Name ________________________
Student Number________________________

Software Engineering 2A04

DAY CLASS — Dr. William M. Farmer
DURATION OF EXAMINATION: 3 Hours
MCMASTER UNIVERSITY FINAL EXAMINATION — December 2000

THIS EXAMINATION PAPER INCLUDES 11 PAGES AND 38 QUESTIONS. YOU ARE RESPONSIBLE FOR ENSURING THAT YOUR COPY OF THE PAPER IS COMPLETE. BRING ANY DISCREPANCY TO THE ATTENTION OF YOUR INVIGILATOR.

Special Instructions

The use of any notes and books is permitted during this exam. Answers to the first thirty-four questions (1–34) are to be marked on the exam itself as well as on the OMR scan sheet. In case of discrepancies, the scan sheet is considered to be your answer. Answer the last four questions (35–38) in the space provided on the exam. Do **NOT** use correction fluid on the exam or on the OMR scan sheet. An answer key will be posted on the course Web site.

OMR Examination Instructions

NOTE: IT IS YOUR RESPONSIBILITY TO ENSURE THAT THE ANSWER SHEET IS PROPERLY COMPLETED: YOUR EXAMINATION RESULT DEPENDS UPON PROPER ATTENTION TO THESE INSTRUCTIONS.

The scanner, which reads the sheets, senses the shaded areas by their nonreflection of light. A heavy mark must be made, completely filling the circular bubble, with an HB pencil. Marks made with a pen or felt-tip marker will **NOT** be sensed. Erasures must be thorough or the scanner may still sense a mark. Do **NOT** use correction fluid on the scan sheet. Do **NOT** put any unnecessary marks or writing on the sheet.

(1) Print your name, student number, course name, section number, and the date in the space provided at the top of SIDE 1 (red side) of the form. The sheet **MUST** be signed in the space marked SIGNATURE.

(2) Mark your student number in the space provided on the sheet on SIDE 1 and **fill in the corresponding bubbles underneath**.

(3) Mark only **ONE** choice from the alternatives (A,B,C,D,E or 1,2,3,4,5) provided for each question. For a True/False question, enter a response of A or 1 for True and B or 2 for False. The question number is to the left of the bubbles. Make sure that the number of the question of the scan sheet is the same as the question number on the exam.

(4) Pay particular attention to the Marking Directions on the form.

(5) Begin answering questions using the first set of bubbles, marked “1”.

Continued on page 2

(1) [2 pts.] As a general rule, the scope of variables and procedures should be as narrow as possible. Is this statement true or false?

A.) $\boxed{\text{True.}}$

B.) False.

(2) [2 pts.] A natural number is an ordinal. Is this statement true or false?

A.) $\boxed{\text{True.}}$

B.) False.

(3) [2 pts.] A sparsely documented piece of code written so that what it does is obvious is better than a heavily documented piece of code whose behavior is not obvious. Is this statement true or false?

A.) $\boxed{\text{True.}}$

B.) False.

(4) [2 pts.] The professional responsibilities of software engineers are much different than the professional responsibilities of other kinds of engineers. Is this statement true or false?

A.) True.

B.) $\boxed{\text{False.}}$

(5) [2 pts.] A specification for one part of a software product design may also be an implementation for another part of the design. Is this statement true or false?

A.) $\boxed{\text{True.}}$

B.) False.

(6) [2 pts.] The narrower the scope of a variable, the more descriptive its name needs to be. Is this statement true or false?

A.) True.

B.) $\boxed{\text{False.}}$

(7) [2 pts.] Self-modifying code is easy to write and debug. Is this statement true or false?

A.) True.

B.) $\boxed{\text{False.}}$

Continued on page 3

(8) [2 pts.] Testing tends to be good for finding high-level errors in a software product, while mathematical verification tends to be good for finding low-level errors. Is this statement true or false?

A.) True.

