ED 388 723 TM 024 188

AUTHOR	Bennett, Randy Elliot; And Others
TITLE	Agreement between Expert System and Human Ratings of Constructed-Responses to Computer Science Problems.
INSTITUTION	Educational Testing Service, Princeton, N.J.
REPORT NO	ETS-RR-88-20
PUB DATE	Aug 88
NOTE	60p.
PUB TYPE	Reports - Evaluative/Feasibility (142)
EDRS PRICE	MF01/PC03 Plus Postage.
DESCRIPTORS	Advanced Placement; *Computer Science; *Constructed Response; *Educational Technology; *Expert Systems; High Schools; *Scoring; Technological Advancement
IDENTIFIERS	Advanced Placement Examinations (CEEB); *MicroPROUST Computer Program

ABSTRACT

This study investigated the extent of agreement between MicroPROUST, a prototype microcomputer-based expert scoring system, and human readers for two Advanced Placement Computer Science free-response items. To assess agreement, a balanced incomplete block design was used with 2 groups of 4 readers grading 43 student solutions to the first problem and 45 solutions to the second. Results showed MicroPROUST to be unable to grade a significant portion of solutions, but to perform impressively on those solutions it could analyze. For one problem, MicroPROUST assigned grades and diagnostic comments similar to those assigned by readers. For the other problem, MicroPROUST's agreement with readers on grades was lower than the agreement of readers among themselves, its grades were higher, and it gave fewer comments, particularly on structure and style. The extent of disagreement on grades, however, was small and much of the disagreement disappeared when papers were rescored discounting style. Three appendixes present programming problems, examples of student solutions and comments, and scoring rubrics. (Contains 10 tables and 13 references.) (SLD)

RR-88-20

RESEARCH REPORT

AGREEMENT BETWEEN EXPERT SYSTEM AND HUMAN RATINGS OF CONSTRUCTED-RESPONSES TO COMPUTER SCIENCE PROBLEMS

Randy Elliot Bennett
Brian Gong
Roger C. Kershaw
Donald A. Rock
Elliot Soloway
Alex Macalalad

Educational Testing Service
Princeton, New Jersey
August 1988

Agreement Between Expert System and Human Ratings of Constructed-Responses to Computer Science Problems

Randy Elliot Bennett

Brian Gong

Roger C. Kershaw

Donald A. Rock

Educational Testing Service

Elliot Soloway

and

Alex Macalalad

Yale University

Acknowledgements

Many people contributed to this study. Mark Stehlik, Terry Gill, Betty Benyon, and eight Advanced Placement Computer Science (APCS) readers carried out the experimental reading. Jeff Wadkins and Jim Braswell advised on organizing the reading as well as on other aspects of the study. Warren Sack and Jim Spoher assisted with the adaptation of MicroPROUST to the APCS problems. Jim Ferris offered advice on research design and Hazel Klein helped sample the student solutions for analysis by MicroPROUST and the APCS readers. Rob Rarich and Erica Hlebowitsh prepared a second, independent sample of student papers and ran MicroPROUST on it. Finally, thanks are due to Dan Eignor and Bill Ward for their helpful comments on an earlier draft of this report, to Carl Haag and Cecil Shaw of the AP program for their encouragement, and to the Program Research Planning Council for its support.

Abstract

If computers can be programmed to score complex constructed response items, substantial savings in selected ETS programs might be realized and the development of mastery assessment systems that incorporate "real-world" tasks might be facilitated. This study investigated the extent of agreement between MicroPROUST, a prototype microcomputer-based expert scoring system, and human readers for two Advanced Placement Computer Science free-response items. To assess agreement, a balanced incomplete block design was used with two groups of four readers grading 43 student solutions to the first problem and 45 solutions to the second. Readers assigned numeric grades and diagnostic comments in separate readings. Results showed MicroPROUST to be unable to grade a significant portion of solutions, but to perform impressively on those solutions it could analyze. For one problem, MicroPROUST assigned grades and diagnostic comments similar to those assigned by readers. For the other problem, MicroPROUST's agreement with readers on grades was lower than the agreement of readers among themselves, its grades were higher, and it gave fewer comments, particularly on structure and style. The extent of disagreement on grades, however, was small and much of the disagreement disappeared when papers were rescored discounting style. MicroPROUST's interchangeability with human readers on one problem suggests that there are conditions under which automated scoring of complex constructed-responses might be implemented by ETS.

Agreement Between Expert System and Human Ratings of Constructed-Responses to Computer Science Problems

Among several recent trends in educational measurement is one toward providing information that more directly benefits individuals as opposed to institutions. One instance of such measurement is diagnostic assessment (e.g., Bejar, 1984; Forehand, 1987), which attempts to offer guidance to teachers and students about the specific problems encountered in learning and how they might be addressed. A second, related instance is constructed-response testing (e.g., Ward, Frederiksen, & Carlson, 1980), in which the test task is closer to that required of students and workers in academic and vocational settings, and therefore of more obvious relevance to the examinee. Both forms of assessment have been practiced for many years; diagnostic assessment has long been a standard, though typically informal, practice of master teachers and constructed response assessment a routine component of classroom tests, graduate school comprehensive exams, and even some standardized testing programs (e.g., the Advanced Placement Program of the College Board).

While these forms of assessment are well established, their use in large-scale testing programs has been hampered by an absence of sound theoretical models upon which to base diagnoses and by practical constraints. With respect to the latter, the primary difficulty has been the subjectivity and high cost associated with scoring: training human readers to achieve acceptable levels of agreement and supporting them while they score thousands of exams could be achieved in only limited ways and by only a very few testing programs.

Recent advances in cognitive psychology, computer science, and educational measurement, however, have opened up new possibilities for making diagnostic, constructed response assessment both scientifically defensible and practical. These advances include work on the nature of expertise (Glaser, 1986), the application of expert and novice cognitive models to diagnostic teaching and measurement (Wenger, 1987), the construction of psychometric models for diagnosis (Tatsuoka, 1985), massive decreases in hardware price/performance ratios, and the development of programming methodologies that permit machines to store, access, and apply knowledge in ways that ever more resemble human cognitive functioning (Barr & Feigenbaum, 1981).

These various advances are reflected in different combinations in a growing crop of educationally oriented computer programs being developed under the rubric of "intelligent tutoring," or more generally, "artificial intelligence" (Wenger, 1987). One prototype, MicroPROUST, was used to diagnostically and numerically rate students' free-response solutions to College Board Advanced Placement (AP) Computer Science problems. This paper describes a study of the level of scoring agreement between MicroPROUST and human readers to determine whether these different graders are interchangeable.

