PLAGIARISM

The use of copied software in the Computer Science Department

Adopted by Computer Science Subdepartment, February 27, 1997

Plagiarism involves taking ideas or other forms of intellectual effort and passing them off as one's own [Guralnik, 68]. A specific example of plagiarism is using a program found in a textbook as *your* solution to a programming assignment, i.e., without mention of the original author and source of the program.

The issue of plagiarism is of particular concern in the Computer Science department, due to the importance placed on the creation of computer programs in most courses. According to University regulations, sanctions for students who engage in plagiarism can be quite serious. These sanctions may include the awarding of a failing grade to the student, and, in the most serious cases, the expulsion of the student from the University. Since most definitions and guidelines relating to plagiarism refer to works in written and spoken language, the faculty members in the Computer Science Department believed it would be helpful if they interpreted some of these guidelines within the context of computer programming.

The Indiana University Code of Student Ethics provides the following definition of plagiarism as part of its discussion of academic misconduct [IU, 93, p17,]:

A student must not adopt or reproduce ideas, words, or statements of another person without an appropriate acknowledgement. A student must give due credit to the originality of others and acknowledge an indebtedness whenever he or she does any of the following:

- Quotes another person's actual words, either oral or written
- Paraphrases another person's words, either oral or written
- Uses another person's idea, opinion, or theory
- Borrows facts, statistics, or other illustrative material, unless the information is common knowledge.

We can interpret the above as meaning that when using programs and program fragments of others, or when using programs based on the ideas, theories, or algorithms of others, one should give due credit to the originator of these intellectual efforts.

The balance of this note will discuss *how* due credit is to be given, what kinds of things one is permitted to copy (with a brief discussion of copyright law), and when there might be a conflict between copying another's code and the educational objectives of an assignment.

How is due credit given?

Due credit is given by clearly indicating within the source code:

1. The original author of the code or algorithm.
2. The source where this code or algorithm was obtained. If the source is a textbook, then provide a full bibliographic citation, including the name of the text, the author, the publisher, edition (if not the first), date, and page number of the algorithm/code. If the source is another student or the instructor, the name of the original programmer should be provided.
3. A description of any alterations that are made to the code or algorithm by the current programmer.

In general, a programmer should document her code so as to clearly indicate what parts of the program are written by her, and what parts are written by others. In addition, it is customary to provide documentation indicating if one is using one's *own* code obtained from a previously written program. This way, a reader can distinguish between newly created and reused work. The Appendix provides a number of example programs that illustrate the use of these citation guidelines.

Copying and educational objectives

In general each assignment in a course has a set of educational objectives associated with it. There are currently only rare instances in which the objective is to locate and combine the works of others, adding little if anything new of one's own. Most assignments are intended to require a large amount of novel expression by the student.

The practical implications of this are that both the student and the instructor have a shared, although different, responsibility in ensuring that the objectives of the assignment are met, and in determining what and how much code may be copied to still meet the course objectives. The instructor's responsibility is to make the objectives clear, as well as to provide guidelines concerning the pieces of code that can, and possibly *must*, be copied from others. The instructor also is charged with monitoring the copying and citation done by students. The student's responsibility is to interpret the objectives and guidelines provided by the instructor, to actively seek clarification if necessary, to engage in and encourage others to engage in ethical behavior pertaining to copying (including the instructor), and to self-monitor one's actions in this regard.

As a general guideline, before including the work of another in one's programming assignment, it is prudent to first check with the instructor.

Here are two examples where copying plays a dramatically different role in relation to the educational objectives of an assignment. In the first case it is not appropriate, while in the second case it is appropriate.

1. An assignment is given in which you are to write a subroutine that takes an `int` as a parameter and outputs the binary encoding of this parameter. Suppose that you find a solution to this problem in a textbook from the library. Should you use this solution, giving the proper citations? Although we cannot be sure about the specific objective of this assignment, in the absence of information to the contrary, we can assume that the instructor wants you to develop the function yourself, based on your understanding of number representations. Thus, using the textbook program would violate the educational objectives of the assignment.
2. An assignment is given in which you are to develop a `graph` class that provides such operations as `print_nodes` and `current_edge_value`. Suppose that in doing this assignment, you would like to reuse code that you developed in the data structures class for linked lists and hash tables. It is fair to assume that the objectives of this assignment do *not* include your developing linked list and hash table implementations; rather, the objectives are that you use known data structures in order to build a larger, more complex program. Thus, in this example, copying your previously written code is well within the educational objectives.

Permissions and copyright

An additional issue concerns whether an individual has *permission*, either explicitly or implicitly, to use another person's code or ideas. The following discussion assumes that the code under discussion is published in a textbook or other printed form, or is publicly readable online via the ftp or http protocol.

We can use as a guideline the laws that exist in the United States pertaining to copyright. According to copyright law, *ideas* themselves are not copyrightable, and therefore anyone can freely use the ideas of another. As indicated above, within the academic setting, the use of another's ideas should always include a citation.

