

# An Assessment of the Computer Science Major Field Achievement Test (MFAT)

Russ. Abbott

*Department of Computer Science, California State University, Los Angeles, California*

**Abstract:** As part of our assessment program, the Computer Science Department at California State University Los Angeles instituted a course in which students prepare for and then take the Computer Science MFAT. That course was offered for the first time during the Fall '01 Quarter. Our experience casts doubt on the usefulness of the MFAT for assessment purposes.

## Our Results

The MFAT was offered as part of a course in which students explicitly prepared for the test. During the approximately 5 weeks prior to the test, we worked through sample test questions and discussed test-taking strategies.

Cal State Los Angeles is neither PhD-granting nor an otherwise high-profile institution. This was our first experience with the MFAT. Eight students (who seemed fairly representative of our student population) registered for the class. They all took the test. We were not expecting particularly high scores. The MFAT score range is 120 .. 200. I was surprised by how well we did.

- Our scores (and percentiles) were 142 (38), 146 (48), 150 (57), 154 (66), 164 (83), 166 (86), 168 (88), 170 (90).
  - Our lowest score (142) was at the 38<sup>th</sup> percentile.
  - Our mean (157.6) and median (159) scores were at the 74<sup>th</sup> and 77<sup>th</sup> percentiles respectively.
  - Our highest score (170) was at the 90<sup>th</sup> percentile.
- Our mean score was above those of 87% of institutions that took the test. (I don't know which institutions took the test..)

The MFAT provides sub-scores. The **Percentage correct** column refers to the percentage of questions in that area that our students got correct overall.

<b>Subsection</b>	<b>Percentage correct</b>	<b>Percentile</b>
<i>Programming Methodologies</i>	62.6	90
<i>Software Systems</i>	40.8	69
<i>Computer Organization &amp; Architecture</i>	30.5	68
<i>Theory &amp; Computational Mathematics</i>	50.9	85

- In our best area, *Programming Methodologies*, we answered only 62.6% of the questions correctly. In most tests, this would be a failing score. Yet in the MFAT, this score was at the 90<sup>th</sup> percentile.
- In our worst area, *Computer Organization & Architecture* (many of these were circuit theory questions), we answered only an embarrassing 30.5% of the questions correctly. Nonetheless, this put us at the 68<sup>th</sup> percentile.
- Of course, our scores are not the point. The lowest percentage correct scores at the 99<sup>th</sup> percentile are as follows.

<i>Programming Methodologies:</i>	74%
<i>Software Systems:</i>	61%
<i>Computer Organization &amp; Architecture:</i>	46%
<i>Theory and Computational Mathematics:</i>	61%

## Assessment

Does it really make sense to have a test in which one can do quite badly on an absolute scale and still achieve a high percentile ranking? One can imagine four (not mutually exclusive) conclusions.

1. The test is not a good way to assess Computer Science programs.
2. Most computer science students graduate having learned very little.
3. The Cal State Los Angeles Computer Science program is in the top 12% in the country.
4. Our MFAT preparation course is very effective.

## Test Critique

The MFAT is a bad test—as is the GRE on which it is based.

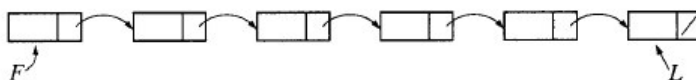
- It asks students to perform in a timed setting tasks that they would normally do in a non-timed setting.
 

The test is organized as two 60 minute tests, each of which includes 30 questions. Students are thus constrained to average 2 minutes/question. The constraint is made relatively rigid in that time not used in one 60 minute segment cannot be used in the other.

Many of the questions require thinking time—no matter how much one knows. That a student may not be able to answer some of these questions in two minutes is not a good measure of that student's capabilities.
- Some of the questions require the student to identify information that would come out during debugging and testing even if the student failed to notice it during initial development.
- Many of the questions are quite problematic. Since the MFAT website ([www.ets.org/hea/mft/compsci.html](http://www.ets.org/hea/mft/compsci.html)) did not provide a

sample MFAT test, the following example questions are taken from the Computer Science GRE subject test<sup>1</sup>.

9. Consider a singly linked list of the form

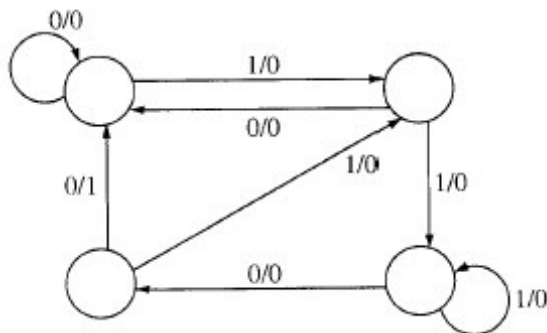


where  $F$  is a pointer to the first element in the list and  $L$  is a pointer to the last element in the list. The time of which of the following operations depends on the length of the list?