Procedure

Subjects

MicroPROUST. MicroPROUST, a version of PROUST (Johnson & Soloway, 1985), attempts to find nonsyntactic bugs in Pascal programs. MicroPROUST has knowledge to reason about (1) selected programming problems, (2) stereotypical ways in which subcomponents of these problems are solved in Pascal, and (3) common faulty implementations of these subcomponents.

MicroPROUST's knowledge base contains information about two simple Advanced Placement Computer Science problems (see Appendix A). This information is a coded version of the English text of each problem, which MicroPROUST uses to guide its analysis of student solutions. MicroPROUST's knowledge base divides the first problem into eight subcomponents, for which it has a total of 116 unique solutions, and the second into two subcomponents with 33 solutions. Forty-eight faulty subcomponent implementations are also included. MicroPROUST analyzes solutions subcomponent by subcomponent, looking for templates in its knowledge base that match portions of the student's code. This subcomponent-matching strategy gives it considerable leverage; correct and incorrect implementations can be put together in different combinations to handle the variety of responses generated by novice programmers.

MicroPROUST's analysis produces a summative score on the 1-9 scale used by the AP program (where 9 represents a perfect solution), and, where appropriate, a diagnostic comment. Diagnostic comments identify the location of a conceptual error, describe the error's nature, and sometimes guide the studen: toward a reasonable correction (see Appendix B).

For the current study, subcomponent solutions and faulty implementations were based on an analysis of student papers selected from AP program files. Because papers can be requested by the institutions from which students are seeking advanced placement and/or course credit, the solutions left in the files may not be fully representative of those produced by the test taking population. However, the full range of numeric grades assigned by AP readers was represented in the remaining solutions; hence, five papers from each of the nine score levels were drawn randomly to form a sample of 45 solutions for each of the two APCS problems.

After developing the knowledge base, point values were assigned to the faulty implementations to allow the program to produce a numeric score for each solution. Point values were based on the grading criteria used to score the problems operationally in 1984 and 1985.

Next, the solutions used to develop the knowledge base were typed in computer-readable format for analysis by MicroPROUST. As part of the typing process, syntactic errors were corrected, introductory and concluding statements were placed around the solutions associated with one problem to make it a complete program, and other minor changes in form and format were made. Because of their poo conceptual quality, two solutions were not entered for one of the problems, bringing to 88 the total number of solutions submitted to the program.

MicroPROUST runs on an IBM Personal Computer/AT with four megabytes of random access memory, a 10 megabyte hard disk, an enhanced graphics adapter, and the Golden Common Lisp environment.

Readers. To organize and coordinate an experimental reading, two individuals were selected based on their central roles in grading past AP Computer Science examinations. These "chief" readers were asked to suggest a list of the most qualified readers from those participating in the 1986 reading. Eight readers were selected and invited to ETS for a one-day reading. Of the eight, five were male; six were from secondary schools and two from universities; five were from the Northeast, two from the South, and one from the Midwest.

Method

Two sets of research questions were posed. One set focused on summative scoring and the other on diagnostic comments.

Summative scoring.

1. To what degree do human readers of APCS free-response questions agree among themselves in assigning scores?

2. Is MicroPROUST's level of agreement with human readers as high as the level of agreement of human readers among themselves?

Diagnostic comments.

1. To what degree do human readers of APCS free-response questions agree among themselves in making diagnostic judgments?

2. Is MicroPROUST's level of agreement with human readers as high as the level of agreement of human readers among themselves?

3. How do machine and human diagnoses differ?

To gather the data needed to answer these questions, a balanced incomplete block (BIB) design was used (see Table 1). In this design, the eight readers were divided into two equal groups. Without knowledge of the ratings assigned by MicroPROUST, each group read half of the student solutions to each of the two problems, with all four readers in a group reading the same set of solutions. In general, the higher quality solutions were assigned to one set of raters for problem 1 and the other set for problem 2.

Insert Table 1 about here

Two types of readings were conducted: summative and diagnostic. The summative reading was intended to simulate the readings typically performed by the AP program in scoring the Computer Science Examination. Briefly, these readings involve the initial collaborative development by the chief reader and the table leaders of analytical scoring rubrics for each of the Exam's five free-response questions. The rubrics are then introduced to the readers, who, in discussions and practice gradings, refine them. Once operational readings begin, the table leaders re-read samples from each reader, and the chief reads samples from each table leader, to check for and resolve instances of significant scoring discrepancy (2 or more points). This entire process is conducted over a five-to-six day period.

For the experimental reading, the two chief readers refined the rubrics originally developed for the problems through the operational testing program. MicroPROUST's scoring algorithms were then adjusted to conform to these refined rubrics. At the experimental reading, the rubrics (see Appendix C) were discussed with the readers and a dozen or so practice solutions scored (considerably fewer than in the typical operational reading). The chief readers then served as table leaders for each group, responding to questions about the scoring as ones arose. The chiefs did not re-read any papers because re-reading such a small sample as used in this study would artificially increase reader agreement to near-perfect levels.

The diagnostic reading was an attempt to simulate the conceptual comments offered by MicroPROUST (such comments are _not_ usually made as part of APCS readings). Readers were asked to read student papers and (1) identify the location of any error, (2) describe its nature, and (3) offer a suggestion for improvement, where appropriate. Suggestions for improvement were generally not to give replacement code, but rather point the student toward a correction, and were not to exceed two sentences.

As table 1 indicates, the summative and diagnostic readings were interspersed; that is, after each group of readers summatively rated one half of a solution set, they made diagnostic comments on the other half.

Summative analysis procedures. To assess the level of agreement among readers, the product moment correlation between each pair of readers grading the same sample of papers was computed. For each of the four samples (i.e., two problems each split into two sets), six correlations were computed, transformed via the Fischer *r*-to-*z* transformation, and averaged. To take account of situations in which papers are reread, the interrater reliability for papers read by two readers was computed from this average using the Spearman-Brown formula (Stanley, 1971).

To evaluate the level of agreement between MicroPROUST and the readers, the correlation between MicroPROUST and each of the four readers in each sample was computed. These correlations were transformed, averaged, and compared with the average correlation among the readers. The difference between the two averages was tested via a two-tailed t-test of the difference between the mean z-scores, where the standard error of the mean for the readers was,

$$1/[6N-18]^{1/2}$$

and for MicroPROUST,

$$1/[4N-12]^{1/2}$$

as per McNemar (1962), and the standard error of the difference was the square root of the sum of the squared standard errors of the means (because the two average correlations are themselves correlated, this is a conservative test). Next, the reliability for two readers was computed to give an estimate of the interrater reliability to be expected when MicroPROUST is used in concert with a human reader whose level of agreement with other humans is equivalent to that of MicroPROUST's.