For code obtained from the instructor in an assignment, permission is implicitly given to the students to use this code, and in fact, the use of the instructor's code is often required.

Code obtained from the net or published in texts may have explicit permissions allowing another person to copy and use the code, although there may be restrictions. For example, the Free Software Foundation, who distribute the GNU C++ compiler that is used on department computers, provide free access to their source code under what they call a *copyleft* agreement, which provides for significant copying, as long as subsequent distribution of the copied code provides due credits and does not place further restrictions on copying. It is much more common for the code either to have no explicit permissions or restrictions, or to be copyrighted. As a general guideline, treat code without explicit permissions as copyrighted code.

Copyright law states that only the copyright holder (e.g., the software author, the software company, the text publisher) has permission to make copies of the copyrighted material. However, there is a provision written into this law that enables individuals to make their own copies of another's work under certain circumstances. This provision is called the *Fair Use Statute*. Much of the following discussion is a paraphrase of portions of the paper *Fair Use: Overview and Meaning for Higher Education*, by Kenneth Crews, Associate Professor at IUPUI and Director of the Copyright Management Center. This paper can be obtained at

`http://www.iupui.edu/it/copyinfo/highered.html`

The copying of code (or any other copyrighted material) under the Fair Use Statute requires the consideration of four factors:

Purpose:
: what is the purpose of the copying? In general, copying for educational and private research purposes has been protected by the courts.

Nature:
: what is the nature of the copyright work? Courts have been favorably inclined toward the copying of non-fiction works that are currently in-print, which is true of most code that students will want to copy.

Amount:
: how much of the work is copied? In general, copying small amounts of printed work (relative to the size of the work as a whole) has received court protection.

Market effect:
: what effect will the work have on the market? Crews writes

> This factor means fundamentally that if you make a use for which a purchase of an original theoretically should have occurred--regardless of your personal willingness or ability to pay for such purchase--then this factor may weigh against fair use. ... If your purpose is research or scholarship, market effect may be difficult to prove. ... Occasional quotations or photocopies may have no adverse market effects, but reproductions of software and videotapes can make direct inroads on the potential markets for those works.

In summary, in most cases of student use of copyright software in programming assignments, all of these conditions are met, and the student may make the copies. However, in cases where the code may be subsequently marketed or where substantial copying occurs, one should be more cautious.

Additional information on copyright, particularly as it pertains to higher education, can be found at the URL provided by the Copyright Management Center at IUPUI:

`http://www.iupui.edu/it/copyinfo/home.html`

Summary

The use of copied code in programming assignments should be clearly documented within the code so as to enable a reader to determine who wrote each portion of the code, and should be done within the constraints of the educational objectives of the programming assignment. If you have any questions regarding this policy, please do not hesitate to discuss them with any of the faculty members.

References

Guralnik, 68
David Guralnik, Editor. Webster's new world dictionary of the American language. The World PuTexinblishing Company, 1968.

IU, 93
Indiana University Code of Student Ethics, 1993

Appendix

1. A student is asked to write a C++ class that implements a linked list. Much of the implementation is borrowed from a data structures textbook. Here is how the citation might look:

```
// int_list.cc

// Class: Integer list
//
// Programmer    : Jake deStudent
// Create Date   : 3 Dec 94
// Last modified : 16 Mar 95
//
// Adapted from the list class from the text
//    _Data Structures and Algorithm Analysis in C++_
// by Mark Weiss, published by Benjamin Cummings, 1994, pp. 62-71.
//
// Changes: Added copy constructor, #define INT_LIST, jdt 3/16/95
// All other code used verbatim from Weiss
.
.
.
/*    ========
      Action     :  Return whether list is empty or not
      Programmer :  Mark Allen Weiss
      Source     :   _Data Structures and Algorithm Analysis in C++_
                    published by Benjamin Cummings, 1994, p.62
      Adaptation :  Changed from inline procedure by jdt for info
                         hiding.
      Parameters :  None passed in or out
      Notes      :  Assumes dummy header node
*/
int_List::Is_Empty( )
{
  return (List_Head -> Next == NULL) ;
}
```

2. An instructor provides students with a main program, and asks them to write several functions that are called by main. When turning in homework, each student should cite both the instructor and herself as the programmers:

```
// homework_4.cc

//
// Programmer    : Sally Student and Jane Instructor
// Create Date   : 5 Feb 97
//
// This program takes as input  ...
// { Remainder of description of program }

//
// The main program was written by Jane Instructor, and handed out to
// the students on 22 Jan 97 in the description to Homework 4 for
// C201.  Functions prompt_for_input and calc_payment were written by
// Sally Student.  The function display_payments was written by Jane
// Instructor.
```

