

Perspectives on Testing for Programming Aptitude

Jack M. Wolfe, Brooklyn College of the City University of New York, Brooklyn, N.Y.

This paper discusses limitations of programming aptitude tests. The use of multiple-choice type questions, the testwiseness of the college graduate group, and the inclusion of questions of mathematical information tend to diminish the effectiveness of such tests as predictors of success in business programming. Timed tests favor the faster worker over the slower one who is often more accurate and of equal or better logical capability. The usual criterion of a test's validity is itself often not valid as a simulation of the true criterion but is accepted as the equivalent of the criterion because of its measurability and its appearance of plausibility. Comparative results of various occupational groups on the author's *Aptitude Assessment Battery: Programming* test are analyzed based on the scores of more than 11,000 persons tested, including more than 2,700 persons with work experience in programming in 317 companies, institutions and government agencies.

KEY WORDS AND PHRASES: aptitude, aptitude tests, criterion, evaluation, multiple choice questions, occupational groups, programming aptitude, reliability, tests, timed tests, validity

CR CATEGORIES: 2.42

INTRODUCTION

It is not likely that there will ever be a programming aptitude test that is 100%

accurate in predicting the quality of a person's future performance in work in programming. There are many reasons that preclude the measurement of a person's aptitude for programming with the same degree of validity and reliability as could be attained in measuring his height or weight at any given time.

The person's performance on a programming aptitude test is a mental activity that is itself variable, even in response to the same test questions. An important factor that will influence the person's performance on the test is his personal motivation in taking the test at that time. Is he a job applicant or a candidate for promotion or, on the other hand, is he an experienced and highly competent programmer whose ability is well recognized by his manager but who was asked to take the test so that the company can try it out? In the latter case unless the person is compulsive about trying to do to perfection every job on the first try, he is likely to lack the strong personal motivation for the intense mental concentration required to do the more difficult parts of the test completely correctly in all its details as well as in the total logic. You can lead your programmer to the test but you cannot make him sustain a half day's hard work on it unless he has some personal identification with the results of the test. Some of your programmers may feel that your interest in improving the selection of new members of your programming staff is not

of personal benefit to them. This factor may well be of importance in conducting a validation study of a test. The importance of personal motivation must not be ignored. A difficult test should not be administered unless some positive personal motivation is provided.

Even in the case of the job applicant, whom we can assume to have a strong personal motivation to do well, we should realize that we are taking a single sample -- a slice of the person's work over an hour or a few hours on a single day -- and then extrapolating about his future work on the job. Even if a test is generally effective as a predictive instrument, because of the many factors of a temporary and sometimes extraneous nature that may affect the person's work during the test, we should expect to find occasionally that a person whose test performance was only average or even lower is actually evaluated by his supervisor as superior in his on-the-job work in programming.

It must be recognized, too, that a programmer with average mental ability for work in programming may well, and justifiably, be rated as superior by his supervisor on account of his perseverance, motivation, overtime availability, and finally getting the program to work. But just as some aspects of the programmer's personality may lead to an over-evaluation of his aptitude by his supervisor, other characteristics may be the basis of the supervisor's under-evaluation.

Table 1 shows the comparison of scores on the IBM PAT and the rankings of these programmers by their supervisors. The lack of agreement cannot be attributed solely to the test, for the supervisors' evaluations were not based on the same criteria that the test was attempting to measure.

Table 1. Relationship between IBM PAT Scores and Supervisors' Evaluations (Business Group)(1)

PAT SCORES	SUPERVISORS' RANKINGS	
	Lower Half	Upper Half
69 and above	49%	51%
57 - 68	36%	64%
45 - 56	63%	37%
44 and below	58%	42%

The highest scoring group was about equally divided in the supervisors' rankings. It is of interest to note, too, that a higher percentage of the second best level in scores appeared in the upper half of the supervisors' rankings than of the group scoring at the highest level in the test. And again, a higher percentage of the lowest group in test scores appeared in the upper half of the supervisors' evaluations than of

the group scoring above that level on the test.