A second method of comparing the agreement between MicroPROUST and the readers was also used. This method involved computing the average rating across readers for each paper in a sample, a value conceptually similar to classical test theory's "true" score. These averages were then correlated with MicroPROUST's ratings to provide an overall index of agreement and subtracted from MicroPROUST's ratings to identify how disagreements ' ere distributed.

A third method of examining agreement involved computing the distribution of the disagreements among readers for each solution. These disagreements were found by comparing the ratings of each individual with each of the other three readers, resulting in 12 disagreements for each paper. MicroPROUST was then substituted for each of the four readers in turn and its ratings compared with each of the remaining three, again producing 12 contrasts per paper. By this method, MicroPROUST had as many chances to disagree with the readers as they had to disagree among themselves.

Last, the scale levels of ratings assigned by the program and the readers were compared to detect any tendency of the program to assign higher or lower ratings and to examine whether the readers themselves graded on comparable scales.

To do this, individual mean ratings within each of the four samples were compared by repeated-measures analysis of variance. For the samples rated by the program, F-tests significant at $p < .05$ were followed by post-hoc, Scheffe contrasts (Scheffe, 1953) of the grand mean of the reader ratings with MicroPROUST's mean ratings. Finally, post hoc, paired-sample t-tests among each pair of readers were conducted to identify the sources of score-level differences.

Diagnostic analysis procedures. The diagnostic comments of the readers and MicroPROUST were analyzed in several ways. First, the number of solutions commented on were compared for readers and for MicroPROUST. Next, the correctness of comments was determined by examining the specific code to which each comment referred. This determination resulted in the percentage of correct comments given by MicroPROUST, the percentage of correct comments given by the readers, and the total number of correct comments offered on each solution set by the readers and by MicroPROUST.

Third, comments were classified according to a multi-stage procedure. In stage one, the readers' comments were grouped into distinct comments, comments of the same nature referring to the same piece of program code regardless of the wording used by the reader. For example, comments expressed as "C [the variable name] is uninitialized," "Need to assign C a value before you use it," and "What is the value of C the first time through the loop?" were all coded as dealing with the distinct comment, "Need to initialize variable."

In the second stage, these distinct comments were grouped into general concepts. For example, the distinct comments, "Should localize loop variable" and "Should localize array variable" were organized under the concept "Localize variables."

The last stage involved classifying the concepts into one of five categories dealing with 1) structured programming, 2) functionality, 3) algorithm, 4) implementation, and 5) style, as defined below.

> Structured programming practices. Structured programming comments dealt with use of structured programming practices, particularly functional decomposition, data abstraction, and information hiding. Examples of such comments included "Should break this code into smaller procedures," "Need to pass as parameter rather than access as global variable," and "Localize this variable initialization to the procedure that manipulates the variable."
>
> Functionality. Program functionality comments indicated that the program did not perform all functions defined in the problem specifications. An example was, "Program only reads and prints list of numbers," when the problem was to read the numbers and print them in reverse order.
>
> Algorithm. Algorithm comments pointed out deficiencies in the procedure used to solve a particular problem. Some algorithms could not be implemented in Pascal; others, if implemented, would produce incorrect results. For example, a programmer might write code that rotates all array elements except the last one. A comment might focus on the need to modify the algorithm to take proper account of the last case.
>
> Implementation. Implementation comments dealt with the way the algorithm was carried out. Comments identified mistakes or indicated better/poorer ways of doing some function.
>
> Style. Programming style comments given by readers dealt with making the program easier for a human to understand or use. These comments usually do not affect the output of the program. Examples of comments were "Use meaningful variable names" and "Provide the user with a prompt when he has to provide input from the keyboard."

Once comments were classified, agreement between readers was calculated for each diagnostic concept. Agreement for each concept was defined as the number of readers giving the same comment (regardless of wording) to the same code in the same program. Reader agreement for all the concepts was calculated as the average of the reader agreements on each concept. MicroPROUST's agreement with the readers was indicated by the number of readers giving the same diagnostic comment as MicroPROUST for the same code on the same program.

Since no reader could be expected to make every possible comment with respect to the sample of papers read, an attempt was made to characterize the extent to which each reader made all possible comments. This characterization was defined as the number of correct distinct comments made by a single reader divided by the total number of correct distinct comments made by all readers (including MicroPROUST) for that sample. This value indicated how complete the reader was in making his or her diagnoses relative to the combined diagnoses made by the group. The percentage of correct distinct comments made by MicroPROUST was also calculated and compared with the individual reader values.

To identify areas in which MicroPROUST differed from readers in the comments it made, the number of comments was tabulated for each of the five categories identified above. To correct for the fact that four readers commented on each paper, the average number of comments per reader was compared with the total number of comments made by MicroPROUST for each category. Variation across readers in the types of comments made was considered by looking at the distribution of comments for each individual and by computing the standard deviation across readers of the number of comments made per category.

Results

Summative Scoring

MicroPROUST was able to analyze only 8 of 23 papers in one of the two problem #1 samples and 12 of 22 in one of the two problem #2 samples; in the remaining two samples it analyzed all papers. Mean correlations between the grades awarded by each reader and MicroPROUST's success or failure to analyze papers for the two samples posing difficulty were .57 for problem 1 ($p < .001$, $t = 5.77$, $df = 80$) and .35 for problem 2 ($p < .01$, $t = 3.19$, $df = 76$), indicating that MicroPROUST tended to be more successful with well-developed problem solutions. Because solutions generally were assigned on the basis of quaility (better solutions to one set of raters for problem 1 and to the other set for problem 2), MicroPROUST's tendency to be more successful with well-developed solutions helps explain why analysis failures were concentrated in two of the four samples.

Table 2 presents results of the summative analysis. As the table indicates, the mean correlations for the readers were .94 and .77 on problem 1, and .83 and .93 on problem 2. Within each problem, the mean correlations were significantly different across reader groups, with the lower values associated with the samples on which MicroPROUST had difficulty.

Insert Table 2 about here

Mean correlations of MicroPROUST with the readers are only shown for the two samples in which MicroPROUST was able to grade all papers. These correlations are .74 for problem 1, significantly lower than the value for the readers alone, and .95 for problem 2, marginally higher than the reader value.

