How to Avoid Infringing the Copyright of a Computer Program: From the Perspective of a Computer Program Turned Attorney/Law Professor

David C. Tunick
Loyola Law School
HOW TO AVOID INFRINGING THE COPYRIGHT OF A COMPUTER PROGRAM: FROM THE PERSPECTIVE OF A COMPUTER PROGRAMMER TURNED ATTORNEY/LAW PROFESSOR

David C. Tunick*

I. INTRODUCTION

This Article deals with copyright and computer programs. Two recent cases have provided guidelines for determining when a copyright infringement of computer programs has occurred. The cases are Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.1 and Computer Associates International, Inc. v. Altai, Inc.2 This Article attempts to use the guidelines from those two cases to show how to write a computer program whose basic function is the same as an earlier program so that neither its code, nor its output infringes the earlier program.3 This is an important question when a programmer working for Company A quits in order to work for Company B. Company B might exclude the programmer from projects similar to those on which the programmer worked at Company A in order to avoid possible copyright or trade secret infringements.4 However, if the programmer is


3. The copyright on the screen and printer output is separate from the copyright on the code. Computer Assocs., 982 F.2d at 703.
4. See id. at 700 (explaining that Altai excluded a former employee of Computer Associates, who now worked for Altai, from writing software on a project similar to Computer Associates’ project). A “trade secret” might consist of “any formula, pattern, device, plan, or compilation of information which is used in one’s business, and which gives him an
properly advised on how not to infringe the rights of Company A, Company B might be able to use the programmer on projects similar to those of Company A. In this way, Company B could save time and money by using nonproprietary ideas which the programmer acquired while working at Company A. Two hypothetical programs are set forth in this Article to demonstrate how to create a noninfringing program that performs functions similar to a pre-existing program.

But first, a brief examination of the history of copyright and computer programs is useful to show how the issue of copyrightability of computer programs arises.

II. THE LAW

A. THE NATIONAL COMMISSION ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS (CONTU)

In 1976, CONTU was created by Congress as part of an effort to revise the United States copyright laws. It became apparent to Congress that the new computer technology was creating copyright concerns, so the Commission was established to recommend changes to the copyright laws that would respect the rights of copyright owners while also considering concerns of consumers and the public. The Commission studied computers and copyrights and issued its report on July 31, 1978.

The Commission said that "[c]omputer programs are the product of great intellectual effort and their utility is unquestionable" and recommended copyright protection for various forms of the
programs, including flowcharts, source code, and object code.

The Commission considered and rejected both trade secret protection and patent protection for computer programs. Trade secrets were said to consist of a "formula, pattern, device, or compilation of information, [which give proprietors] an opportunity to obtain an advantage over competitors who do not know or use it." The Commission, however, rejected trade secrecy since the secret is lost upon wide distribution, thus precluding trade secrecy for programs widely sold. Furthermore, since trade secrecy is a creature of state law, the lack of uniform national law reduces its utility.

Patent protection was also rejected. The Commission predicted that even if patent protection was available, only a few programs could survive the rigorous application and appeals procedure which require that the programs be novel.

The Commission recommended that the definition of "computer program" be added to the copyright law and that rightful possessors of computer programs be allowed to have the programs "read into" (i.e., copied into) the computer in order to use the computer and to make back-up copies, and Congress complied.

---

11 Id. at 21.
12 A flowchart is a graphic representation for the definition, analysis or solution of a problem in which symbols are used to represent operations, data flow, or equipment. Id. at 21 n.109.
13 [S]ource code is a computer program written in any of several programming languages employed by computer programmers. Id. at 21 n.109.
14 [O]bject code is the version of a program in which the source code language is converted or translated into the machine language of the computer with which it is to be used. Id.
15 CONTU, supra note 6, at 16-19.
16 Id. at 16 (citing to RESTATEMENT OF TORTS § 757 cmt. b (1939)).
17 Id. at 17.
18 Id.
19 Id.
21 Id. at 12. Congress added the following definition to Section 101 of the Copyright Act: "A 'computer program' is a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result." 17 U.S.C. § 101 (1994) (originally enacted as Act of Dec. 12, 1980, Pub. L. No. 96-517, § 10(a), 94 Stat. 3028).
22 CONTU, supra note 6, at 12-13.
CONTU also suggested that various forms of computer programs be given copyright protection, including flowcharts, source code, and object code.

This Article will review three cases dealing with infringements. The first deals with infringement of object code; the second and third cases deal with infringement of source code.

B. COPYRIGHT PROTECTION OF OBJECT CODE—APPLE V. FRANKLIN

Both Apple and Franklin manufactured computers. Franklin copied several of Apple's programs in their entirety to sell with Franklin computers. Franklin did this in order to allow its computers to be able to run programs developed for Apple comput-

23 Congress added the following to the Copyright Act:
§ 117. Limitations on exclusive rights: computer programs
Notwithstanding the provisions of section 106, it is not an infringement for the owner of a copy of a computer program to make or authorize the making of another copy or adaptation of that computer program provided:
(1) that such a new copy or adaptation is created as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner, or
(2) that such new copy or adaptation is for archival purposes only and that all archival copies are destroyed in the event that continued possession of the computer program should cease to be rightful.

Any exact copies prepared in accordance with the provisions of this section may be leased, sold, or otherwise transferred, along with the copy from which such copies were prepared, only as part of the lease, sale, or other transfer of all rights in the program. Adaptations so prepared may be transferred only with the authorization of the copyright owner.


24 See supra note 12 and accompanying text (defining flowcharts).
25 See supra note 13 and accompanying text (defining source code).
26 See supra note 14 and accompanying text (defining object code).
28 Id. at 1242-43.
29 Id. at 1245.
30 Id.
ers. The question arose whether computer programs expressed in object code existing in the memory of the computer could be protected by copyright. The lower court had questioned whether copyright was limited to works capable of being read by a human. On appeal, the Third Circuit noted that "[t]he suggestion that copyrightability depends on a communicative function to individuals stems from the early decision of White-Smith Music Publishing Co. v. Apollo Co."

In rejecting the lower court's argument, the Third Circuit said that "copyright extends to works in any tangible means of expression 'from which they can be perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device.' " Since the computer program residing in the computer's memory could be printed and then perceived, it was eligible for copyright protection.

The court also said that computer programs are eligible for copyright protection as "literary works." The court added, "Thus a computer program, whether in object code or source code, is a 'literary work' and is protected from unauthorized copying, whether from its object or source code version."

Apple presents the easiest of cases because Franklin copied entire programs. Once it is determined that copyright protection exists for computer programs, it is easy to say that the copying of an entire program constitutes an infringement.