FACTORS FAVORING COLLEGE GRADUATES

Testwiseness

An important factor contributing to a lack of agreement of test scores and supervisors' evaluations in many aptitude tests is the great difference in educational background of business applications programmers. This group consists of three substantial subgroups, namely, college graduates, high school graduates, and those with one or two years of college work. Relatively few students leave college after three years. The mere taking of a test is more of an ordeal for the high school graduate, who probably has taken very few tests since high school. For the young college graduate, however, test taking is a way of life. The college graduate group is also more testwise about taking tests of the multiple-choice type. These persons are aware that if they can eliminate any of the choices as incorrect, they should select one of the other answers even if it is only a guess. The scoring system, in fact, may make it advantageous for them to guess at random, even without reading the questions if they are short of time. The college graduate applicant is aware of these features of such tests from various courses and tests that he took in the course of his college work. The person who has not been schooled in taking tests of this kind may be inclined not to guess when he does not feel sure of the answer -- even when he has narrowed the choice down to only two possibilities of the five answers presented in the problem.

The characteristic of guessing, which is actually rewarded by the test, may truly be a detriment as a work habit in programming. If something is unclear or appears to be ambiguous to the programmer on the job, we do not want him to guess. We do not want him even to take the more likely meaning. We want him to find out and be sure. Thus the very characteristic that may impel the high school graduate to leave a question unanswered when he is not sure may well be a factor in his being evaluated highly by his supervisor.

Reading

Another source of discrepancy between aptitude test scores in general and supervisors' evaluations is the fact that the college graduate group is by its very nature a group comprised of persons who can learn by reading. The written test may not be a good simulation of the on-the-job environment. A

person who can readily grasp complicated, compactly worded procedures presented in *written form* will appear highly superior on a written test to another person who could learn the same thing if explained and demonstrated by a supervisor or another programmer on a person to person basis with opportunities for questions and answers and further elaboration where necessary. The latter person may finally have as completely satisfactory an understanding of the problem or procedure as the college graduate despite the fact that his learning was inductive rather than deductive.

The high school graduate may well learn from particular cases and then get to understand the generalization. The college graduate is at an advantage on a test that is more likely to be oriented toward generalized instructions which are then to be applied to specific cases. Because a written test cannot permit the person to ask questions for clarification while he is taking the test, it is heavily loaded in favor of the deductive learner. This factor of the strong bias of a written test in favor of the college graduate, the superior reader and the deductive learner probably accounts for the many evidences of business programmers whose on-the-job ratings are far better than their test results.

Mathematical Knowledge

In the case of the IBM PAT scores and the supervisors' evaluations for the group of business programmers one source of the discrepancy is the fact that the test is more heavily loaded toward mathematical applications than is necessary for business programming. Again in this regard the test favors the college graduate group at the expense of those whose study of mathematics ended with high school. For business programming the actual mathematical content of the work is generally very elementary and only a minor part of the job. By far the more important aspect of the work is the logic *in a data processing setting*. No group -- not mathematics majors or computer science majors or college graduates -- or any other group or combination of groups has a monopoly on logical reasoning. To varying degrees we have found persons with above average capabilities for work in programming in every occupational group tested. (See Tables 4 and 5.)

In the IBM PAT test the mathematical content favors the college graduates in the business programmer group while the highly logical business programmer with less training in mathematics will be recognized nevertheless by his supervisor as a superior programmer. For example, one problem in the

PAT requires that the person know the formula for the volume of a cylinder, $V = \pi r^2 h$, and that $\pi = 22/7$. This definitely places the college graduate science major at an advantage. Another type of problem is the 'work' problem: If John can paint a fence in 3 hours and Bill can do it in 4 hours, how long will it take them to paint the fence working together? The mathematically trained person knows immediately that this kind of problem is done by working with reciprocals, that is, to determine the portion of the job done in one hour. But this is not something that he reasoned out by himself at the time of the test. It is a standard procedure that he was taught to use in this kind of problem. This same kind of problem also appears on various intelligence tests that the college graduate probably took somewhere in the course of his school work. The high school graduate on the other hand probably never saw this kind of problem since high school, if, in fact, even then. As an original analysis not many persons would figure out by themselves that they should work with reciprocals in this kind of problem. Success with this problem is more likely to depend on remembering how to do it than on actual logical reasoning.

In another problem the person must know that the sum of the three angles in a triangle is 180° . This is, of course, well known to many persons taking the test but the high school graduate who does not remember this from his high school geometry will probably not be able to figure it out. Thus for one group it may require a significant amount of reasoning while for the other group it is an immediately known item of information. In another problem where it should be known that a full revolution contains 360° the two groups may be considered as knowing this fact, but it is not at all obvious to the high school group that the sum of the angles of a triangle constitutes half of a full revolution.