Table 2 also presents reliabilities for two readings. These estimates indicate the level of consistency that might be expected between the average ratings of pairs of readers. If MicroPROUST was to be paired with a human reader whose agreement with other humans was similar to MicroPROUST's, the reliability for problem 1 would be .85, a respectable value. The reliability for two human readers, however, is considerably higher at .97. For problem 2, the reliability for two human readers is equivalent to that for a human paired with MicroPROUST.

The correlation between MicroPROUST's ratings and the average rating for each problem taken across all four raters was also computed. These average ratings can be taken to represent an estimate of classical test theory's "true" score, the examinee's actual standing on the attribute being measured. For problem 1, the correlation with the average rating was .75 whereas for problem 2 it was .96.

In Table 3, the distribution of absolute differences between MicroPROUST's ratings and the mean of the reader ratings is presented. For the problem 1 sample rated by MicroPROUST, half of the papers had discrepancies of one or two points. For the problem 2 sample, only one paper had such a difference.

Insert Table 3 about here

In table 4, the distribution of absolute differences is presented between pairs of readers and between MicroPROUST-reader pairs. As the table indicates, for the problem 1 sample rated by both MicroPROUST and the readers the largest discrepancies are on the order of two points and belong to MicroPROUST. The readers never disagree among themselves by more than one point. For the problem 2 sample rated by both machine and humans, the distributions are very similar to one another, consistent with the correlational results. Finally, the disagreements among readers are far higher for the problem sets that MicroPROUST did not rate than for those that it did. The magnitude of disagreements among readers 5-8 is about twice as great for the former than for the latter problem set; for readers 1-4, it is almost six times larger.

Insert Table 4 about here

Table 5 shows the mean ratings given by each reader. For problem 1, significant differences among the five individual mean ratings (four readers and MicroPROUST) were detected (p < .001, F = 9.84, df = 4,84). Post hoc contrasts showed MicroPROUST's mean rating to be significantly higher than the mean for all readers (p < .001, t = -6.27, df = 84) and to be higher than each individual reader mean (t range = 3.20 to 4.54, p range < .001 to .005, df = 21). No human reader mean, however, was different from any other human reader mean. For the same group of readers (minus MicroPROUST) on problem 2, however, differences in score level did exist (p < .01, F = 5.22, df = 3,63). Paired-samples t-tests showed reader 3 to differ from readers 1 and 4 and reader 2 to differ with reader 4 (t range = 2.59 to 3.14, p range < .005 to .02, df = 21).

Insert Table 5 about here

For readers 5-8, no differences among the individual mean ratings were found on the problem set graded with MicroPROUST (p = .56, F = .765, df = 4,80) or the set graded by the readers only (p = .28, F = 1.30, df = 3,66).

Insert Table 6 about here

Diagnostic Comments

As noted earlier, MicroPROUST was able to grade only 8 of 23 papers in one of the two problem 1 samples and 12 of 22 in one of the two problem 2 samples. In the two remaining samples, it graded every paper. Even though it was able to grade every paper, it did not comment on every paper it graded. For the samples in which it graded all papers, it commented on 7 of 22 solutions to problem 1 and 18 of 21 for problem 2, the difference between problems being due to the proportion of papers to which MicroPROUST awarded perfect grades (68% for problem 1 and 14% for problem 2; on these perfect papers, it made no commentary). In contrast, every human reader commented on each solution, except one reader who made no comment on two papers. On perfect solutions (i.e., those graded 9 by the other group of readers), readers made mostly stylistic comments not included as errors in the scoring rubric.

Over all solutions analyzed by MicroPROUST, approximately 90% of the program's comments were correct. Taken across problems and groups, the human readers averaged 98% of their comments correct; the worst human error rate was 97% correct. The errors committed by the humans and MicroPROUST did not overlap, nor were there patterns among readers or between problems.

Table 7 presents the number of correct comments given each solution set by the readers and by MicroPROUST. Several points should be noted. First, within some groups readers varied widely in the number of comments they assigned. For example, on problem 2 reader 1 made almost twice as many comments as reader 4, while on problem 1 reader 1 made nearly twice as many as reader 3. Second, the number of comments varied considerably by group and problem with many more comments assigned to the less adequate solutions (i.e., the ones not analyzed by MicroPROUST). Finally, while readers gave many more comments than MicroPROUST overall, this relationship varied by problem. For problem 1, readers 5-8 gave on average four times as many comments, while for problem 2 the numbers of comments awarded by the program and the readers were comparable with the program equalling or exceeding two of the four readers.

Insert Table 7 about here

When agreement between readers is expressed as the number of readers giving the same comment to the same code in the same program, agreement was moderate, with an average 2.4 readers of a possible four giving each comment (calculated across both reader groups and problems). The readers had high agreement with MicroPROUST on most of the errors it diagnosed. An average 3.2 human readers (out of a possible four) noted each of the 64 errors correctly diagnosed by the program (calculated across all problems analyzed by the program).

Table 8 gives the percentage of total correct distinct comments generated by the readers that were identified by individuals. As the table indicates, readers identified, on average, between 59% and 68% of the total number of diagnostic comments identified by their subgroup, indicating a substantial amount of variability in the comments made across individuals. On problem 1, MicroPROUST identified a far smaller proportion of the total (11%) than the readers did. For problem 2, however, its performance, at 52%, was comparable to the readers, at least in terms of the proportion of the domain covered.

Insert Table 8 about here

How exactly did MicroPROUST differ from the readers in the kinds of comments it made? Table 9 shows the number of correct comments made by category for readers and for MicroPROUST. To correct for the fact that four readers (but only one MicroPROUST) commented on each paper, the average number of comments per reader is presented against the total number of comments for MicroPROUST. As indicated in the table, the distribution of reader comments varied across problems and reader groups. Across problems and groups, readers made multiple comments in each of the five categories. MicroPROUST, in contrast, concentrated its comments on implementation. When compared to readers within problems, the biggest differences between MicroPROUST and the readers were for problem 1, a finding consistent with the correlational analysis. On this problem, MicroPROUST performed comparably to readers on function, algorithm, and implementation, but made no comments on structure and style (readers made 11 and 25, respectively). For problem 2, the numbers of comments were distributed similarly for MicroPROUST and the readers with the possible exception of structure, which readers made more mention of. Finally, it is well to note that reader comments were extensive on the problem sets that MicroPROUST could not analyze completely.