In order to clearly indicate authorship, it is sometimes necessary to document who has authored each individual function in a manner similar to the following, in addition to documenting the different programmers within the program header. This is particularly the case in a large project authored by several different programmers.

```
   // Programmer: Sally Student
   // {Description of function, and other header information}
   void prompt_for_input (double initial_balance, int number_of_years,
                          double interest_rate)
   {
      /* function body */
   }

   // Programmer: Jane Instructor
   // Source    : Description in handout for Homework 4
   // Adaptations: None
   // {Description of function, and other header information}
   void display_payments (double initial_balance, int number_of_years,
                          double interest_rate)
   {
      /* function body */
   }
```

3. You want to write a windows-based tic-tac-toe program for your own enjoyment. You find a nifty solution on the World Wide Web that someone has written in Java, that you adapt to C++ and X-windows. It will reinforce professional practice to cite the sources from which you obtained the code, and document the changes that you made.

```
// tic-tac-toe.cc

// Programmers: Joe Student and Jose Javaguru
//
// This program is an adaptation by Joe Student of a Java program
// written by Jose Javaguru.  The Java program was obtained from the
// website http://www.javajunkie.joe/~javaguru/waycool/tictactoe
//
// The design of the main program and the main subroutines were taken
// from Javaguru's code.  The port from Java to C++/X-windows was done
// by Joe Student.  Due to this port, functions display_board,
// input_move, and new_board had to be completely rewritten.
//
// { More details describing changes that needed to be made }
//
```

4. You are asked to write a program that takes as input a set of pairs of courses, where a pair **<C1,C2>** indicates that course **C1** is a prerequisite of course **C2**, and outputs whether there is a *cycle* in the prerequisites for any set of courses, i.e., whether some course is a prerequisite for itself. You write a program based on the *topological sorting* algorithm described in one of your textbooks, but for which code is not explicitly presented.

```
// course_cycle.cc
//
// Programmer    : Chris van Student
// Create Date   : 9 Apr 96
// Last modified : 16 Apr 96

// {Description of general problem}
//
// The central subroutine of this program is check_for_cycle,
// which is an implementation of the topological sorting algorithm
// described in the text
//    _Foundations of Computer Science, C edition_
// by Alfred Aho and Jeffrey Ullman, published by Computer Science
// Press, 1995, pp. 497-498.
```

5. You adapt someone else's code that has citations to a *second* (or third or fourth ...) person's code. In this case, you should preserve the original citations within your code, adding comments indicating which particular additions are yours. Sometimes this needs to be done on a line-by-line basis, if different parts of a function are written by different people over time. For example, here is a fragment of the citations from code handed in by a student in the C431 Compilers course (adapted by deleting several comments and adding citations for original programmer):

```
/*
semacts.c

This file contains the semantic action routines for the MACRO
compiler.

Original programmer:
                      Provided by publisher (Benjamin Cummings) to
                      accompany the text _Crafting a Compiler with C_ by
                      Fischer and Leblanc, 1991.  No programmer
                      identified.  Also obtainable via anonymous ftp
                      from kaese.cs.wisc.edu.

8/15/92  Josh Tenenberg - Commented out the #ifdef TURBOC stuff
10/25/94 Byron Miller, -  Added IntConstType to Semantic
                          declarations and routines
10/25/94 Byron Miller, -  Added IntConstRec to Semantic declarations
                          and routines
03/25/96 Robert Wiseman - Added convert_type Semantic routine
03/25/96 Robert Wiseman - Added process_ids Semantic routine
03/25/96 Robert Wiseman - Added predefined types integer and !error to
                          Semantic routine InitializeSemActions
*/
```

6. You develop code yourself without using code from any external sources. In this case, list yourself as the sole author. The following example is written in the Scheme programming language, and the lines beginning with `;' denote comments.

```
;;; Author: Josh Tenenberg
;;; Create date: 10/29/96
;;;
;;; Action: Takes a problem as input, and runs each instance
(define run-problem
  (lambda (prob)
    (writeln "*** Testing function " (problem->print-label prob) " ***")
    (newline) (newline)
    (for-each
     (lambda (inst)
       (run-instance inst (problem->thunk prob) (problem->comp prob)))
     (problem->instances prob))))
```

7. You develop code from an algorithm that is common knowledge within computer science. For example, suppose a program that you are writing needs a function to find a minimal element in a binary search tree. In this case, use the following rule-of-thumb as to whom to cite. If you develop the code yourself, then you need not cite any other sources. If you use one or more texts or other code sources as a reference, then cite these texts. Here is the code that I developed for this find-min problem in Scheme, making reference to the text that I used in its development:

```
;;; Author: Josh Tenenberg
;;; Create date: 9/22/96
;;; Algorithm referenced from text:
;;;    Author: Mark Allen Weiss
;;;    Source: _Data Structures and Algorithm Analysis in C++_
;;;            published by Benjamin Cummings, 1994, p.131
;;;    Adaptation :  changed from C++ to scheme
;;;
;;; Action: Find a minimal element in a BST

    (define find-min
      (lambda (tree)
        (cond
          ((empty-tree? tree) '())
          ((empty-tree? (left-subtree tree)) (root tree))
          (else (find-min (left-subtree tree))))))
```