It is our belief that the inclusion of questions of mathematical information tends to diminish the validity of a programming aptitude test for business programmers although it appears quite justifiable for scientific programmers. It should not be difficult for a person or group that has access to the pertinent data to perform a statistical study that will support or refute this hypothesis. In general, studies show correlations of the order of .30 to .40 between aptitude test scores and supervisors' evaluations. (2,3)

The fact that the IBM PAT is noticeably better with scientific programmers than with business programmers we believe supports our analysis of the test as favoring college graduates and containing items

of mathematical information. The scientific programmers are college graduates thus eliminating a source of discrepancy between test scores and supervisors' evaluations that became apparent with business applications programmers. The mathematical content and the questions of mathematical information are also more pertinent to the person's on-the-job performance in scientific programming than in business programming. The fact that the scientific programmers are also college graduates tends to make the group homogeneous in testwiseness. This in turn eliminates a source of the discrepancy found with the test scores and the supervisors' evaluations for the group of business programmers.

Timed Tests

Another characteristic that tends to decrease the validity of an aptitude test is the rigid timing of the test. The strict adherence to the specified timing is generally deemed necessary in such tests because such timing was an important factor in the establishment of the norms. But such strictly timed tests are virtually always tests that most persons cannot finish in the allotted time. In this sense they become speed tests. To stop a person from working further on a test although he is capable of answering some of the remaining questions correctly is to underestimate the person who works more slowly and carefully and checks his work as he goes along. The very person who has work habits that are highly desirable for work in programming may readily be rated low by a timed aptitude test although he may in fact be the kind of programmer who would have been rated high by his supervisor if he were given the opportunity of employment. The person who reads carefully and deliberately for precise meaning and who has a compulsion for perfection in all his work may actually be eliminated by a timed test. In such a case, moreover, the error made by the aptitude test would probably never come to light.

These observations regarding the nature of most aptitude tests as speed tests explain why the correlation between aptitude tests and training class ratings is about .45 to .50 while it is generally in the .30 to .35 range with supervisors' evaluations for business programmers. (2) The training instructor's evaluation is likely to be based on the person's performance on timed tests while the supervisor is likely to base his evaluation on the person's actual effectiveness in job performance, which is never timed as is a test. The slower worker on a test, moreover, may not actually be a slow worker at all from the viewpoint of the supervisor.

If he is highly accurate and desk checks his work as he goes along, he will require fewer debugging tests and may quite possibly even be rated by his supervisor as a faster than average worker.

In our study of our *Aptitude Assessment Battery: Programming* test based on the results of 727 programmers and trainees in 185 companies, institutions and government agencies we found little correlation between speed and score. The test is in effect open-ended with regard to time. The correlation was only .186 although we eliminated from the study the lowest scoring 10% of the group because they tended to omit problems that they did not understand. Thus their appearance of being fast workers was often the result of their not doing the complete test. Table 2 shows the relation of score and speed.

Table 2. Relation of Score and Speed (4)

SCORE	SPEED			
	Slow- est 25%	Next 25%	Next 25%	Fast- est 25%
Highest 25%	21%	20%	25%	34%
Next 25%	22%	24%	26%	28%
Next 25%	24%	29%	27%	20%
Lowest 25%	34%	27%	21%	17%

These results were calibrated to give equal weight to programmers and trainees. As many as 21% of the persons who scored in the top 25% of the group actually worked at a pace that placed them in the slowest quarter of the group. In on-the-job programming the person works at the pace that is best suited for himself. In this important respect a rigidly timed test is not a very good simulation of the on-the-job working environment. This discrepancy contributes to a decreased validity of a timed test.

Authors of timed tests are likely to believe that the timing is necessary because of the norms and, moreover, that a person's rating would not be changed to an appreciable extent even if he had more time. Their basis for this belief is that the problems are graded in difficulty and that the person taking the test has reached the limit of his capabilities in the allotted time.

Our own studies contradict such beliefs. In fact the inconsistencies in maintaining the rigidly timed speed tests become apparent when we ask what the objection would be to allowing more time for those who need it if it is really true that they have already reached the limit of their power in the allotted time. Norms in which speed is important are appropriate in assessing a person's aptitude for working on an assembly line in a factory but they seem most inappropriate as a major consideration when compared to the far

more important factors of accuracy and logical ability. In our opinion the norms should not have been dependent on speed, which is the case when time is called at a point where the person taking the test is capable of doing some of the problems that he did not reach. To then use these norms as a reason for not allowing more time is indeed to compound the serious error of making a programming aptitude test a speed test.