Insert Table 9 about here

Within groups, there were some striking differences in the categories of errors individual readers chose to comment on (see Table 10). For instance, although they read the same programs, reader 1 made style comments over four times as often as reader 3. Similarly, readers 5 and 7 commented on program functionality much more frequently than did readers 6 and 8. Overall, the greatest variability across readers was found for style comments and the least for algorithm comments.

Insert Table 10 about here

Discussion

This study examined the extent of scoring agreement between a computer program and expert human readers for two constructed response problems taken from the College Board's Advanced Placement Computer Science examination. The results suggest several conclusions. First, MicroPROUST appears unable to grade a significant portion of solutions to a given problem (28% in the current study and 58% in an independent sample). This comes as no surprise, for the program is a prototype running on a personal computer (albeit, a substantially expanded one); the fact that it can analyze <u>any</u> substantial portion of solutions is an accomplishment.

MicroPROUST is unable to analyze all solutions because the great variety of correct and incorrect ways of formulating problem solutions requires more knowledge than the program currently has. While its knowledge base can be increased, it is questionable whether the program will ever be able to evaluate the same wide range of responses that humans can. One solution to this shortcoming may be to place constraints on the types of responses students can make. For example, instead of giving students a specification and asking them to write a program to implement it, one might present a faulty program and ask them to write a correct version of it (Braun, 1988). These corrected versions would then be given to MicroPROUST for analysis.

While it is clear that MicroPROUST cannot analyze some significant portion of papers, it is not completely clear whether this failure is related to the quality of the solution. In the current sample, there was a clear tendency for the program to analyze well-formed solutions more frequently than poorly-formed ones. However, in a second independent sample for problem 2, analysis failures were <u>not</u>

associated with reader score level ($\underline{r}$ = .16, $\underline{p}$ > .1, $\underline{t}$ = 1.04, $\underline{df}$ = 43,). These contradictory findings may be in part a result of the development process: MicroPROUST's knowledge base was built from the same set of solutions that was used in the study. Hence, an effort was made to include in the knowledge base as many correct program plan subcomponents from the development sample as possible. This effort might be expected to result in the program's being able to analyze a high proportion of well-formed responses. Because poorly-formed responses contain many types, levels, and combinations of bugs, it is more difficult to represent them in a knowledge base. Therefore, a smaller proportion of the total number of bugs in the development sample was included in the knowledge base. As a result, for the development sample MicroPROUST was able to handle more well- than ill-formed responses.

Students' development strategies, however, might differ somewhat from sample to sample (there are many ways to correctly solve programming problems and a development sample of 45 solutions probably provides only a small number of these). To the extent this occurs, MicroPROUST might have more trouble analyzing well-formed responses in new samples. In an independent sample, therefore, one might expect MicroPROUST's ability to analyze well-formed solutions to degrade to a level more comparable to its ability to analyze poorly-formed ones. Should MicroPROUST's knowledge base be extended, encompassing more and more of the universe of common well-formed solutions, a generalized bias toward analyzing well-formed responses might result.

It is interesting to note that, like MicroPROUST, readers also had more difficulty grading poorer solutions. For both problems, their agreement levels were significantly lower for these solutions than for the more well-formed ones. In contrast to MicroPROUST, readers assign a grade, albeit a less reliable one than they otherwise generate.

Whereas MicroPROUST could not analyze a significant proportion of responses, its performance on the subset it could analyze was impressive, though not always as good as the readers. For one of the two problems studied, MicroPROUST assigned grades and diagnostic comments that were very similar to those assigned by readers. For the other problem, several important differences were evident: MicroPROUST's level of agreement with the readers for summative scores was lower than the level of agreement for readers among themselves; its grades were, on average, higher; and it gave fewer comments, particularly in matters related to structure and style. Finally, though MicroPROUST was incomplete compared to readers in the diagnostic comments it assigned, readers agreed more with MicroPROUST on its comments than they did on their own remarks.

Though MicroPROUST disagreed with readers on one problem in assigning summative scores, the magnitude of disagreement was, by most measures, relatively small. For example, the average correlation between MicroPROUST and the readers was in the .70s, which while smaller than the agreement level among readers, indicates a substantial degree of association all the same. Second, whereas MicroPROUST's ratings were higher, they were so by only six tenths of a point on a nine-point scale. Finally, the program's discrepancy with individual readers reached the level required for resolution in operational readings (2 points) for only 10% of the possible disagreements for the problem.

Disagreement appeared to be more substantial for this problem on the diagnostic analysis, particularly the proportion of the total universe of potential correct remarks covered. While readers were far from perfect, they covered a much greater segment of this universe than the program, indicating that their diagnostic commentary is generally more complete. This indicates that even on those solutions that MicroPROUST can analyze it is apt to miss student errors.

Many of the errors MicroPROUST failed to comment on were related to programming style. Style omissions also would seem to play a considerable role in score disagreements between the program and readers: when solutions are rescored to remove the effect of style, MicroPROUST's agreement with readers increases substantially (for problem 1, the average correlation between MicroPROUST and the readers increases from .74 to .86 and level differences are eliminated ($\underline{p}$ = .636, $\underline{F}$ = .639, $\underline{df}$ = 4,84). Work on incorporating style knowledge in MicroPROUST might, therefore, remove a major source of disagreement with human readers.

Whether solutions are graded by readers or MicroPROUST, the problem and/or adequacy of the solution seems to affect reliability. For example, the same group of readers was able to grade on a common scale for one problem but not the other. Because the adequacy of the solutions across these two problems differed considerably, further investigation of the stability of rater reliability across problems and solution quality seems warranted.

What limitations might have affected this study's outcome? One important limitation was the use of the same sample of solutions for both knowledge base development and evaluation of interrater agreement. To get a rough indication of whether MicroPROUST's performance would generalize to other solution samples, the program was given a set of 45 solutions to problem 2 randomly drawn from AP files. The correlation between MicroPROUST's ratings and the original reader-assigned grades for the 19 solutions the program was able to analyze was .82 ($\underline{p} < .001$, $\underline{t} = 5.8$, $\underline{df} = 17$), a respectable performance. All the same, a more complete cross-validation might provide stronger evidence of the program's power.

A second set of limitations that might have affected the study's outcome is that the experimental reading differed from operational ones in several ways. On the negative side, less time was spent going over and practicing application of the scoring rubric; discussing problems with one's neighbors was discouraged; within groups, papers were more restricted in quality, providing less variety; and the indentations made by students in their solutions were changed inadvertently by the printing program, making the papers somewhat harder to read. On the positive side, the problems presented were the easiest on the Advanced Placement Computer Science Examination, the reading lasted one day instead of six, and all the readers had participated in previous operational readings. Because no other studies of rater reliability for the APCS exam exist, it is impossible to estimate confidently the effect of these differences. The reliabilities observed here, however, are not dramatically different from those found for analytically-scored AP exams in other subjects (e.g., physics, mathematics). Consequently, there is little reason to suspect that these differences had any material overall effect.