The next two cases, Whelan and Computer Associates, present more difficult issues because only portions of programs, not

---

31 Id. at 1243.
32 Apple Computer, 714 F.2d at 1246-47.
33 Id. at 1248.
34 Id. (citing White-Smith Music, 209 U.S. 1 (1908)).
35 Id. at 1248 (quoting 17 U.S.C. § 102(a)).
36 Id. at 1249. CONTU also believed that computer programs qualify as literary works. CONTU, supra note 6, at 16. Copyright protects literary works under 17 U.S.C. § 102(a)(1) (1994).
37 Apple Computer, 714 F.2d at 1249.
38 See supra notes 28-32 and accompanying text (summarizing the facts of Apple Computer).
39 The program must be original to its author and fixed in a tangible medium of expression. 17 U.S.C. § 102(a) (1994).
40 Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222 (3d Cir. 1986).
entire programs, were copied. Resolution required the courts to
determine which aspects of computer programs are eligible for
copyright protection.\textsuperscript{42}

C. COPYRIGHT PROTECTION FOR SOURCE CODE

1. Whelan v. Jaslow.\textsuperscript{43} Whelan Associates developed computer
programs to be used by dental laboratories.\textsuperscript{44} Whelan entered into
an agreement with Jaslow Dental Laboratory by which Jaslow
would market Whelan's programs.\textsuperscript{45} The programs were deliv-
ered by Whelan to Jaslow,\textsuperscript{46} and within two years, Jaslow had
developed similar dental programs.\textsuperscript{47} Whelan alleged that
Jaslow's programs infringed Whelan's copyright.\textsuperscript{48}

The Third Circuit said that it must "determine whether the
structure (or sequence and organization) of a computer program is
protectible by copyright, or whether the protection of copyright law
extends only as far as the literal computer code."\textsuperscript{49}

Prior to making its determination of whether copyright infringe-
ment occurred, the court noted that "[A]s it is rarely possible to
prove copying through direct evidence, [citation omitted] copying
may be proved inferentially by showing that the defendant had
access to the allegedly infringed copyrighted work, and that the
allegedly infringing work is substantially similar to the copyrighted
work."\textsuperscript{50}

In reaching its conclusion that an infringement occurred,\textsuperscript{51} the
court noted the following:

- "[C]opyright protection extends to a program's
  source code and object code."\textsuperscript{52}

\textsuperscript{42} Whelan, 797 F.2d at 1233-48; Computer Assocs., 982 F.2d at 701-15.

\textsuperscript{43} 797 F.2d 1222 (3d Cir. 1986).

\textsuperscript{44} Id. at 1225.

\textsuperscript{45} Id. at 1225-26.

\textsuperscript{46} Id. at 1226.

\textsuperscript{47} Id.

\textsuperscript{48} Whelan, 797 F.2d at 1226.

\textsuperscript{49} Id. at 1224. The court added: "We use the terms 'structure,' 'sequence,' and
'organization' interchangeably when referring to computer programs, and we intend them
to be synonymous in this opinion." Id. at 1224 n.1.

\textsuperscript{50} See Whelan, 797 F.2d at 1231-32.

\textsuperscript{51} Id. at 1224, 1248.

\textsuperscript{52} Id. at 1233. See supra notes 13 and 14 for definitions of "source code" and "object code."
Scenes a faire are ‘incidents, characters or settings which are as a practical matter indispensable . . . in the treatment of a given topic.’ [Citations omitted.] It is well settled doctrine that scenes a faire are afforded no copyright protection . . . because the subject matter represented can be expressed in no other way than through the particular scene a faire. Therefore, granting a copyright ‘would give the first author a monopoly on the commonplace ideas behind the scenes a faire.’ ” [Citation omitted.]

The copyrights of other literary works can be infringed even when there is no substantial similarity between the works’ literal elements. One can violate the copyright of a play or book by copying its plot or plot devices. By analogy, it appears “that the copyrights of computer programs can be infringed even absent copying of the literal elements of the program.”

The structure of the infringed dental program was not essential to its task of aiding dental businesses. There were other dental programs performing the same functions but with different structures.

Mere ideas are not copyrightable, but the manner in which the idea is expressed can be copyrighted. Therefore, the idea of a computer program for operating a dental laboratory is not eligible for copyright protection, but the detailed structure of a dental program is part of the copyrightable expression of the idea.

53 Whelan, 797 F.2d at 1236.
54 Id. at 1234 (citing Twentieth Century-Fox Film Corp. v. MCA, Inc., 715 F.2d 1327, 1329 (9th Cir. 1983), which noted that 13 alleged distinctive plot similarities between two movies may be the basis for finding copyright violation).
55 Id.
56 Id. at 1238.
57 Id. at 1238-39.
The organization of a computer program is copyrightable.\(^{58}\)

"File structures" are storage places for data, similar to a file drawer or a manila folder.\(^{59}\) These file structures are copyrightable.\(^{60}\)

Copyright infringement may be shown if the supposedly infringing program accomplishes about the same results as the infringed program in about the same way—using similar file structures, similar sequences of operating, and using similar names within the program to identify information.\(^{61}\)

Thus, \textit{Whelan} makes many points about copyright and computer software. \textit{Computer Associates} will be discussed next. Rules may be extracted from these cases to be used in determining if one hypothetical program infringes another.

2. \textit{Computer Associates v. Altai}.\(^{62}\) Claude Arney worked as a computer programmer for Computer Associates. James Williams worked for Altai. Williams recruited Arney, his lifelong friend, to work for Altai. When Arney left Computer Associates, Arney brought with him to Altai copies of source code on which he worked for Computer Associates. It was unknown to anyone else at Altai that Arney had brought the source code with him. Arney wrote programs for Altai, referring to the source code he had taken with him. Computer Associates learned that Altai may have appropriated some of Computer Associates’ software and sued Altai for infringement. Upon being sued, Altai first learned from Arney that he had copied from Computer Associates’ source code. Altai rewrote the programs using programmers other than Arney.\(^{63}\)

\(^{58}\) \textit{Whelan}, 797 F.2d at 1239.

\(^{59}\) \textit{Id.} at 1242.

\(^{60}\) \textit{Id.} at 1243.

\(^{61}\) \textit{Id.} at 1247-48.