B.) $\boxed{\text{False.}}$

(9) [2 pts.] Consider the interface of the list module in Lab Exercise 6. The equation

$$\mathsf{list\_append}(\mathsf{list\_take}(n, L), \mathsf{list\_drop}(n, L)) = L$$

holds for all $n \in \mathsf{Natural}$ and $L \in \mathsf{List}(\mathsf{Element})$. Is this statement true or false?

A.) $\boxed{\text{True.}}$

B.) False.

(10) [2 pts.] An abstract data type specifies a set of elements inductively. Is this statement true or false?

A.) $\boxed{\text{True.}}$

B.) False.

(11) [2 pts.] Every module is an object, but some objects are not modules? Is this statement true or false?

A.) True.

B.) $\boxed{\text{False.}}$

(12) [2 pts.] There are simple and effective ways to measure software quality. Is this statement true or false?

A.) True.

B.) $\boxed{\text{False.}}$

(13) [2 pts.] A loop that has a strictly decreasing natural number value is necessarily

A.) Correct.

B.) $\boxed{\text{Terminating.}}$

C.) Efficient.

D.) All of the above.

(14) [2 pts.] A state transition diagram graphically shows

A.) What effect inputs have in a finite state machine.

B.) What outputs are produced in a finite state machine.

C.) What transitions are possible between states in a finite state machine.

D.) $\boxed{\text{All of the above.}}$

Continued on page 4

(15) [2 pts.] The interface language of a module should be

A.) Small.

B.) Orthogonal.

C.) Highly expressive.

D.) $\boxed{\text{All of the above.}}$

(16) [2 pts.] If the next state and output relations of a finite state machine are not functions, then the state machine is

A.) Stable.

B.) Minimal.

C.) $\boxed{\text{Nondeterministic.}}$

D.) Undefined.

(17) [2 pts.] Let $f = (\lambda\, x : \mathbf{Z}\, .\, \mathsf{if}(x < 0, -x, x))$ and $g = (\lambda\, x, y : \mathbf{Z}\, .\, f(x - y))$. What is the value of $g(5, 6)$?

A.) -1.

B.) 0.

C.) $\boxed{1.}$

D.) 11.

(18) [2 pts.] Suppose the relation $S = \{(x, y) \in \mathbf{Z} \times \mathbf{Z} : x = y * y\}$ is a specification of an integer square root function. If $f : \mathbf{Z} \to \mathbf{Z}$ satisfies S, then

A.) $f(5)$ must be undefined.

B.) $f(-2)$ must be undefined.

C.) $f(4)$ may equal -2.

D.) $\boxed{\text{All of the above.}}$

(19) [2 pts.] The name of the Unix shell command for changing the access privileges to a file or directory is

A.) $\boxed{\texttt{chmod.}}$

B.) `chper`.

C.) `chown`.

D.) `priv`.

Continued on page 5

(20) [2 pts.] What would be a legitimate reason for rejecting a software product whose design satisfies its requirements specification and whose implementation satisfies its design?

A.) The software product is not readily maintainable.

B.) The requirements specification does not include all the requirements the product must satisfy.

C.) The software product's design and implementation are not reviewable.

D.) $\boxed{\text{All of the above.}}$

(21) [2 pts.] Which component of a Turing Machine corresponds to a computer program?

A.) The two-way tape.

B.) The finite state control mechanism.

C.) The finite set of states.

D.) $\boxed{\text{The transition relation.}}$

(22) [2 pts.] Tables are good for

A.) Specifying functions.

B.) Specifying relations.

C.) Writing code.

D.) $\boxed{\text{All of the above.}}$

(23) [2 pts.] An MIS can be viewed as a ________ of a module.

A.) $\boxed{\text{Blackbox specification.}}$

B.) Whitebox specification.

C.) Description.

D.) None of the above.

(24) [2 pts.] Which of the following is not used to relate input values to output values?

A.) Function description.

B.) Relation description.

C.) $\boxed{\text{Precondition formula.}}$

D.) Postcondition formula.