What does this research suggest for practical uses of MicroPROUST-like systems in scoring constructed response items? First, the data suggest that, given certain constraints, such systems can do as well as readers. This ability is limited to a portion of the solutions the system encounters, perhaps to the quality of the solutions and/or a particular class of problem, and, with respect to diagnostic commentary, to domains other than style. These limitations make clear that such systems must be used in conjunction with people; depending upon the setting, those individuals might be classroom teachers or Advanced Placement readers. For example, in an instructionally-based diagnostic system, if the computer cannot analyze a solution it might present a series of multiple choice or more constrained free-response items.

The student's performance on these items might be used either to compute an estimate of the student's ability to solve programming problems or to give it the information needed to successfully analyze the student's original production. Should the computer still be unable to analyze the original solution, that response could be routed to the teacher. Research will need to be conducted on whether and how information gathered from multiple choice and more constrained free-response item types might facilitate analysis of open-ended items.

In the Advanced Placement setting, a MicroPROUST-like system might also work in conjunction with people. After submitting a validation sample to the program to insure that it graded on a common scale with the readers, the program could be used as a "first-pass" reader. Solutions the program was unable to grade would be routed to people. Alternatively, the program could be used to help readers stay on scale. Papers could be pre-read by MicroPROUST between their arrival at ETS and the time they are submitted for grading; or, already-graded papers could be re-read by the program during the operational reading. Discrepancies would be resolved by the table leaders and/or chief reader.

Using MicroPROUST-like systems in the classroom presents only temporary implementation problems. Though it is true that such systems require computing resources beyond those currently found in schools, the price of these resources is rapidly decreasing. Consequently, advanced computer systems should eventually be common enough to permit the use of expert assessment systems in educational settings.

Significant implementation problems will, however, need to be addressed before such systems can be used in the Advanced Placement environment. In this setting, the major issues relate to getting solutions into machine-readable form. Students currently submit handwritten solutions in examination booklets. Having these solutions transcribed at ETS might introduce errors in punctuation, spacing, or other subtleties that could affect test score. Therefore, it would seem wise to determine the frequency and effect of errors introduced in transcription if this method is to be considered seriously.

Requiring students to submit their solutions in machine readable format is an obvious alternative to transcription. However, this alternative also presents problems. Should solutions be entered using only an editor, or are interpreters and compilers permissible? If interpreters and compilers are allowed, those using these tools would have the benefit of programs that locate certain types of programming errors. If all students are advised to use an interpreter or compiler, other inequities are introduced. For example, interpreters permit programs to be developed faster than compilers giving students more time to produce their solutions. In addition, some tools are more capable than others, providing more help in error prevention and location. How serious these potential inequities are is not clear. Experimental studies should be conducted to elucidate the effects of these different data entry options.

Given that the conversion problem can be solved, significant savings in grading free-response items might be realized if the student solutions that MicroPROUST will not analyze can be predicted. Performance on the multiple-choice section, for example, might predict the program's success in grading free responses. If so, only selected papers would be transcribed; others would be routed to readers as is. Assuming that MicroPROUST could analyze 50% of student responses, only half the current number of graders might be needed (or the same number of graders for half the time).

What general conclusion can be drawn from MicroPROUST's performance? MicroPROUST is an "existence proof" of the fact that a machine is, within certain important limitations, interchangeable with humans in grading complex constructed responses. Our next task is to find practical ways to exploit this capability to improve the grading of AP examinations and, further, to create powerful, individualized learning and assessment environments that advance the way Advanced Placement content is taught and knowledge of it measured.

References

Barr, A., & Feigenbaum, E. A. (1982). The handbook of artificial intelligence (Volume 1). Stanford, CA: HeurisTech Press.

Bejar, I. I. (1984). Educational diagnostic assessment. Journal of Educational Measurement, 21, 175-189.

Braun, H. I. (1988, February 4). Personal communication.

Forehand, G. A. (1987). Development of a computerized diagnostic testing program. Collegiate Microcomputer, 5, 55-59.

Glaser, R. (1986). The integration of instruction and testing. In The redesign of testing for the 21st century (Proceedings of the 1985 ETS Invitational Conference). Princeton, NJ: Educational Testing Service.

Johnson, W. L., & Soloway, E. (1985). PROUST: An automatic debugger for Pascal programs. Byte, 10(4), 179-190.

McArthur, D., Stasz, C., & Hotta, J. Y. (1986-87). Learning problem-solving skills in algebra. Journal of Educational Technology Systems, 15, 303-324.

McNemar, Q. (1962). Psychological statistics. New York: Wiley.

Scheffe, H. (1953). A method for judging all contrasts in the analysis of variance. Biometrika, 40, 87-104.

Stanley, J. (1971). Reliability. In R. L. Thorndike (Ed.), Educational measurement. Washington, D.C.: Ame ican Council on Education.

Tatsuoka, K. K. (1985). Diagnosing cognitive errors: Statistical pattern classification and recognition approach (Research Report 85-1-ONR). Urbana-Champaign: University of Illinois.

Ward, W. C., Frederiksen, N., & Carlson, S. B. (1980). Construct validity of free-response and machine scorable forms of a test. Journal of Educational Measurement, 17, 11-29.

Wenger, E. (1987). Artificial intelligence and tutoring systems: Computational and cognitive approaches to the communication of knowledge. Los Altos, CA: Morgan Kaufmann.

Table 1

Study Design

Type of Reading	Student Paper	Readers 1	2	3	4	5	6	7	8
Numeric	1-1	x	x	x	x				
	.	.	.	.	.				
	1-22	x	x	x	x				
	2-1					x	x	x	x
	.					.	.	.	.
	2-21					x	x	x	x
Diagnostic	1-23	x	x	x	x				
	.	.	.	.	.				
	1-45	x	x	x	x				
	2-22					x	x	x	x
	.					.	.	.	.
	2-43					x	x	x	x
Numeric	1-23					x	x	x	x
	.					.	.	.	.
	1-45					x	x	x	x
	2-22	x	x	x	x				
	.	.	.	.	.				
	2-43	x	x	x	x				
Diagnostic	1-1					x	x	x	x
	.					.	.	.	.
	1-22					x	x	x	x
	2-1	x	x	x	x				
	.	.	.	.	.				
	2-21	x	x	x	x				

Table 2

Comparison of Reader vs. MicroPROUST Ratings

	Problem 1			
	Readers 1-4		Readers 5-8	
Index	Readers	Micro-PROUST	Readers	Micro-PROUST
Number of papers	22	22	23	--
Mean correlation	.94	.74*	.77**	--
Reliability for two readers	.97	.85	.87	--
	Problem 2			
Number of papers	22	--	21	21
Mean correlation	.83***	--	.93	.95
Reliability for two readers	.91	--	.97	.97

Note. For readers, the mean correlation is the average of the transformed correlations between each pair of readers. For MicroPROUST, the mean is the average of the transformed correlations of MicroPROUST with each reader.