- (A) Delete the last element of the list.
- (B) Delete the first element of the list.
- (C) Add an element after the last element of the list.
- (D) Add an element before the first element of the list.
- (E) Interchange the first two elements of the list.

The correct answer is (A). If answer (C) were eliminated, the pointer  $L$  would not be needed. (It is no help in deleting the last element because the next-to-last element must be set to **null**.) If  $L$  were not provided, more students would get the problem right.  $L$  is included primarily to mislead test takers, a bad reason to include information in a question.

11. Consider an output-producing, deterministic finite state automaton (DFA) of the kind indicated in the figure below, in which it is assumed that every state is a final state.



Assume that the input is at least four bits long. Which of the following is(are) true?

- I. The last bit of the output depends on the start state.
  - II. If the input ends with "1100", then the output must end with "1".
  - III. The output cannot end with "1" unless the input ends with "1100".
- (A) I only
  - (B) II only
  - (C) I and II only
  - (D) II and III only
  - (E) I, II, and III

This is a bad question because the correct answer (D) seems to test only the ability to work through lots of detailed cases. Is there an easy way to see that the last bit of the output does not depend on the start state, which implicitly could be any state? I didn't see it in 2 minutes—and I still don't.

19. If the expression  $((2 + 3) * 4 + 5 * (6 + 7) * 8) + 9$  is evaluated with  $*$  having precedence over  $+$ , then the value obtained is the same as the value of which of the following prefix expressions?

- (A)  $+ + * + 2 3 4 * * 5 + 6 7 8 9$
- (B)  $+ * + + 2 3 4 * * 5 + 6 7 8 9$
- (C)  $* + + 2 3 4 * * 5 + + 6 7 8 9$
- (D)  $* + + + 2 3 4 * * 5 + 6 7 8 9$
- (E)  $+ * + * 2 3 4 + + 5 * 6 7 8 9$

There is nothing difficult about this problem. But why is the expression so long? If one is a good test-taker, one notices that the two outermost operators are “+” and “+”, which means that the only possible correct answer is (A). Should we really be testing test-taking skills?

21. Consider a computer design in which multiple processors, each with a private cache memory, share global memory using a single bus. This bus is the critical system resource.

Each processor can execute one instruction every 500 nanoseconds as long as memory references are satisfied by its local cache. When a cache miss occurs, the processor is delayed for an additional 2,000 nanoseconds. During half of this additional delay, the bus is dedicated to serving the cache miss. During the other half, the processor cannot continue, but the bus is free to service requests from other processors. On average, each instruction requires 2 memory references. On average, cache misses occur on 1 percent of references.

What proportion of the capacity of the bus would a single processor consume, ignoring delays due to competition from other processors?

- (A)  $\frac{1}{50}$
- (B)  $\frac{1}{27}$
- (C)  $\frac{1}{25}$
- (D)  $\frac{2}{27}$
- (E)  $\frac{1}{5}$

Here is one way to think this through.

1. In 100 instructions a processor makes 200 memory accesses of which 1% or 2 fail.
2. These 100 instructions take  $100 * 500 + 2 * 2,000 = 50,000 + 4,000 = 54,000$  nanoseconds.
3. During these 54,000 nanoseconds, the bus was used for  $2 * 1,000 = 2,000$  nanoseconds.
4. Therefore the processor used  $2/54 = 1/27$  of the bus, answer (B).

The first time I thought this through, I forgot that the memory accesses that use the bus add to the total time. So in step (3), I got a total of 50,000

nanoseconds, which yields  $1/25$  as the answer. Since that is one of the given answers, I would have gotten the question wrong.

All of the possible answers can be reached with simple arithmetic errors. So even if the student sees through the question (which itself is not clear about what has to be done) and does the required arithmetic, s/he may easily get it wrong, especially given the time pressure.

According to the test booklet, only 21% of test takers get this question correct. So this relatively simple question (which tests primarily reading comprehension and arithmetic) is used to identify the top Computer Science students. This is not a good way to make this distinction.

23. A particular disk unit uses a bit string to record the occupancy or vacancy of its tracks, with 0 denoting vacant and 1 denoting occupied. A 32-bit segment of this string has the hexadecimal value D4FE2003. The percentage of occupied tracks for the corresponding part of the disk, to the nearest percent, is
- (A) 12%                      (B) 25%                      (C) 38%                      (D) 44%                      (E) 62%

This question is really asking about the percentage of 1's in the hex string D4FE2003. The correct answer is (D). Is this a good way to ask if a student can transform hex to binary?