\(^{63}\) \textit{Computer Assocs.}, 982 F.2d at 699-700.
Altai conceded that it had infringed the programs that Arney copied. However, the lower court found, and the Second Circuit agreed, that those programs written by Altai programmers other than Arney were not substantially similar to Computer Associates' programs, and thus did not infringe.\footnote{Computer Assoc., 775 F. Supp. at 560-62.}

In reaching its conclusion, the Second Circuit noted:

- "Literal elements of computer programs, i.e., their source and object codes are the subject of copyright protection."\footnote{Computer Assoc., 982 F.2d at 702.}

- "[A] program's structure includes its non-literal components such as general flow charts\footnote{See supra note 12 (defining "flow chart").} as well as the more specific organization of inter-modular relationships,\footnote{A "module" or "subroutine" is a portion of the program responsible for a certain task, such as updating accounts receivable. Computer Assoc., 982 F.2d at 697.} parameter lists,\footnote{A "parameter list" consists of information sent to and received from a subroutine, for example, for accounts receivable, the time frame and customer identification number. Id.} and macros.\footnote{A "macro" is "a single instruction that initiates a sequence of operations." Id. at 698.}

- Copyright does not protect an idea, only the expression of the idea.\footnote{Id. at 703.}

- The court in Whelan failed to account for the difference between a program's "static structure" and "dynamic structure."\footnote{Id. at 705.} This point was discussed in more depth by the district court in Computer Associates, which said that the sequence of operations (the behavior) of a program can be obtained by many different sequences of instructions in the program (the text). Thus, there is no necessary relationship between the sequence of the text and the sequence of the program's dynamic behavior. The Whelan court's assertion that a program's sequence is...
protected by copyright is ambiguous. Does the sequence refer to the text or to the behavior? Furthermore, since Section 102 (a) of the Copyright Act excludes a “process” and “method of operation” from protection, and since the behavior of a computer program falls within these terms, the program’s behavior is excluded from copyright protection.  

Copyright infringement is proved “by showing that (1) the defendant had access to the plaintiff’s copyrighted work and (2) the defendant’s work is substantially similar to the plaintiff’s copyrightable material.”  

The framework for analyzing substantial similarity consists of three steps: (1) Abstraction, (2) Filtration, and (3) Comparison. The Second Circuit described this three step process.

**Step One: Abstraction.** With any work, there will be a great number of patterns that will fit. The last abstraction may be no more than the title. At some point in the series of abstractions, there no longer is protection because there are only ideas. With computer programs, the lowest level of abstraction may

---

72 *Computer Assocs.,* 775 F. Supp. at 559-60. The *Computer Associates* district court failed to explain why a program’s behavior can be considered a “process.” In patent cases, “process” has been defined as “an act or series of acts, performed upon the subject-matter to be transformed and reduced to a different state or thing.” *Gottschalk v. Benson,* 409 U.S. 63, 70 (1972) (citing Cochrane v. Deener, 94 U.S. 780, 787-788 (1876)). An example would be the manufacturer of “fat acids and glycerine from fatty bodies by the action of water at a high temperature and pressure.” *Id.* at 70 (citing Tilghman v. Proctor, 102 U.S. 707, 721 (1880)). If the term “process” has the same meaning in copyright and patent law (a reason that a “process” may be ineligible for copyright is that it is protected by patent law. 35 U.S.C. §§ 100, 101), then the lower court in *Computer Associates* possibly could have explained why a computer program is a “process,” which means the program transforms matter from one state to another. *Gottschalk,* 409 U.S. at 70. Since computer programs generally are not thought of as transforming matter, it seems doubtful that programs are “processes.”  

73 *Computer Assocs.,* 982 F.2d at 701.  
74 *Id.* at 706-11.
be the set of instructions and subroutines. At a higher level, the subroutines may be replaced by the function of the subroutines. Ultimately, the abstractions process may result in the purpose of the program.\textsuperscript{76}

**Step Two: Filtration.** This process entails examining each level of abstraction to determine whether the inclusion at such level is an “idea” whose elements are:

(a) elements dictated by efficiency: when there is only one way, or a small number of efficient ways to accomplish a task, the expression merges with the idea and is not protectible;\textsuperscript{76}

(b) elements dictated by external factors: examples are mechanical specifications of the computer, compatibility with other programs, and widely accepted programming practices;\textsuperscript{77} or

(c) elements taken from the public domain: such material is free for the taking even if included in a copyrighted work.\textsuperscript{78}

If the inclusion is deemed to fall into any of the above categories, then there is a nonprotectible expression.\textsuperscript{79}

**Step Three: Comparison.** Once nonprotectible elements are sifted out,\textsuperscript{80} there may remain protectible expression. At this point, the substantial simi-

\textsuperscript{76}Id. at 706-07.  
\textsuperscript{77}Id. at 707-08.  
\textsuperscript{78}Computer Assocs., 982 F.2d at 710.  
\textsuperscript{79}Id. at 707.  
\textsuperscript{80}Such nonprotectible elements are those ideas dictated by efficiency or external factors, or those taken from the public domain. Id. at 710.
larity inquiry takes place in order to determine if infringement occurred.81

D. RULES FOR DETERMINING INFRINGEMENT BETWEEN PROGRAMS—COMBINING WHELAN AND COMPUTER ASSOCIATES

The next step is to make a list of the factors to be used as rules in determining if one computer program infringes another, and more specifically, whether the second hypothetical program in this article infringes the first. This involves combining factors from Whelan and Computer Associates.

While the cases are compatible in most respects, they disagree in one important area. The Whelan court believes that the sequence of the program is copyrightable, whereas the Computer Associates court believes the sequence, or behavior, is a process and not copyrightable.82 Computer Associates pointed out that "sequence" could refer either to the text (the code or instructions) or dynamic behavior of a program (the sequence in which the program appears to be operating according to one watching the computer screen or printouts).83

This Article will assume that both sequences (i.e., the sequence of computer instructions and the sequence in which the program appears to observers to operate) are copyrightable. There are several reasons for this assumption. First, Whelan cites authority that plot sequences in films are copyrightable.84 Secondly, the more conservative legal position is to avoid possible problems and not use the same sequences, if practical not to do so. However, this conservative legal position, if wrong, could lead a second company into doing more work than is necessary in order to avoid possible infringement. Finally, Computer Associates did not show why the court concluded that sequential behavior is a process, thus making it ineligible for copyright protection.85

81 Id.
82 Whelan, 797 F.2d at 1238-39; Computer Assocs., 775 F. Supp. at 559-60, aff'd 982 F.2d at 705-06.
83 Computer Assocs., 775 F. Supp. at 559-60, aff'd 982 F.2d at 705-06.
84 Whelan, 797 F.2d at 1236.
85 See supra note 72 (discussing district court's failure to explain its characterization of sequential behavior as a process).
The following important copyright factors from *Whelan* and *Computer Associates* will be used with the two hypothetical programs in this Article in determining if the second program infringes the first. These factors are:

- Copyright protection extends to both source code and object code.\(^{86}\)
- *Scenes a faire* are not copyrightable.\(^{87}\)
- The structure and sequence of a computer program can be infringed.\(^{88}\)
- Mere ideas are not copyrightable; however, expression of ideas is copyrightable.\(^{89}\)
- File structures are copyrightable.\(^{90}\)
- Copyright infringement may be proved by a showing of access and substantial similarity.\(^{91}\)
- Substantial similarity is analyzed using abstraction, filtration, and comparison.\(^{92}\)

\(^{86}\) *Whelan*, 797 F.2d at 1233; *Computer Assocs.*, 982 F.2d at 702. See *supra* notes 13 and 14 for definitions of "source code" and "object code."