*p < .001, t (two-tailed) = 7.19, 190 df; computed against the problem 1 mean for readers 1-4.

** p < .001, t (two-tailed) = 5.61, 234 df; computed against the problem 1 mean for readers 1-4.

*** p < .001, t (two-tailed) = 3.59, 222 df; computed against the problem 2 mean for readers 5-8.

Table 3

Distribution of Absolute Differences Between MicroPROUST's Ratings and the Mean of the Reader Ratings

	Frequency	
Absolute Discrepancy	Problem 1 Readers 1-4	Problem 2 Readers 5-8
2	2	1
1 - 1.99	9	0
0 - .99	11	20

Table 4

Distribution of Largest Differences Between Twelve Pairs of Readers and Twelve MicroPROUST-Reader Pairs

	Problem 1			
	Frequency of Absolute Discrepancy			
Absolute Discrepancy	Among Readers 1-4	Between Readers 1-4 and Micro-PROUST	Among Readers 5-8	Between Readers 5-8 and Micro-PROUST
4			4	-
3			10	-
2		26	68	-
1	38	141	112	-
0	226	97	82	-
Mean	.14	.73	1.07	
	Problem 2			
6	4	-		
5	4	-		
4	4	-		
3	10	-	2	
2	32	--	24	27
1	66	-	68	54
0	144	-	158	171
Mean	.83		.48	.43

Table 5

Mean Ratings Given by MicroPROUST and Human Readers

			Readers 1-4			
Problem	Micro-PROUST	All Readers	Reader 1	Reader 2	Reader 3	Reader 4
1[a,b]	8.46	7.85	7.82	7.86	7.82	7.91
2[c]	----	2.73	3.32	2.5	2.86	2.23

[a] $p < .001$, $F = 9.84$, $df = 4,84$ for differences among readers including MicroPROUST.

[b] $p < .001$, $t = -6.27$, $df = 84$ for the difference between MicroPROUST and the mean for all readers.

[c] $p < .01$, $F = 5.22$, $df = 3,63$ for differences among readers including MicroPROUST.

			Readers 5-8			
Problem	Micro-PROUST	All Readers	Reader 5	Reader 6	Reader 7	Reader 8
1	----	4.62	4.96	4.61	4.39	4.52
2	6.43	6.53	6.48	6.57	6.38	6.67

Table 6

Univariate Repeated Measures F-Test for Table 5,

Mean Ratings Given by MicroPROUST and by Readers

Problem 1: Readers 1-4					
Source	S.S.	df	M.S.	F	P
Readers	6.51	4	1.63	9.84	.000
Error	13.89	84	0.17		
Total	20.40	88			

Problem 2: Readers 1-4					
Source	S.S	df	M.S.	F	P
Readers	14.73	3	4.91	5.22	.003
Error	59.27	63	0.94		
Total	74.00	66			

Table 7

Number of Correct Comments Given by Readers and by MicroPROUST

Readers 1-4						
Problem	Micro-PROUST	Reader Mean	Reader 1	Reader 2	Reader 3	Reader 4
1	----	73	93	78	54	67
2	22	25	32	27	22	18

Readers 5-8						
Problem	Micro-PROUST	Reader Mean	Reader 5	Reader 6	Reader 7	Reader 8
1	10	42	34	43	46	45
2	-----	67	69	69	65	65

Table 8

Percentage of Total Correct Distinct Comments Generated by Group that were Also Identified by Individual Readers

Problem	Micro-PROUST	Reader Mean	Reader 1	Reader 2	Reader 3	Reader 4
		Readers 1-4				
1	----	59	76	63	44	54
2	52	59	76	64	52	43

Problem	Micro-PROUST	Reader Mean	Reader 5	Reader 6	Reader 7	Reader 8
		Readers 5-8				
1	11	46	37	47	51	49
2	----	68	70	70	66	66

Table 9

Number of Correct Comments by Concept Category for Readers and MicroPROUST

Concept Category	Readers 1-4 Mean Comments	Micro-PROUST Total Comments	Readers 5-8 Mean Comments	Micro-PROUST Total Comments
Problem 1				
Structure	13	--	11	0
Function	15	--	1	3
Algorithm	7	--	0	0
Implementation	23	--	5	7
Style	15	--	25	0
Problem 2				
Structure	4	0	13	--
Function	0	0	3	--
Algorithm	9	9	23	--
Implementation	11	13	26	--
Style	1	0	3	--

Table 10

Number of Correct Comments Made by Individual Readers within Concept Categories

Concept	Reader								
Category	1	2	3	4	5	6	7	8	Mean(SD)
Structure	22	15	12	18	24	30	29	13	20(6)
Function	17	16	17	10	5	1	8	2	10(6)
Algorithm	18	16	14	17	23	21	20	28	20(4)
Implementation	47	39	28	24	30	33	31	28	33(6)
Style	21	19	5	16	21	27	23	40	21(8)

Appendix A: Programming Problems

Problem 1: Write a program to read eight integers from the terminal, display them in reverse order, and display the number of negative integers read. For example, given the input:

```
-4   3   -2   1   -18   -20   5   -7
```

the program should produce the output:

```
-7   5   -20  -18   1   -2   3   -4

5 of the integers are negative.
```

Problem 2: Write a procedure that rotates the elements of an array *s* with *n* elements so that when the rotation is completed, the old value of *s*[1] will be in *s*[2], the old value of *s*[2] will be in *s*[3],..., the old value of *s*[*n*-1] will be in *s*[*n*], and the old value of *s*[*n*] will be in *s*[1]. Your procedure should have *s* and *n* as parameters. You may assume that the type *Item* has been declared and *s* is of type *List* which has been declared as *List* = array[1...*Max*] of *Item*.

