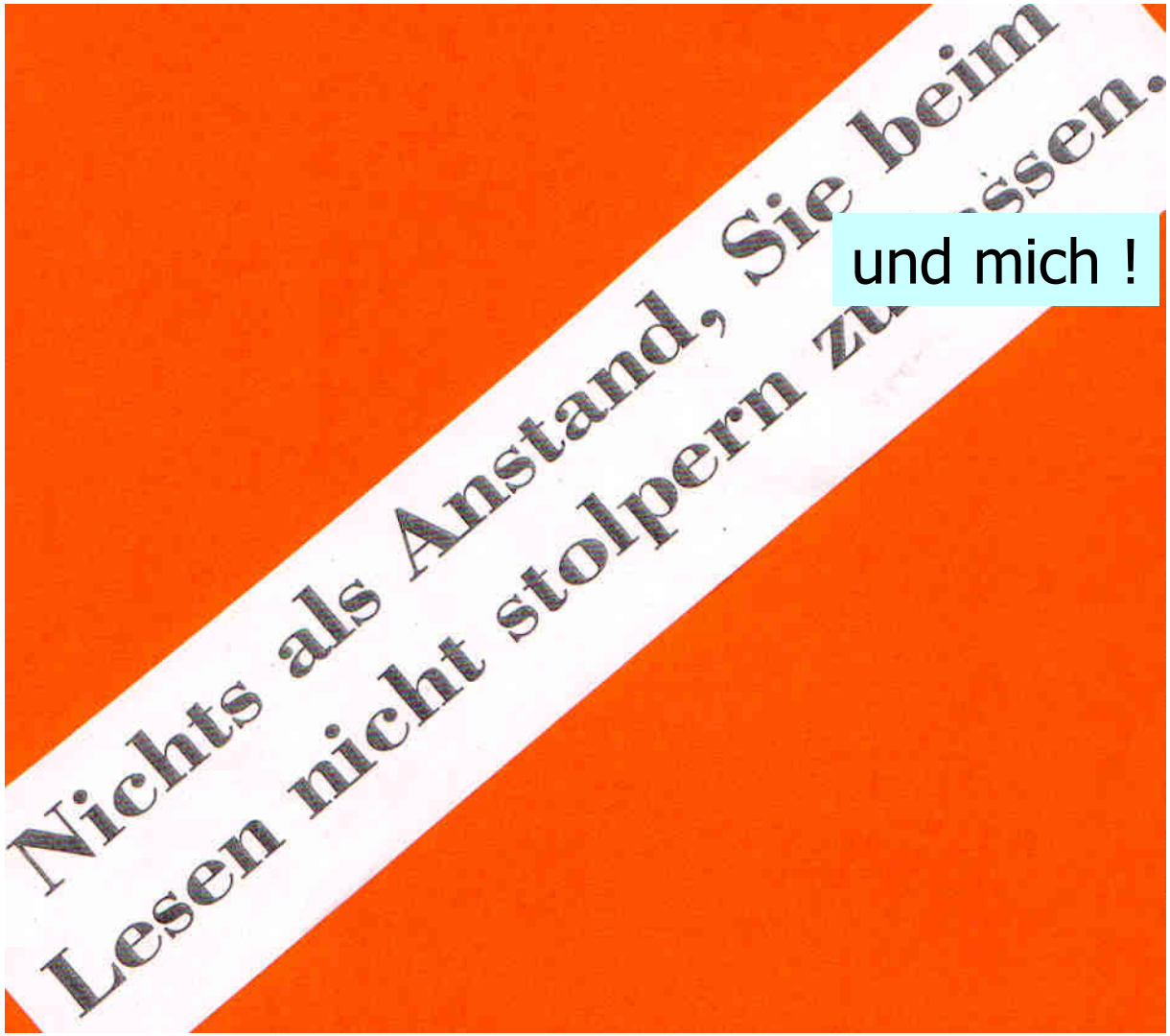
"Der Fachbereich Informatik mißt der Einhaltung der Grundregeln der wissenschaftlichen Ethik großen Wert bei. Zu diesen gehört auch das strikte Verfolgen von Plagiarismus. Mit der Abgabe einer Lösung (Hausaufgabe, Programmierprojekt, Seminararbeit Diplomarbeit, etc.) bestätigen Sie, daß (Sie/Ihre Gruppe) der alleinige Autor/die alleinigen Autoren des gesamten Materials sind. Falls Ihnen die Verwendung von Fremdmaterial gestattet war, so müssen Sie dessen Quellen deutlich zitiert haben.

Bei Unklarheiten zu diesem Thema finden Sie weiterführende Informationen unter **www.informatik.tu-darmstadt.de/Plagiarism** oder sprechen Sie Ihren Betreuer an."

Web-Seite des Seminars:

http://www.pu.informatik.tu-darmstadt.de/Seminar-Speichern/

(dort findet sich u.a. die Vorlage für das
Deckblatt Ihrer Ausarbeitung
und der **Terminplan** !)


Nichts als Anstand, Sie beim
Lesen nicht stolpern
und mich !

Fortsetzung

HJH#??
2009/2010

??