\(^{87}\) *Whelan*, 797 F.2d at 1236.

\(^{88}\) *Id.* at 1236-39; *Computer Assocs.*, 982 F.2d at 703.

\(^{89}\) *Whelan*, 797 F.2d at 1242-43.

\(^{90}\) *Id.* at 1242-43; *Computer Assocs.*, 982 F.2d at 701.

\(^{91}\) *Computer Assocs.*, 982 F.2d at 701.

\(^{92}\) *Id.* at 706-10. See *supra* note 75 and accompanying text (discussing abstraction process).

\(^{93}\) *Computer Assocs.*, 982 F.2d at 707-10. See *supra* notes 79-78 and accompanying text (discussing filtration process).

\(^{94}\) *Computer Assocs.*, 982 F.2d at 710. See *supra* notes 80-81 and accompanying text (discussing comparison process).
III. CREATING A SUBSEQUENT NONINFRINGEMENT PROGRAM THAT ACCOMPLISHES THE SAME TASK AS A PRE-EXISTING PROGRAM

A. THE PROGRAMS

Below are two listings of computer programs, both written in the BASIC\textsuperscript{96} programming language. Both programs accomplish the task of reading in from the keyboard the student identification and grades in Algebra, History, English, and Chemistry for up to 50 students. The programs compute and print the average grade in each course and the average grade for each student.\textsuperscript{97} This Article will use the factors obtained from Whelan\textsuperscript{98} and Computer Associates\textsuperscript{99} to show that the second program accomplishes the same results as the first, but does not infringe the copyright of the first program.

Program #1

20 REM - Program GRADING
40 ON ERROR GOTO 1260 : REM - In case of printer problem
60 DIM NM$(50) : REM - Student Name
80 DIM AL(50) : REM - Algebra grade
100 DIM HI(50) : REM - History grade
120 DIM EN(50) : REM - English grade
140 DIM CH(50) : REM - Chemistry grade
160 CLS : REM - Clear screen
180 S=0 : REM - Number of students

\textsuperscript{96} Beginners All-purpose Symbolic Instruction Code. See KENT PORTER, THE NEW AMERICAN COMPUTER DICTIONARY 32 (Signet 1983). The programs in this Article are written in IBM BASIC 3.0. See generally IBM BASIC REFERENCE, PERSONAL COMPUTER HARDWARE REFERENCE LIBRARY (3d ed. 1984) (discussing BASIC as a computer language); see also IBM BASIC HANDBOOK, GENERAL PROGRAMMING INFORMATION, PERSONAL COMPUTER HARDWARE REFERENCE LIBRARY (3d ed. 1984) (explaining the elements of a BASIC program).

\textsuperscript{97} For brevity purposes, the program is fairly simple. It could, in fact, perform more than it does. For example, the program could check to be sure the student identifications were not over a certain number of letters or numbers long and that each grade is in a specified grade range.


200 PRINT "Enter student identification. If there are no more students, then enter"
220 PRINT "the letter ‘S’ (for STOP)"
240 INPUT "Student I.D. (or ‘S’) = ", NM$(S+1)
260 IF NM$(S+1)="S" OR NM$(S+1)="s" GOTO 440
280 S=S+1 : REM - One more student
300 INPUT "Algebra grade=", AL(S)
320 INPUT "History grade=", HI(S)
340 INPUT "English grade=", EN(S)
360 INPUT "Chemistry grade=", CH(S)
380 PRINT " ": REM - Blank line
400 IF S<50 GOTO 200 : REM - Maximum number of students=50
420 PRINT " ";TAB(81);"Maximum of 50 students reached."
440 IF S>0 GOTO 500 : REM - At least one student
460 PRINT "No students. End program."
480 GOTO 1160 : REM - Back to operating system
500 PRINT " ";TAB(81);"Computations will begin."
740 CO$="Algebra" : REM - For printing ‘Algebra’
760 TT=AA : REM - Sum of Algebra grades
780 GOSUB 1180 : REM - Print average Algebra grade
800 CO$="History" : REM - For printing ‘History’
820 TT=HH : REM - Sum of History grades
840 GOSUB 1180 : REM - Print average of History grades
860 CO$="English" : REM - For printing ‘English’
880 TT=EE : REM - Sum of English grades
900 GOSUB 1180 : REM - Print average of English grades
920 CO$="Chemistry" : REM - For printing ‘Chemistry’
940 TT=CC : REM - Sum of Chemistry grades
960 GOSUB 1180 : REM - Print of average of Chemistry grades
980 PRINT "Overall average grade=":AV/4;TAB(81);" ";
1000 LPRINT "Overall average grade=":AV/4;TAB(81);" ";
1020 PRINT "Student";TAB(22);"Algebra";TAB(32);"History";
TAB(42);"English";TAB(52);"Chemistry";TAB(62);"Average"
1040 LPRINT "Student";TAB(22);"Algebra";TAB(32);"History";
TAB(42);"English";TAB(52);"Chemistry";TAB(62);"Average"
1060 FOR Z=1 TO S : REM - Loop through all students
1080 PRINT NM$(Z);TAB(22);AL(Z);TAB(32);HI(Z);TAB(42);EN(Z);TAB(52);CH(Z);TAB(62);(AL(Z)+HI(Z)+EN(Z)+CH(Z))/4:REM - Print each grade, average grade
1100 LPRINT NM$(Z);TAB(22);AL(Z);TAB(32);HI(Z);TAB(42);EN(Z);TAB(52);CH(Z);TAB(62);(AL(Z)+HI(Z)+EN(Z)+CH(Z))/4:REM - Print, each grade, average grade
1120 NEXT Z:REM - End loop
1140 PRINT "";TAB(81);"Outputting finished. End program."
1160 END:REM - Return to operating system
1180 PRINT "Average grade in ";CO$;"=";TT/S:REM - Course, average grade
1200 LPRINT "Average grade in ";CO$;"=";TT/S:REM - Course, average grade
1220 AV=AV+TT/S:REM - Sum of averages
1240 RETURN:REM - Back to main program from subroutine
1260 IF ERR=24 OR ERR=25 OR ERR=27 GOTO 1320:REM - Printer problem
1280 PRINT "";TAB(81);"Unrecoverable error; type=";ERR;"End program."
1300 ON ERROR GOTO 0:REM - Stop, print error message, then go back to operating system
1320 INPUT "Ready printer, then press ENTER ",YN$
1340 RESUME:REM - Back to place in program using printer