Appendix B: Examples of Student Solutions and Diagnostic Comments for Problem 1

Note: In MicroPROUST, the student solution is presented on the cathode ray tube. Diagnostic comments are displayed at the bottom of the solution.

```
Program Find(input,output);

    Type
        ListType = Array[1..8] of Integer;

    Var
        List: ListType (* array of eight integers *);
        Found: Integer (* holds number of negative numbers found *);

procedure RData(List1: ListType);   YOUR ARRAY SHOULD HAVE BEEN PASSED
                                    AS A VARIABLE PARAMETER RATHER THAN A
                                    VALUE PARAMETER.
    Var
        I: integer (* counter used to move through List1 array *);

    Begin
        for I := 1 to 8 do
            Begin
                Write('Enter number ',I,': ');
                Readln(List1[I]);
            End (* end for when I := 8 *);
    End (* end procedure RData->ReadData *);

procedure DWrite(List1: ListType);

    Var
        I: integer (* counter used to move through List1 array *);

    Begin
        for I := 8 downto 1 do
            Write(List1[I]:5);
        Writeln;
    End (* end procedure DWrite->DataDisplay *);

procedure Scand(List1: ListType);

    var
        I: integer (* index to array List1 *);

    begin
        for I := 1 to 8 do
            if List1[I] < 0 then Found := Found + 1;
    end (* end Scand->ScanList *)


begin (* main program *)
    RData(List) (* go input a list of 8 numbers *);
    DWrite(List) (* print list in reverse order *);
    Found := 0;
    Scand(List) (* find all negative integers in list *);
    Writeln (Found, ' of the integers are negative.');
end.
```

```
program ReadIntegers(input,output);

type
   Num = array[1..8] of integer;

var
   Number: Num;
   I, Total: integer;

procedure Enter;

   begin
      I := 0; Total := 0;
      while I < 8 do
         begin
            Write('Enter a Integer ==>');
            readln(Number[I]);
            I := I + 1;
            if I < 0 then Total := Total + 1;
            writeln
         end
   end;

procedure PrintOut;

   begin
      writeln('Output ==>');
      i := 8;
      while I > 0 do
         begin
            write(Number[I],'  ');
            i := i - 1
         end;
      writeln;
      writeln(Total,' of the integers are negative.');
      writeln
   end;

begin
   Enter;
   Printout
end.
```

YOU HAVE USED THE WRONG INITIAL VALUE. IT'LL NOT BE WITHIN THE BOUNDS OF YOUR ARRAY.

YOU HAVE USED THE WRONG BOUNDARY VALUE. THE WHILE LOOP WILL EXIT BEFORE YOU HAVE READ IN THE ~~LAST~~ NUMBER.

IT APPEARS YOU HAVE USED THE WRONG VARIABLE TO TEST FOR A NEGATIVE VALUE.

Appendix C: Scoring Rubrics

Grading Rubric - Question #1 1984

Program/Procedure Headers Present and Correct	+1	
Type/Var Declarations Present and Correct	+1	
	max	-1
Negative Counter Initialized Correctly	+1	
Input Loop Present and Correct	+1	
	max	-2
Reads and Stores 8 Integers Correctly (read statement present & correct) (also includes calling a read module if implemented)	+2	
Negative Test Present and Correct (valid if reading of integers present)	+1	
Negative Count Incremented Correctly (valid if reading of integers present)	+1	
	max	-3
Reverse Loop Direction Correct (and consistent with the other loop)	+1	
Displays Negative Count Correctly (including a descriptive string in the output)	+1	
Displays All Integers Present	+1	
	max	-2
Style and Syntax	max	-1

user prompt missing	- 1
confusing indentation(formatting)	- 1
redefining standard procedures or using reserved words incorrectly	- 1
improper data structure (8 variables)	- 1
unnecessary or useless code	- 1 / 2
3 or more syntax errors	- 1 / 2
[] and () confusion	- 1 / 2

1984 QUESTION #1	Program/Procedure Headers	Type / Var Declarations	Negative Counter Initialized	Input Loop Present	Reads 8 Integers	Negative Test Present/Correct	Negative Count Incremented	Reverse Loop Direction Correct	Displays Negative Count	Displays All Integers Present	Syntax & Style	Total Points
Reader #												
☒	Max -1		Max -2		Max -3			Max -2			Max -1	☒
Booklet #	+1	+1	+1	+1	+2	+1	+1	+1	+1	+1	+1	+9

Grading Rubric - Question #1 1985

Procedure Header Error	- 1	
Missing VAR	- 1	
Missing Parameter	max	-1

Interpretations:

extra VAR or unnecessary parameter in formal parameter list -1/2 style

Declaration and Statement Errors	- 1	
Type Error	- 1	
Missing Declaration	- 1	
Incorrect or Missing Loop Initialization	- 1	
Incorrect or Missing Loop Conclusion	max	-2

Interpretations:

initialization/conclusion:

if consistent w/problem
temp:=s[N] --OK

if consistent w/ reversed rotation solution
temp:=s[1] --OK

points are lost for inconsistency or incorrect statements

Loop Logic Errors

Minor Loop Error
(using max, index out-of-bounds) -2

Loop Direction Error
(single array using 'TO') - 2

Loop Inconsistency
(over-writing array elements) - 3

-- max -5

Interpretations:

Really minor errors:

initialization inside loop -1

rotation correct except one
element is wrong -1

Loop Direction/Loop Consistency:

```
for index :=2 to n do
   s[index]:=s[index-1]       {1111} -2 direction
                                     -3 overwriting

for index :=2 to n do
   s[index-1]:=s[index]       {2341} -2 direction

for index := n downto 2 do
   s[index]:=s[index-1]       {4123} OK

for index:=n downto 2 do
   s[index-1]:=s[index]       {4444}  -3 overwrite
```

Style and Syntax Errors	max	-1
confusing indentation(formatting)	- 1	
redefining standard procedures or using reserved words incorrectly	- 1	
disregard for efficiency(two arrays)	- 1	
unnecessary or useless code	- 1 / 2	
3 or more syntax errors	- 1 / 2	
[] and () confusion	- 1 / 2	

Total Points	☒	+9
Syntax & Style	Max -1	-1
Loop Logic Loop Inconsistency	Max -5	-3
Loop Logic Loop Direction		-2
Loop Logic Minor Error		-2
Incorrect/Missing Loop Termination	Max -2	-1
Incorrect/Missing Loop Initialization		-1
Declaration Missing		-1
Declaration Type Error		-1
Procedure Header Missing Parameter	Max -1	-1
Procedure Header Missing VAR		-1
1985 QUESTION #1 Reader #	☒	Booklet #