We have found that even with problems of graded difficulty there are very many instances where a person was unable to do a problem but was able to do correctly another problem that was established statistically as a more difficult problem. Problems can be graded by difficulty for a group but the research worker will soon find that there are numerous cases of individuals for whom the problems did not rank in the same order of difficulty as was found for the group as a whole.

We have found a great variation in speed even for persons who scored in the top 25% of our norm group of persons actually employed in programming. Almost one quarter of this superior group, 21%, worked at a pace that placed them in the slowest 25% of the norm group, which included persons of average and below average scores. This slow but capable group required from 50% to 100% more time to complete the test than the fastest 25% of the norm group. On a rigidly timed test of the usual type, this group of capable but 'slow' workers would probably have been eliminated from further consideration for employment in programming.

The personnel director should not delude himself into thinking that by administering a programming aptitude test that is also a speed test he will actually be selecting those who are both capable and fast workers. This would be a serious mistake. Every time that a basically superior applicant is eliminated because of a speed test, that position is being filled with a person of lower logical capabilities. When the personnel director is buying speed, he is paying for it with a rarer and more valuable coin.

We suggest an experimental study that can be conducted by any user of one of the various timed tests of programming aptitude. Administer the test with the usual timing but instead of stopping the person's work merely have him place a line on his paper showing the last problem that he finished on that part of the test when time was called. Then permit him to continue on that part of the test for the same amount of time as additional time if he so wishes. During this extension of time he may change his answers to any of the earlier problems but he should leave both answers annotating them clearly so that his paper can be graded for two ratings. The

first rating should be based on his work under the specified timing instructions for the usual administration of the test. The second rating should be based on his work as amended and extended with the additional allowance of time. From our own studies regarding time we believe you will find two very important results:

- 1) a substantial number of persons will improve their ratings with the additional time allowance, and
- 2) a substantial number of persons will be ranked significantly differently relative to the rest of the group in the two ratings. The extended timing will not merely have the effect of raising everyone's score while preserving their ranking.

Type of Questions

Another source of dissimilarity between most aptitude tests and the on-the-job environment is that the aptitude tests generally consist of a large number of short problems. These tests do not evaluate the applicant's aptitude for prolonged concentration on a long sequence of steps. The bright applicant who thinks very quickly will do well on such tests; he may find attractive the quick jumping from one question to another. But getting a program to work often requires extensive and meticulous concentration on very detailed work. The kind of person who may do very well on short type questions may not be capable of the sustained concentration required in debugging from a memory dump. He may not have the perseverance or the high threshold of frustration to work on a difficult problem in depth. If an aptitude test does not provide an assessment to work on a problem intensively and in depth, it should not be surprising when it is sometimes found that after an expensive period of training the trainee decides that he does not wish to continue in the field of programming.

Criterion

An aptitude test should not be the sole basis for predicting a person's future work in programming. Successful work depends on many factors of personality and character traits. Such factors as dedication, dependability, cooperation, perseverance, general agreeableness and many other characteristics may not be included within the scope of the test but are important considerations regarding the person's overall performance on the job.

If supervisors' evaluations are used to validate a programming aptitude test, or if the grades of a training instructor are used

for that purpose, such overall evaluation of the person's work is not what should be used as the criterion. Such a broad evaluation will include many factors that the aptitude test is not designed to measure. The supervisors and training instructors should be asked to evaluate only the logical capabilities of the person for work in programming and to neither reward nor penalize any other characteristics of the person regardless of their relevance to an overall evaluation.

The overall evaluation is of importance to the personnel department and to the programming supervisor, but it should not serve as the criterion in a validation study of a test of the kind of logical ability necessary for work in programming. The applicant's personality characteristics, dependability and dedication to the needs of the department should not be included in the criteria unless the test makes claims of measuring those attributes.

In a study of the validity of a school edition of a programming aptitude test we asked the instructors to submit two evaluations for each student tested. The first evaluation was to be based on the overall work in the course as a whole while the second evaluation was to be based solely on the student's logical capabilities. Table 3 shows the coefficients of correlation between the test results and the instructors' evaluations.