Program #2

10 REM - Program Students
20 ON ERROR GOTO 450
30 NU=50:REM - Maximum number of students
40 DIM GR(4*NU)
50 DIM ID$(NU):REM - For computing averages in each course
60 DIM VR(4):REM - For computing averages in each course
70 CLS:REM - Clears monitor/screen
80 FOR A=1 TO NU
90 IF A=NU THEN PRINT "This is student ";NU;"- which is the maximum number of students allowed."
100 PRINT "Input the student identification. When no more students exist, then input"
110 PRINT "the letter 'N' - meaning NO MORE."
120 INPUT "Input the Student Identification (or 'N') ",&ID$(A)
130 IF NOT(ID$(A)="N" OR ID$(A)="n") GOTO 160
140 NM=A-1
150 GOTO 230
160 INPUT "Grade in Algebra=";GR(A)
170 INPUT "Grade in Chemistry=";GR(A+NU)
180 INPUT "Grade in English=";GR(A+NU*2)
190 INPUT "Grade in History=";GR(A+NU*3)
200 NM=A:REM - Number of students
1996] COPYRIGHT OF A COMPUTER PROGRAM 65

210 PRINT " ": REM - Blank line
220 NEXT A
230 IF NM>0 GOTO 260 : REM - More than zero students
240 PRINT "There are no students.";TAB(81);"This program
will be exited."
250 GOTO 430
260 REM - Compute averages for each student and for each
course
270 FOR V=1 TO 4: REM - For computing averages in each
course
280 VR(V)=0: REM - Initialize each average to zero
290 NEXT V
300 PRINT " ": REM - Blank line
310 PRINT "Student";TAB(20);"Algebra";TAB(31);"Chem.";
TAB(42);"English";TAB(53);"History";TAB(65);"Average";
TAB(81);"Identification";TAB(101);"Grade";TAB(111);
"Grade";TAB(123);"Grade";TAB(134);"Grade";TAB(146);
"Grade"
320 LPRINT "Student";TAB(20);"Algebra";TAB(31);"Chem.";
TAB(42);"English";TAB(53);"History";TAB(65);"Average";
TAB(81);"Identification";TAB(101);"Grade";TAB(111);
"Grade";TAB(123);"Grade";TAB(134);"Grade";TAB(146);
"Grade"
330 FOR A=1 TO NM
340 PRINT ID$(A);TAB(21);GR(A);TAB(31);GR(A+NU);TAB(43);
GR(A+NU*2);TAB(54);GR(A+NU*3);TAB(65);(GR(A)+GR(A+NU)+
GR(A+NU*2)+GR(A+NU*3))/4
350 LPRINT ID$(A);TAB(21);GR(A);TAB(31);GR(A+NU);TAB(43);
GR(A+NU*2);TAB(54);GR(A+NU*3);TAB(65);(GR(A)+GR(A+NU)+
GR(A+NU*2)+GR(A+NU*3))/4
360 VR(1)=VR(1)+GR(A): REM - For computing average Algebra
grade
370 VR(2)=VR(2)+GR(A+NU): REM: - Chemistry
380 VR(3)=VR(3)+GR(A+NU*2): REM - English
390 VR(4)=VR(4)+GR(A+NU*3): REM - History
400 NEXT A
410 PRINT "Average grades";TAB(21);VR(1)/NM;TAB(31);VR(2)/
NM;TAB(43);VR(3)/NM;TAB(54);VR(4)/NM;TAB(65);(VR(1)+VR
(2)+VR(3)+VR(4))/(NM*4)
420 LPRINT "Average grades";TAB(21);VR(1)/NM;TAB(31);VR(2)
/NM;TAB(43);VR(3)/NM;TAB(54);VR(4)/NM;TAB(65);(VR(1)+VR
(2)+VR(3)+VR(4))/(NM*4)
430 PRINT "Program STUDENTS has finished."
440 END: REM - Back to operating system
450 IF ERR=27 OR ERR=25 OR ERR=24 GOTO 510 : REM - Printer
error
460 PRINT "Program STUDENTS has encountered an
unrecoverable error."
470 PRINT "The error type=":ERR
480 PRINT "Return to the Operating System."
490 ON ERROR GOTO 0: REM -Will print error message, then
return control to
500 REM - Operating System
510 INPUT "Printer problems. Ready the printer, then push ENTER.", FAKE$
520 RESUME : REM - Return back to the place in the program using the printer

B. STRATEGY

Of course a programmer leaving one company to work at another should not take listings of the programs to the new company. To do otherwise might violate an agreement with the first company, or it may even constitute theft. However, that programmer might remember what code the programmer had seen or written. So as not to infringe the copyright of the first company, the programmer of the second hypothetical program will accomplish the required result of reading student names and grades, and outputting various grades and averages, but should change as much code and output as possible; however, the programmer need not change code that is required by the purposes to be served by the program, the programming language (BASIC), or common coding methods. A programmer who is familiar with the original program may be able to write a new program more quickly than another programmer. This is true because the programmer having familiarity may be able to recall and use noninfringing ideas.

C. THE CODE

1. Line Numbers. In BASIC, line numbers show the order in which the program is stored in the computer and in which it operates. BASIC requires that line numbers be in the range 0-65529. The first program cannot preempt the use of line numbers since they must be used in order to tell the computer in

100 See Computer Assocs., 982 F.2d at 699-700 (noting that employee who took copies of source code with him to new employer violated his employment contract).
101 For example, it is a crime in New York to take or possess, without proper authority, a listing of a computer program. N.Y. PENAL LAW §§ 155.00 (1), 155.05 (1), 156.00 (2), 156.30 (1) (McKinney 1996).
102 Ideas are not copyrightable, but may be the subject of trade secret protection. See supra note 5.
103 IBM BASIC HANDBOOK, supra note 96, at 3-3.
what order to execute the program. However, while starting with 20 and increasing 20 per line of code may not be copyrightable if it represents a common increment, the second program will begin at line 10 and increase 10 per line of code so as to avoid any questions of infringement. Leaving spaces between line numbers is useful for inserting new code which might become necessary to add features or correct errors.