Continued on page 6

(25) [2 pts.] Which software structure is useful for dividing a product into subproducts?

A.) Entity-relationship.

B.) Abstraction hierarchy.

C.) $\boxed{\text{Uses hierarchy.}}$

D.) All of the above.

(26) [2 pts.] The type of a programming language variable indicates

A.) What kind of number the variable can hold.

B.) $\boxed{\text{What values the variable may have.}}$

C.) Where the variable is stored.

D.) Whether the variable is local or global.

(27) [2 pts.] Which of the following is the least efficient in the use of space (when compiled by a state-of-the-art compiler)?

A.) A tail recursive procedure.

B.) $\boxed{\text{A nontail recursive procedure.}}$

C.) A while loop.

D.) A do loop.

(28) [2 pts.] Which method is the most useful for checking the reliability of a software product.

A.) Testing with planned test case selection.

B.) Testing with wild random test case selection.

C.) $\boxed{\text{Testing with statistical random test case selection.}}$

D.) Inspection.

(29) [2 pts.] Suppose that the implementation of a module is to be tested. Which kind of testing can be reused if the implementation of module is modified or completely replaced.

A.) $\boxed{\text{Blackbox.}}$

B.) Greybox.

C.) Clearbox.

D.) All of the above.

Continued on page 7

(30) [2 pts.] Why does software tend to age?

A.) It is modified as part of routine maintenance.

B.) The environment in which it is used changes over time.

C.) It is not designed for change.

D.) $\boxed{\text{All of the above.}}$

(31) [2 pts.] The secret of a module is hidden in its

A.) Interface.

B.) $\boxed{\text{Implementation.}}$

C.) MIS.

D.) Access functions.

(32) [2 pts.] According to Frederick Brooks's mythical man-month idea, the *worse* way to help a software project behind schedule is to

A.) $\boxed{\text{Add more workers to the project team.}}$

B.) Reschedule the project over a longer period of time.

C.) Cut back on the goals of the project.

D.) Reuse completed work from other projects.

(33) [2 pts.] According to Brooks, the purpose of using a "surgical" team on a software project is to

A.) Enable more workers than usual to participate in the project.

B.) Facilitate rapid prototyping.

C.) Remove the parts of the product that do not meet their specifications.

D.) $\boxed{\text{Improve the conceptual integrity of the final product.}}$

(34) [2 pts.] According to Brooks, the project workbook should

A.) Contain only software specifications.

B.) Be available to only the project managers.

C.) $\boxed{\text{Be a framework for all the project's documentation.}}$

D.) Be a small working document that states the objectives of the project.

Continued on page 8

(35) [8 pts.] Let $\mathsf{Integer}$ be the set of integers and Tree be the set of trees defined in Lab Exercise 2. Consider a procedure P that takes a tree t as input and returns some integer n in t (i.e., $n \in t$ as defined in Lab Exercise 2) as output.

a. Specify P as a relation R according to the relation description model.

Answer: $R = \{(t, n) \in \mathsf{Tree} \times \mathsf{Integer} : n \in t\}$.

b. Describe a function that satisfies R.

Answer: $f : \mathsf{Tree} \to \mathsf{Integer}$ where:

$f = (\lambda\, t \in \mathsf{Tree}\,.\; \mathsf{if}(t \in \mathsf{Integer}, t, f(\mathsf{left}(t))))$.

$\mathsf{left} = (\lambda\, t \in \mathsf{Tree}\,.\; (\mathrm{I}\, t_1 \in \mathsf{Tree}\,.\; (\exists\, t_2 \in \mathsf{Tree}\,.\; t = (t_1, t_2))))$.

Note: There are many other functions and descriptions that are acceptable.

Continued on page 9

(36) [8 pts.] Consider a staple gun with four buttons labelled staple, safety_off, reset, and switch_size. The stable gun has a safety lock and a light that is either blue or green. The behavior of the staple gun is described by the following rules:

a. If the safety lock is off and staple is pressed, a big or small staple is shot. The staple is big if the light is blue and small if the light is green. The safety lock and light are not affected.

b. If the safety lock is on and staple is pressed, nothing happens.

c. The safety lock is off immediately after safety_off is pressed.

d. The safety lock is on and the light is green immediately after reset is pressed.

e. If switch_size is pressed, the color of the light is switched.