Table 3. Correlations of Wolfe P A T (school edition, experimental form) and Evaluations by the Schools

School Group	No. of Students	Evaluation 1, Overall	Evaluation 2, Logical
High school	93	.679	.697
Private dp school	48	.410	.444
Junior college	165	.538	.561
Total	306	.575	.618

It should be noted that in each type of school the test evaluations correlated better with the instructors' evaluations of the students' logical capabilities than with the overall evaluations made by the same instructors. It is likely, moreover, that the correlation with logical ability is even higher than appeared in Table 3 because for some of students whom the instructor might not have gotten to know very well, the evaluation of logical ability was likely to have been influenced by the student's overall work in the course.

It is likely that all aptitude tests will show a higher correlation with the cri-

terion if the specifications of the criterion are properly delimited. A test should not be held responsible for predicting aspects of the person's work not included in the scope of the test.

OCCUPATIONAL GROUPS

Overlapping of Occupational Groups

In the search for persons with aptitude for work in programming, particularly among those who wish to start a career in programming after working in some other field, it is important for the personnel director and the programming supervisor to be aware that while some occupational groups show higher aptitude for programming than other groups, there is a very substantial amount of overlap in the distributions of most groups. No individual should be accepted or rejected because of his previous occupational group. The objective is to identify the highly capable person regardless of the characteristics of his occupational group.

Group characteristics may serve as a guide, however, and to indicate the proportion of superior persons one may expect to find in the various occupational groups. Table 4 shows some of the results of testing more than 11,000 persons with the *Aptitude Assessment Battery: Programming* test in more than 500 companies, institutions and government agencies in the period from December 1967 through March 1971. The tests were administered by the employer organization and were then sent to the publisher(5) for evaluation by personnel trained by and under the supervision of the author. The test is not made available to individuals.

Evaluation Report

The evaluation report tells not only the person's score and percentile ranking but also includes information about the person's strengths and weaknesses as revealed by his work on the test. There are only five problems on the test and the test requires an average of about three hours. The person's entire work is submitted and not merely his final answers.

An evaluation report is important because a score or percentile rank, being one-dimensional, cannot adequately describe the various aspects of the person's aptitude, which, of course, is actually multi-dimensional. Two persons with the same score may display quite different characteristics pertaining to capabilities for work in programming. One may be highly accurate and may lose points in the more intricate logical work of the test. The other may do the in-

tricate problems completely correctly but may make careless errors in the problem of merely following a long sequence of instructions. A very detailed scoring guide aids in attaining a high degree of objectivity and reliability in the grading of the test books.

Norm Group

The group used for the establishment of norms consisted of 2783 persons with actual employment in programming. These persons were tested in 317 companies, institutions and government agencies. The results of the programmer/analysts and the systems analysts with programming experience were given single weight, the results of current programmers were given double weight, and the programmer trainee group was given triple weight in the establishment of the norms. The person's time taken to complete the test was utilized in ranking persons with the same score.

In addition to the norm group various occupational and other groups are tabulated in Table 4. The median of each group is shown in terms of its percentile rank in the norm group. Thus the entry '65th' for the group of computer science majors means that the median score of the computer science majors fell at the 65th percentile of the norm group. Thus the computer science majors who were job applicants actually might be considered the most superior group by this measure. Also it is seen from Table 4 that the computer science group had only 16% scoring in the below average range where the norm group had 30%. And the computer science group had as many as 43% scoring in the above average range where the norm group had only 30%. There were 68 persons tested as job applicants who had majored in computer science in college. Our definition of computer science refers to a bachelor's degree in a regular college. It does not include curricula in data processing, computer technology, or private programming schools.

Computer Science/Math/Physics

Table 4 shows these groups of new college graduates as the most promising groups for recruitment. These results tend to corroborate the validity of the test as a test of aptitude rather than knowledge of programming. The math/physics group consisted mostly of persons with no instruction in programming or with a single course in Fortran programming. The computer science majors, however, had a substantial course of study including various courses in programming. Since most colleges do not have a major in computer science, it is likely that the computer science majors are basically

the same calibre of students as those in the math/physics group of other colleges and who have an interest in computers as a career. The results in Table 4 show these groups as virtually equivalent in programming aptitude as measured by this aptitude test.

It is of the utmost importance, however, that the personnel director should not assume that every member of these superior groups will prove to be superior individually. Although the 16% and 18% below average scorers were even fewer proportionately than for the norm group of employed programmers, they still constitute a substantial number who would not be recommended for employment in programming on the basis of their performance on this test.

