UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 1999

MEng Honours Degrees in Computing Part IV
MSci Honours Degree in Mathematics and Computer Science Part IV
MSc Degree in Advanced Computing
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute

PAPER 4.71

SEMANTICS OF FUNCTIONAL AND OBJECT-ORIENTED LANGUAGES

Tuesday, May 11th 1999, 10.00 – 12.00

Answer THREE questions

For admin. only:
paper contains 4 questions

1a i Give the definition of normal form.

ii Show that $(\lambda xy.x(xy))(\lambda z.z) = \lambda y.y$

iii Give an example of a strongly normalizing term (i.e. a term whose all reduction sequences terminate).

b Define a *linear term* as a $\lambda-$term in which all bound variables appear exactly once in its body up to $\alpha-$equivalence (e.g. $\lambda x.x$ is linear but $\lambda xy.x$ and $\lambda x.xx$ are not).

i Find a linear term in head normal form but not in normal form.

ii Find a linear term with two different reduction sequences.

iii Is the fixed point of a term (as given by the Fixed Point Theorem) linear?

c Define `true` as the term $\lambda xy.x$, `false` as the term $\lambda xy.y$ and given $M_1, \ldots, M_n$ define $< M_1, \ldots, M_n >$ as the term $\lambda x.xM_1 \ldots M_n$.

i Show that by defining `not` as the term $<$ `false`, `true` $>$ one has `not true` $=$ `false` and `not false` $=$ `true`.

ii Show that for all terms M, N by defining `cond`MN as $< M, N >$ one has `cond` MN`true` $= M$ and `cond` MN`false` $= N$.

2a i Give the operational semantics for the PCF boolean constant `cond`.

ii Show that $(\lambda x.\mathtt{cond}\ x\mathtt{tt}\ \mathtt{tt}\ )\mathtt{tt} = \mathtt{tt}$ and $(\lambda x.\mathtt{cond}\ x\mathtt{tt}\ \mathtt{tt}\ )\mathtt{ff} = \mathtt{tt}$.

iii What is the difference between $(\lambda x.\mathtt{cond}\ x\mathtt{tt}\ \mathtt{tt}\ )$ and `tt`?

b i Define a PCF term of type $\mathtt{int} \to \mathtt{int}$ which implements the following function f:

$$f(0) = 5, f(n+1) = f(n-1) + f(n)$$

ii Define a PCF term F of type $(\mathtt{int} \to \mathtt{int}) \to \mathtt{bool}$ such that $F(f) = \mathtt{iszero}(f(3))$

c i Give the translation of the `while` construct $\mathtt{while}\ B\mathtt{do}\ c; \mathtt{od}; c'$ into I.A.

ii Give the operational rules (i.e. the notebook rules) for the IA constants `assign` and `deref`.

3 Consider the fist order typed ς–calculus, the type rules for which are given at the end of the paper.

a Assume that *3* has type *Integer*, and give all possible types for the expression

$$[\ l_1 = \varsigma(x_1{:}A)3,\quad l_2 = \varsigma(x_2{:}A)\ _2.l_1,\quad l_3 = \varsigma(x_3{:}A)\ ]\ ].$$

b What is the minimal type of the following expression, *ie* what are possible type expressions for *B* and *C*?

$$[\ l_1 = \varsigma(x_1{:}B)[\ l_1{=}\varsigma(x_3{:}C)[]\ ],\ l_2 = \varsigma(x_2{:}B)[]\ \ ]$$

c Show the derivation of the type of the expression from b using the type inference rules for the ς – calculus.

d Consider the ςλ-calculus, the following extension of the ς-calculus, whose terms are

a, b	$::=$	x	a variable
	$\mid$	$[\ l_i = \varsigma(z_i : A)\ b_i^{\ i=1..n}\ ]$	an object
	$\mid$	$b.l \Leftarrow \varsigma(z{:}\ A)\ a$	method override
	$\mid$	$b.l$	method call
	$\mid$	$\lambda\ (x{:}A)b\{x\}$	function
	$\mid$	$b(a)$	function application

and whose types are

A, B	$::=$	Top	the biggest type
	$\mid$	$[\ l_i : B_i^{\ i=1..n}\ ]$	object type
	$\mid$	$A \rightarrow B$	method override

i) Extend the type system of the ς–calculus with appropriate type rules for functions and function applications.

ii) Give the minimal type for the following expression

$$[\ l_1{=}\varsigma(x{:}A)\ \lambda(x{:}B)x.l_2{+}3\ \ ].$$

4 Consider the ς–calculus, with terms defined by

a, b	$::=$	x	*a variable*
	$\mid$	$[\ l_i = \varsigma(z_i : A\ )\ b_i{}^{i=1..n}\ ]$	an object
	$\mid$	$b.l \Leftarrow \varsigma(z{:}\ A)\ a$	method override
	$\mid$	$b.l$	method call

and evaluation, for $o \equiv [\ l_i = \varsigma(z_i{:}A)b_i{}^{i=1..n}\ ]$ (l_i distinct) defined by

$$o.\ l_j \ \rightarrow\ b_j\{\{\ x_j \leftarrow o\ \}\} \qquad (j \in 1..n)$$
$$o.\ l_j \Leftarrow \varsigma(y)b \ \rightarrow\ [\ l_j = \varsigma(y)b,\ \ l_i = \varsigma(z_i{:}\ A)b_i{}^{i=(1..n)\setminus j}\ ] \qquad (j \in 1..n)$$

a Write out the steps involved in the evaluation of the term *counter.tick.contents* where counter is defined as

$$counter \equiv [\ cont\ = \varsigma(x\ )0,\ \ tick = \varsigma(y\ )y.cont \Leftarrow \varsigma(z)y.cont + 1\ ]$$

b Consider the terms

$$tt \equiv [\ \ if = \varsigma(x\ )x.then,\ \ then = \varsigma(y\ )y.then,\ \ else = \varsigma(z\ )z.else\ ]$$
$$ff \equiv [\ \ if = \varsigma(x\ )x.else,\ \ \ then = \varsigma(y\ \ y.then,\ \ else = \varsigma(z\ )z.else\ ]$$

which encode in the ς-calculus the meaning of *true* and *false*. Assume further terms *term1*, *term2*, and *booleanTerm,* where *booleanTerm* will evaluate either to *tt* or to *ff*. Encode a conditional expression which will evaluate *term1* if *booleanTerm* returns *tt*, and *term2* otherwise.

c Consider the ςbool-calculus, the following extension of the ς – calculus, with terms defined by

a, b, c	$::=$	x	a variable
	$\mid$	$[l_i = \varsigma(z_i : A\ )\ b_i{}^{i=1..n}\ ]$	an object
	$\mid$	$b.l \Leftarrow \varsigma(z{:}\ A)$ a	method override
	$\mid$	$b.l$	method call
	$\mid$	**true** $\mid$ **false**	
	$\mid$	**if** *a* **then** *b* **else** *c*	

i) Give the necessary additional rules for the operational semantics.

ii) Give a translation from the the ςbool-calculus to the ς-calculus.

iii) Formulate a theorem relating evaluation in the ςbool-calculus and in the ς-calculus.

The three parts carry, respectively, 30%, 20% and 50% of the marks.

End of Paper