2. **On Error.** An **ON ERROR GOTO** statement is required in order to have the program respond properly to a computer error condition, so its use cannot be preempted. In both programs, the printer will be used for outputting results; the **ON ERROR GOTO** statement and address to which the programs branch are necessary in order to deal with printer problems. Thus, the use of **ON ERROR GOTO** cannot be preempted. Since an **ON ERROR GOTO** statement needs to be in the program prior to an attempt to use the printer, its use at the beginning of the program does not infringe any sequencing rights of an earlier program. The **ON ERROR GOTO** statement should be near the beginning of the program in order to tell the program how to respond to an error condition, which, of course, can happen early in the running of the program.

---

104 *See supra* note 77 and accompanying text (discussing noncopyrightability of elements dictated by computer specifications). The same reasoning applies to elements dictated by programming language specifications.

105 Elements dictated by common programming practices are likewise noncopyrightable.

106 *Id.*


108 *IBM BASIC REFERENCE, supra* note 96, at 201-02.

109 *See supra* note 77 (discussing noncopyrightability of elements dictated by computer specifications); *see also supra* note 104 and accompanying text (discussing noncopyrightability of elements dictated by programming language specifications).

110 *See Program #1, supra* line 40 (branch to 1260); *Program #2, supra*, line 20 (branch to line 450).

111 *See supra* note 77 and accompanying text (discussing noncopyrightability of elements dictated by computer specifications); *see also supra* note 104 and accompanying text (discussing noncopyrightability of elements dictated by programming language specifications).

112 *See supra* notes 76-77 and accompanying text (discussing noncopyrightability of elements relating to efficiency and elements relating to common programming techniques).

113 *Id.*
3. **REM.** A REM statement inserts explanatory remarks in a program. REM statements are not executed, but are displayed when the program is listed. REM statements are used to explain what the program instructions are accomplishing.\(^\text{114}\) Since REM statements are part of the BASIC language, the use of REM is not copyrightable.\(^\text{115}\) However, the programmer's comments included in a REM statement would likely qualify as part of a literary work. Thus, comments in the second program must be different from those in the first program, unless a particular section of code is necessary in order to accomplish the task and there are only a few reasonable ways of commenting.\(^\text{116}\)

D. **NAME OF PROGRAM**

BASIC requires that a program be given a name in order to store the program in the computer and then retrieve and run the program having that name. The name helps BASIC find the program in order to run it.\(^\text{117}\) The name of the program is on the first line of each program in a REM statement. While the name is not required to appear in the program, it is useful in assisting the programmer in remembering what program to request BASIC to load into the computer and then run.\(^\text{118}\) While the sequence of programming steps may be copyrightable,\(^\text{119}\) placing the program's name at the beginning is probably not copyrightable since it is either merely an idea\(^\text{120}\) or a necessary step in remembering the program name in order to load and run the program.\(^\text{121}\) However, the name of the program might be the expression of the

\(^{114}\) Id. at 291-92.

\(^{115}\) See supra note 77 and accompanying text (discussing noncopyrightability of elements dictated by computer specifications); see also, supra note 104 and accompanying text (discussing noncopyrightability of elements dictated by programming language specifications).

\(^{116}\) See supra notes 57, 76, 89 and accompanying text (discussing idea/expression dichotomy).

\(^{117}\) IBM BASIC REFERENCE, supra note 96, at 167, 306-07, 308-09.

\(^{118}\) Id.


\(^{120}\) Ideas are not copyrightable. 17 U.S.C. § 102(b) (1994).

\(^{121}\) See also supra note 117 and accompanying text (discussing relevance of program name in the BASIC language).
idea/requirement to use a name, therefore, the second program is given a different name from the first.

E. ARRAYS

Data is stored in the computer memory using an array, which is a list of values (e.g., names or test scores) that is referred to by a single name. Declaring the name, type of array, and number of elements is called "defining" or "dimensioning" the array, thus the notation DIM is used when dimensioning arrays.

Each array must be given a name, but the name selected may possibly constitute an expression of the idea to use a name. Therefore, different array names are chosen for Program #2.

The first program has five arrays: one for the student identifications and four for the course grades. While it seems that the use of five arrays might be necessitated by the task, to be certain there is no infringement, only three arrays will be used in the second program. The student identifications will appear in one array, while all grades will appear in a second array; a third

---

122 See supra note 57 and accompanying text (discussing noncopyrightability of mere ideas and copyrightability of expression of ideas).
123 IBM BASIC HANDBOOK, supra note 96, at 3-19.
124 An array can be used to store numbers only, such as test scores, or data containing characters other than numbers, such as a name. Id. at 3-19 to 3-20.
125 An example would be 50 elements because there can be up to 50 students. See supra note 97 and accompanying text (discussing the hypothetical programs in this Article).
126 IBM BASIC HANDBOOK, supra note 96, at 3-19 to 3-20. See also IBM BASIC REFERENCE, supra note 96, at 70.
127 See Program #1, supra, lines 60-140; Program #2, supra, lines 40-60.
128 IBM BASIC HANDBOOK, supra note 96, at 3-19; IBM BASIC REFERENCE, supra note 96, at 70-71.
129 See supra note 57 and accompanying text (discussing noncopyrightability of mere ideas and copyrightability of expression of ideas).
129 Algebra, History, English, Chemistry. See supra note 97 and accompanying text (discussing the hypothetical programs in this Article).
130 See supra note 76 and accompanying text (discussing noncopyrightability of elements necessitated by efficiency).
131 This is done just in case the number of arrays used is considered the expression of the idea to use arrays. See supra note 57 and accompanying text (discussing noncopyrightability of mere ideas and the copyrightability of expression of ideas).
array will be used to compute average grades in each course. The names of the arrays will be different, and rather than having each array be a maximum numeric length, the length will be stored in an item, and that item will be used to set the length of the arrays.

F. CLEAR SCREEN

The screen is cleared early in both programs. The CLS (clear screen) instruction is part of the BASIC language, and thus, its use by Program #2 would not be an infringement. Furthermore, the idea to clear the screen early in a program would seem to be common, and thus not copyrightable.