Programmers

Of the groups already employed in programming, the programmer/analysts and the systems analysts with programming experience performed better on the test than did the programmer group, which in turn did better than the trainee group. The relative ranking of these three groups that constituted the norm group may well be due to the fact that the trainee group still contains its potential dropouts and the analysts group consists of programmers who were promoted. These rankings tend to support the hypothesis of the validity of the test.

Instructors

Of particular importance to the personnel director is the observation that programming instructors do not show as high an aptitude for programming as do the programmers. Although the instructors obviously had more knowledge of programming than the group of trainees, their performance on this aptitude test was comparable with that of the trainees.

Similarly the group of mathematics teachers does not show as high an aptitude for programming as the group of mathematics majors who applied directly for employment in programming without first working as mathematics teachers.

Systems Analysts (without programming experience)

This group showed significantly less aptitude for programming than the group of systems analysts who came up through the ranks of programming. The systems analysts without programming experience performed slightly less well on the test than the group of programming trainees. With proper selection from this group, however, a sub-

Table 4. Performance of Computer Related Groups on *Aptitude Assessment Battery: Programming*

GROUP	No.	Norm Percen- tile of Group Median	Below Average (Bottom 30% of Norm Group	Average (Middle 40% of Norm Group)	Above Average (Top 30% of Norm Group)
Computer science majors	68	65th	16%	41%	43%
Mathematics/physics majors	633	63rd	18%	37%	45%
Analysts with programming experience	300	60th	18%	43%	39%
Programmers	2076	51st	29%	40%	31%
NORM GROUP	2783	50th	30%	40%	30%
Programming trainees	407	45th	35%	39%	26%
Programming instructors	23	45th	39%	48%	22%
Mathematics teachers	85	42nd	34%	39%	27%
Systems analysts without programming	139	38th	40%	34%	26%
D.P. supervisors	141	30th	50%	28%	22%
Private d.p. school (one or two courses)	114	29th	51%	37%	12%
Private d.p. school (1 yr full time or equiv.)	60	27th	55%	35%	10%
Computer operators	924	24th	58%	28%	14%
Tab operators	248	22nd	60%	27%	13%
Keypunch operators	80	10th	79%	12%	9%

stantial number of above average persons can be found -- 26% as compared with the norm group of 30% in the above average level.

D.P. Supervisors

Although this group does not rank high on the scale in Table 4, it is because there is a substantial number, 50%, ranking below average in programming aptitude. There is still a suitably large portion, 22%, who can be identified as of superior potential for work in programming. The personnel director will probably not wish to lose a superior supervisor and gain a mediocre programmer, but it would be very worthwhile to identify the d.p. supervisor with high aptitude for programming. Because of his supervisory experience this person is likely to become a superior systems analyst after perhaps two years of experience in programming. His overall view of the flow of work in the company along with his knowledge of the work of the various departments when combined with his successful experience in working well with people augur well for the d.p. supervisor with a superior record in programming to

become a superior systems analyst. A careful selection from the d.p. supervisory group for training in programming may well prove to be a highly worthwhile investment for the company. Of particular importance, too, is the fact that a d.p. supervisor so selected is likely to remain in the company as a long term employee. He has roots in the company and is not likely to seek other employment after his training in programming.

Private D.P. Schools

Of the persons who attended a private data processing school for instruction in programming, those who attended a shorter time seem to have done slightly better on the aptitude test than those who completed a full course of study requiring at least one year of full time attendance or two years part time. But these two groups were not equivalent initially. There was a higher proportion of college graduates in the short term group.

Operators

From the fact that so many computer operators were tested by their companies it appears that personnel directors and the computer operators themselves consider this group as a natural source for recruitment for programming. Selection should be made very carefully, however, because the great majority of computer operators do not show superior aptitude for work in programming.

Only about one-fourth as many tab operators as computer operators were tested by their companies. Both groups did approximately equally well on the test. If the tab operator with superior aptitude for programming has been identified, the personnel director would do well to assign him for programming training directly, without first training him for computer operator. The tab operator with superior capabilities for work in programming does not need the work experience as a computer operator first any more than does the college graduate who is newly employed as a programming trainee.

Even the unlikely group of keypunch operators, which had more than half its group scoring at the level of the lowest 10% of the norm group, actually showed as many as 9% scoring in the above average level of the norm group. Keypunch operators who request consideration for programming training should not be denied such consideration. Of the seven keypunch operators who performed in the above average range of the test, only one was a college graduate. Three were high school graduates, two had one year of college, and one had three years of college.