Model the behavior of the staple gun as a finite state machine. Let (the press of) each button be an input, and let the outputs be n (for no staple is shot), b (for a big staple shot), and s (for a small staple shot). There are four stable states:

1: The safety lock is on and the light is blue.

2: The safety lock is on and the light is green.

3: The safety lock is off and the light is blue.

4: The safety lock is off and the light is green.

Below are incomplete next state and output tables for the finite state machine. Fill in the missing entries.

ns	staple	safety_off	reset	switch_size
1	1	3	2	2
2	2	4	2	1
3	3	3	2	4
4	4	4	2	3

out	staple	safety_off	reset	switch_size
1	n	n	n	n
2	n	n	n	n
3	b	n	n	n
4	s	n	n	n

Continued on page 10

(37) [8 pts.] Let $\mathsf{List}(\mathsf{Integer})$ be the set of finite lists of integers as defined in Lab Exercise 6 with $\mathsf{Element}$ replaced by $\mathsf{Integer}$. For $L \in \mathsf{List}(\mathsf{Integer})$, let $\mathsf{sum}(L)$ be the sum of the members of L. Consider the following pseudocode definition of the procedure summa which uses the interface functions defined in Lab Exercise 6 (with $\mathsf{Element}$ replaced by $\mathsf{Integer}$):

```
procedure summa(L : List(Integer) : Integer
  s : Integer;
  s ⇐ 0;
  it
    ((list_length(L) = 0 → stop) |
     (list_length(L) ≠ 0 → s ⇐ s + list_head(L);
                           L ⇐ list_tail(L);
                           go))
  ti;
  return s
end procedure
```

a. What is the strictly decreasing natural number value for the loop above?

Answer: $\mathsf{list\_length}(L)$.

b. What invariant for the loop best describes the behavior of the loop?

Answer: $\mathsf{sum}(L_0) = s + \mathsf{sum}(L)$ where L_0 is the inital value of L.

Continued on page 11

(38) [8 pts.] Let Node be a set of abstract nodes and Edge = Node × Node be the set of pairs of nodes. Let a *graph* be a pair (N, E) such that $N \subseteq \mathsf{Node}$, $E \subseteq \mathsf{Edge}$, and

$$\forall n_1, n_2 \in \mathsf{Node}\,.\; (n_1, n_2) \in E \Rightarrow n_1, n_2 \in N.$$

Suppose M is a module for constructing, storing, and examining a graph. M provides the following services:

a. The graph can be reset to the empty graph containing no nodes and edges.

b. A node or edge can be added to the graph.

c. The graph can be checked to see whether a given node or edge is a member of the graph.

Using the state transition model, specify an interface for M consisting of five interface functions. Assume that the state variables of your specification are N and E and that (N, E) is initially a graph and remains a graph after each application of one of the interface functions. For each function, say what its name is, what its input is, when it is defined, and what its output is and how it changes the state variables when it is defined. Put your answers into the table below; one interface function is done for you.

Name	Input	When Defined	Output	State Transition
add_node	$n \in \mathsf{Node}$	always	none	$N \Leftarrow N \cup \{n\}$
add_edge	$(n_1, n_2) \in \mathsf{Edge}$	$n_1, n_2 \in N$	none	$E \Leftarrow E \cup \{(n_1, n_2)\}$
reset	none	always	none	$N \Leftarrow \emptyset,\ E \Leftarrow \emptyset$
node_member	$n \in \mathsf{Node}$	always	$n \in N$	none
edge_member	$(n_1, n_2) \in \mathsf{Edge}$	always	$(n_1, n_2) \in E$	none

Note: Other sets of interface functions are acceptable.

The End