G. OBTAINING STUDENT IDENTIFICATIONS AND GRADES

The next part of each program obtains the student identifications and grades. Since this is necessary before outputting the student identifications, grades, student averages, and course averages, it must be entered early in the program. Although the sequence in which a program operates is copyrightable, entering the data early in the second program's operation would not seem to be an infringement because such entry is dictated by the specifications for the program. However, the method by which the second program obtains this information is different from the first program. For example, the words printed on the screen when

---

133 Just like the number of arrays used, the name of the array may also be the expression of an idea. Id.
134 That length would be fifty for Program #1. See Program #1, supra, lines 60-140.
135 Again, the number used as the array length might be considered the expression of the idea to use arrays. See Whelan, 797 F.2d at 1238-39 (allowing copyright protection for expression of ideas, not ideas themselves).
136 IBM BASIC REFERENCE, supra note 96, at 36-37.
137 See Computer Assocs., 982 F.2d at 709-10 (holding accepted program practices to be external factors dictating element structure).
138 See Whelan, 797 F.2d at 1247-48 (holding operating sequence to be copyrightable element of program).
139 See Computer Assocs., 982 F.2d at 707-10 (holding unprotectible any elements dictated by necessity, efficiency, or external factors).
asking for the student identification are different.\textsuperscript{140} Also, the first program asks for the grades in this order: Algebra, History, English, Chemistry.\textsuperscript{141} The second program asks for the grades in alphabetical order.\textsuperscript{142}

The programs also use different words and formats on the screen in asking for the grades. For example, Program \#1 says "Algebra grade =," whereas Program \#2 says "Grade in Algebra=." Additionally, the first program sets up its own counter to keep track of the number of students,\textsuperscript{143} whereas the second program uses a FOR loop.\textsuperscript{144}

Also, if the user does not enter grades for any students, each program notifies the user that there are no students. But the words used in the notices are different.\textsuperscript{145} Each program notifies the user if the maximum number of students has been reached, but the messages are different for each program,\textsuperscript{146} and the notifications come at slightly different times. In Program \#1, the user is notified after the fiftieth student has been entered,\textsuperscript{147} whereas the user is notified just before making the fiftieth entry in Program \#2.\textsuperscript{148}

\section*{H. COMPUTING AVERAGE GRADE IN EACH COURSE}

The first program does most of the calculations needed for computing the average grade in each course prior to printing any

\begin{itemize}
\item \textsuperscript{140} Program \#1 will prompt:
  \begin{quote}
  Enter student identification. If there are no more students, then enter the letter ‘S’ (for stop).
  
  Student I.D. (or ‘S’)=
  \end{quote}

\item \textsuperscript{141} See Program \#1, supra, lines 300-360.

\item \textsuperscript{142} See Program \#2, supra, lines 160-190.

\item \textsuperscript{143} See Program \#1, supra, lines 180-460. "s" is used as the counter.

\item \textsuperscript{144} See Program \#2, supra, lines 80-220; see also IBM BASIC REFERENCE, supra note 96, at 106-10.

\item \textsuperscript{147} See Program \#1, supra, lines 440-480. Program \#2 says: "There are no students. This program will be exited." See Program \#2, supra, lines 230-250.

\item \textsuperscript{148} See Program \#1, supra, lines 400-440; Program \#2, supra line 90.

\end{itemize}
information on the printer, and it uses a subroutine for printing information and for computing the overall average grade over all courses. Further, the averages for each course are printed prior to printing any information about each student.

However, in the second program, while information on each student is being output to the printer, averages for each course are being computed. After information on each student is printed, average grades in each course and overall average grades are printed. Program #2 does not use a subroutine to accomplish any of this.

I. SUBROUTINES

Use of subroutines is common in programming, so their use cannot be preempted. However, the sequence of operation within a subroutine might be an expression of the idea being accomplished within the subroutine, and thus copyrightable.

Each program has a subroutine to check for computer errors. Each program attempts to continue running if a printer problem is encountered or to abort if another kind of error is found. The messages given by the programs are different, and the order in which the programs check for a printer error is different.

149 See Program #1, *supra*, lines 500-720.
150 *Id.* at lines 780, 840, 900, 960, 1180-1240.
151 *Id.*
152 See Program #2, *supra*, lines 310-400.
153 *Id.*
154 *Id.* at lines 410-420.
155 *Id.* at lines 270-290, 310-420.
156 See *supra* at note 67 (defining "subroutine").
157 *IBM BASIC REFERENCE, supra* note 96, at 118, 203-04.
158 See *Whelan*, 797 F.2d at 1238-39 (permitting expressions of ideas to be copyrightable).
159 See Program #1, *supra*, lines 40, 1260-1340; Program #2, *supra*, lines 20, 450-520.
160 See Program #1, *supra*, lines 1260, 1320-1340; Program #2, *supra*, lines 450, 510-520.
161 See Program #1, *supra*, lines 1280-1300; Program #2, *supra*, lines 460-500.
162 See Program #1, *supra*, lines 1280, 1320; Program #2, *supra*, lines 450-510.
163 See Program #1, *supra*, line 1260; Program #2, *supra*, line 450. Each program checks for the same printer error conditions (error types 24, 25, and 27) because that is what BASIC requires. *IBM BASIC REFERENCE, supra* note 96, at app. A.
The process of continuing if a printer error is found and aborting otherwise, probably cannot be copyrighted because that is necessary to accomplish the task.\textsuperscript{164}

Program #1 also uses a subroutine to print the average grades in each of the four courses and for computing the overall average grade in all four courses.\textsuperscript{165} Program #2 accomplishes all of this in the main body of the program and does not use a subroutine.\textsuperscript{166}

J. PROGRAM OUTPUTS

Screen and Printer outputs also are protected by copyright, but separate from the program's code.\textsuperscript{167} However, the factors for determining if the output from the second program infringes the output from the first program appear to be the same factors used to determine if the code from Program #2 infringes the code from Program #1.\textsuperscript{168} For example, if there are relatively few ways to print student identifications and grades, then the format chosen may not be copyrightable.\textsuperscript{169} Moreover, printouts which are indispensable are not copyrightable.\textsuperscript{170}

Each program prints the grades for each student. The outputs are as follows:

\textsuperscript{164} Computer Assocs., 982 F.2d at 707-10. Since the only input is from the keyboard, and the only output is on the screen and from the printer, it seems that only printer errors are expected and can be handled.

\textsuperscript{165} See Program #1, supra, lines 1180-1240.

\textsuperscript{166} See Program #2, supra, lines 360-390 and 410-420.