24. A program that checks spelling works in the following way. A hash table has been defined in which each entry is a Boolean variable initialized to *false*. A hash function has been applied to each word in the dictionary, and the appropriate entry in the hash table has been set to *true*. To check the spelling in a document, the hash function is applied to every word in the document, and the appropriate entry in the hash table is examined. Which of the following is (are) correct?
- I. *true* means the word was in the dictionary.
  - II. *false* means the word was not in the dictionary.
  - III. Hash table size should increase with document size.
- (A) I only  
(B) II only  
(C) I and II only  
(D) II and III only  
(E) I, II, and III

This is a garden-path question. Because it is possible for an incorrectly spelled word to hash to a table entry associated with a correctly spelled word, the correct answer is (B) not (C). The way the question is written (and because of time pressure), one is led to think that hashing to a *true* hash table entry means a correct spelling. Only 28% of students noticed this trap!

This could be the basis for a reasonable question if it were asked in the form: *What is wrong with this algorithm?* But that would make the question too easy. A bug of this sort would certainly come out during debugging/testing. Not noticing it immediately, especially given the time pressure, is not very significant.

57. It is known that the language  $L \subseteq \{a,b\}^*$  that consists of all strings that contain an equal number of  $a$ 's and  $b$ 's is context-free. Let  $M$  be the regular language  $a^*b^*$ . Which of the following is (are) true?

- I.  $L \cap M$  is a context-free language.
  - II.  $L \cap M$  is a regular language.
  - III.  $L \cap M = \{a^n b^m \mid n \text{ is a positive integer less than integer } m\}$ .
- (A) None  
 (B) I only  
 (C) III only  
 (D) I and III  
 (E) II and III

This is another relatively trivial question that is confusing because of how it is presented. The question is asking about  $L \cap M = \{a^n b^n \mid n \geq 0\}$ . The answer is (B), but only 43% of test takers got it right.

61. A block of 105 words of memory is used for dynamic storage for objects of sizes 3 and 10 words. The operations supported by the storage are:

```

x := alloc(n)  {Allocate any block of n consecutive words and }
                  {return its starting address ; note that 3 and   }
                  {10 are the only legal values for n.           }

free(y)  {Make the previously allocated block starting at }
            {address y available for re-use.                }
  
```

At a point where a certain request for allocation of a block of words cannot be granted because of a lack of a sufficiently long block of consecutive words to fill that request, what is the minimum possible number of words that might actually be in use?

- (A) 10                      (B) 24                      (C) 53                      (D) 96                      (E) 103

This is an easy question that takes a long time to read and that looks complex. Only 24% of students get it correct. The correct answer is (B). Fill memory with 12 blocks, each of which consists of 9 empty words followed by 3 allocated words. That leaves nothing but blocks of 9 words (including 9 words at the end). So a request for 10 words cannot be satisfied. Only 12 blocks of 3 words, i.e., 24 words, are used.

62. Languages with a structure that implements abstract data types (e.g., a C++ class) can prevent access to components of this structure by all operations except those that are part of this structure. However, definitions of such a structure often contain declarations of components of the structure (e.g., the header file for a C++ class may contain declarations of its private components).

For such a language, an object's name could be bound at run time to stack storage for its component values (direct representation) or to a stack pointer referencing heap storage for its component values (indirect representation). Which of the following statements about comparisons between direct and indirect representations is (are) true?

- I. Indirect representation noticeably increases compilation time.
- II. Direct representation decreases the time needed to access components of a variable.
- III. When the storage size of some private component of a variable changes, indirect representation minimizes the number of recompilations of source modules that must be performed.

- (A) I only
- (B) III only
- (C) I and II only
- (D) II and III only
- (E) I, II, and III

I've never heard of the terms *direct representation* (no pointer) and *indirect representation* (pointer), and I'm not sure what III means at all. But I know that I is false and II is true. The only possible correct answer is (D).

66. Church's thesis equates the concept of "computable function" with those functions computable by, for example, Turing machines. Which of the following is true of Church's thesis?

- (A) It was first proven by Alan Turing.
- (B) It has not yet been proven, but finding a proof is a subject of active research.
- (C) It can never be proven.
- (D) It is now in doubt because of the advent of parallel computers.
- (E) It was never believed, but was assumed in order to simplify certain undecidability results.

The correct answer is (C), which itself is somewhat misleading since Church's Thesis isn't unprovable; it isn't intended to be proven. The fact that only 22% of students got this right simply means that we aren't telling students what Church's Thesis means, not that this is a difficult question.

## Conclusion

Although our students did quite well, our experience with the Computer Science MFAT suggests that it is not a good assessment tool.

## Reference

1. **GRE Computer Science Test Practice Book**, Educational Testing Service, <ftp://ftp.ets.org/pub/gre/CompSci.pdf>, 2001.