Other Groups

Of the various occupational groups other than those reported in Table 4 above, the only ones that performed better on the test than the norm group are the professional engineers and the professional statisticians.

The median of the 90 engineers' scores fell at the 55th percentile of the norm group. The engineers had 30% scoring below average, 33% average, and 37% above average.

The median of the 136 statisticians fell at the 53rd percentile of the norm group. The statisticians had 27% scoring below average, 41% average, and 32% above average.

Table 5 shows other occupational groups and their percentage that scored in the above average range of the norm group.

Table 5. Other Occupational Groups

Group	No.	% Above Average
Recent college graduates (not computer sci/math/physics)	813	24%
Teachers (not math)	129	20%
Personnel dept. employees	47	17%
Credit evaluators	45	16%
Office procedures analysts	34	15%
Supervisors (not d.p.)	241	15%
Accounting personnel	459	14%
Technicians	555	13%
Clerks/secretaries	665	8%
Recent graduates of 2-year courses of study	334	8%
Management trainees	40	8%
Salesmen/buyers	86	7%
Bank tellers	52	6%
Production control workers	75	5%
Customer service personnel	102	5%
Police and revenue officers	82	5%
Insurance underwriters/agents	30	3%

Comparative analyses of these groups may be seen in (6).

SUMMARY

One of the most needed improvements in the field of testing for programming aptitude is the selection of criteria against which tests should be validated. A person's performance on the job is many faceted and is not a suitable criterion against which to measure the validity of a test. As nearly as possible the criteria should reflect only those aspects of the work that the test is attempting to measure. The test result should be interpreted by the personnel director as a measure of the logical capabilities of the person for work in programming. He, and not the test, should attempt to evaluate the person's conscientiousness, dependability, cooperativeness, etc. Thus these and similar factors should not be involved in the criteria against which the test will be validated.

Test problems must be carefully screened to assure that they do not favor the person with more education. They should not depend on recall of information biased in favor of the college graduate. The reading that is necessary should be pertinent to the field of data processing or should be on a level that would be comprehensible to a person with the requisite logical capabilities and no more than a high school education in formal schooling. Vocabulary tests probably discriminate against the high school graduate and favor the college graduate, particu-

larly if the words are not of the types likely to be encountered in work in data processing.

The rigid timing of tests should be investigated. It is likely that all of the current tests which do have such restrictions would show an improvement in validity if the time for each part of the test were doubled, provided that the criterion used in validating the test does not itself contain speed as a factor. In doubling the time for the test, it will be necessary to develop a new table of norms, which, in effect, will tend to accentuate the use of the test as a power test rather than a test in which speed plays an important role.

All occupational groups tested were found to contain some persons with superior aptitude for work in programming. No person should be denied consideration for work in programming merely because of his former occupation. But some occupational categories can be recruited more actively than others because of their higher incidence of persons with superior aptitude for programming. On the other hand, no person should be accepted for employment in programming merely on the basis of his being a member of one of the superior groups. Even those superior groups contained a substantial number of individuals who do not appear to possess the capabilities for successful work in programming.

REFERENCES

1. Reinstedt, R. N. et al, 'Computer personnel research group programmer performance prediction study' (RM-4033-PR), The RAND Corporation, Santa Monica, Calif., March 1964, as printed in Proceedings of the Fifth Annual Computer Personnel Research Conference, June 1967
2. Cross, E. M., 'The behavioral styles of computer programmers', Computer Personnel Research Group, Annual Conference, 1970
3. Hall, R. S., 'The construction of a selection battery for programmers adapted to South African conditions', Computer Personnel Research Group, Annual Conference, 1970
4. Wolfe, J. M., 'Testing for programming aptitude', Datamation, April 1969
5. Programming Specialists, Inc., P. O. Box 160, Brooklyn, N. Y. 11234
6. Wolfe, J. M., 'A new look at programming aptitudes', Business Automation, August 1970

Additional References Not Cited Specifically

SIGCPR, all publications of proceedings of annual conferences

McNamara, W. J. and Hughes, J. L., 'A review of research on the selection of computer programmers', Personnel Psychology, 1961

Oliver, T. C. and Willis, W. K., 'A study of the validity of the programmer aptitude test', Educational and Psychological measurement, vol. 23, no. 4, 1963

Palermo, J. M., Computer Programming Aptitude Battery, 1967, Science Research Associates, Inc., Chicago, Illinois