\textsuperscript{167} Computer Assocs., 982 F.2d at 703. Output from computer video games is protected as an audiovisual display under the Copyright Act, 17 U.S.C. § 102(a)(6) (1994). Williams Elecs., Inc. v. Artic Intl, Inc., 685 F.2d 870, 873-74 (3d Cir. 1982). In this case, the games display various objects, such as spaceships and other symbols. Id. at 872. However, the two computer programs used in this Article print words and sentences. Therefore, they may be literary works, protected under 17 U.S.C. § 102(a)(1) (1994), the same as a book is protected. Computer Assocs., 982 F.2d at 706-07.

\textsuperscript{168} Cases relied upon by both Whelan and Computer Associates deal with general copyright principles, not only principles relating to infringement of computer code. Whelan, 797 F.2d at 1231-43; Computer Assocs., 982 F.2d at 695-96, 701-15.

\textsuperscript{169} Computer Assocs., 982 F.2d at 707-08.

\textsuperscript{170} Whelan, 797 F.2d at 1236.
Average grade in Algebra = 84.66666
Average grade in History = 81.5
Average grade in English = 77.5
Average grade in Chemistry = 85
Overall average grade = 82.16666

<table>
<thead>
<tr>
<th>Student</th>
<th>Algebra</th>
<th>History</th>
<th>English</th>
<th>Chemistry</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary Alpha</td>
<td>97</td>
<td>86</td>
<td>62</td>
<td>98</td>
<td>85.75</td>
</tr>
<tr>
<td>Joe Beta</td>
<td>92</td>
<td>87</td>
<td>62</td>
<td>88</td>
<td>82.25</td>
</tr>
<tr>
<td>Catherine Gamma</td>
<td>79</td>
<td>88</td>
<td>95</td>
<td>85</td>
<td>86.75</td>
</tr>
<tr>
<td>Donald Delta</td>
<td>78</td>
<td>64</td>
<td>99</td>
<td>75</td>
<td>79</td>
</tr>
<tr>
<td>Nadine Epsilon</td>
<td>94</td>
<td>69</td>
<td>68</td>
<td>85</td>
<td>79</td>
</tr>
<tr>
<td>Sam Sigma</td>
<td>68</td>
<td>95</td>
<td>79</td>
<td>79</td>
<td>80.25</td>
</tr>
</tbody>
</table>

Output - Program #2

<table>
<thead>
<tr>
<th>Student Identification</th>
<th>Algebra Grade</th>
<th>Chem. Grade</th>
<th>English Grade</th>
<th>History Grade</th>
<th>Average Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary Alpha</td>
<td>97</td>
<td>86</td>
<td>62</td>
<td>98</td>
<td>85.75</td>
</tr>
<tr>
<td>Joe Beta</td>
<td>92</td>
<td>87</td>
<td>62</td>
<td>88</td>
<td>82.25</td>
</tr>
<tr>
<td>Catherine Gamma</td>
<td>79</td>
<td>88</td>
<td>95</td>
<td>85</td>
<td>86.75</td>
</tr>
<tr>
<td>Donald Delta</td>
<td>78</td>
<td>64</td>
<td>99</td>
<td>75</td>
<td>79</td>
</tr>
<tr>
<td>Nadine Epsilon</td>
<td>94</td>
<td>69</td>
<td>68</td>
<td>85</td>
<td>79</td>
</tr>
<tr>
<td>Sam Sigma</td>
<td>68</td>
<td>95</td>
<td>79</td>
<td>79</td>
<td>80.25</td>
</tr>
<tr>
<td>Average grades</td>
<td>84.66666</td>
<td>81.5</td>
<td>77.5</td>
<td>85</td>
<td>82.16666</td>
</tr>
</tbody>
</table>

Both programs use rows and columns to output information.\(^{171}\) It is doubtful that the use of rows and columns to print information can be preempted since it would seem to be a common method of listing information.\(^{172}\) However, the print positions used for printing and the specific column headings might be expressions of ideas, and copyrightable.\(^{173}\) Therefore, the output is slightly different for each program.\(^{174}\) Specifically, the column headings are different. While it may be difficult to tell by looking at sample

\(^{171}\) See Output - Program #1, supra; Output - Program #2, supra.

\(^{172}\) See Computer Assocs., 982 F.2d at 707-10 (rejecting copyright protection for widely accepted programming practices and techniques); see also IBM BASIC REFERENCE, supra note 96, at 340.

\(^{173}\) See Whelan, 797 F.2d at 1238-39 (allowing copyright protection for expressions of ideas, just not ideas themselves).

\(^{174}\) All grades were generated randomly from a range of 60 through 100.
printouts, the various columns start in different print positions. Additionally, the courses are listed in different orders.

K. ENDING RUNNING OF THE PROGRAM

Following the printing of the information, each program finishes running. While sequence of operation is copyrightable, because the purpose of the program is over once the grading information is finished, each has nothing left to do and therefore returns control to the operating system. The programs print slightly different messages when finishing. Printing a message that the program has finished is probably not copyrightable since it would seem to be needed to notify the user of what is happening.

IV. CONCLUSION

This article has been an attempt to show how a second computer program might be written so as not to violate any copyright or trade secrets in a first program. If a computer programmer is able to accomplish this, the programmer might leave the employ of one company and write similar, but noninfringing, software for a second company. This could save time and money for the second company that may otherwise need to put another programmer on the project who would have to spend time learning about the

176 Program #1 uses print positions 1, 22, 32, 42, 52, and 62. See Program #1, supra, lines 1020-1040. Program #2 uses print positions 1, 21, 31, 43, 54, and 65. See Program #2, supra, lines 340-350. See IBM BASIC REFERENCE, supra note 96, at 177-80 and 263-76 (describing how programming commands affect printing).
177 Whelan, 797 F.2d at 1247-48.
178 See Program #1, supra, lines 1140-1160; Program #2, supra, lines 430-440. The operating system is "[t]he master control program that governs the operation of the computer system." COMPUTER DICTIONARY, supra note 96, at 199-200. It is present in the computer or on some storage media while other programs are running. Id.
179 Program #1 prints "Outputting finished. End program." See Program #1, supra, line 1140. Program #2 prints "Program STUDENTS has finished." See Program #2, supra, line 430.
180 Trade secrets are not the focus of this article, therefore they are not discussed in as much depth as copyright. See supra note 16 and accompanying text (discussing rejection of trade secret protection for computer programs by CONTU).
The original programmer might be able to save this time and money using noninfringing ideas learned during the writing of the original software.

---

181 See Computer Assocs., 982 F.2d at 699-700 (providing example of bringing in additional programmers for projects to avoid copyright problems).