4 Computation Theory (AMP)

(*a*) Give inductive definitions of the relations $M \to N$ and $M \twoheadrightarrow N$ of *single-step* and *many-step* β*-reduction* between λ-terms M and N. (You may assume the definition of α-conversion, $M =_\alpha N$.) [6 marks]

(*b*) *Turing's fixed point combinator* is the λ-term $\mathtt{A}\,\mathtt{A}$ where $\mathtt{A} = \lambda x.\lambda y.\, y(x\,x\,y)$. Use it to show that given any λ-term M, there is a λ-term X satisfying $X \twoheadrightarrow M\,X$. [2 marks]

(*c*) The sequence of λ-terms $\mathtt{N}_0, \mathtt{N}_1, \mathtt{N}_2, \ldots$ is defined by $\mathtt{N}_0 = \lambda x.\lambda f.\, x$ and $\mathtt{N}_{n+1} = \lambda x.\lambda f.\, f\,\mathtt{N}_n$. Say that a function $\mathtt{f} \in \mathbb{N}^k \to \mathbb{N}$ is *Scott definable* if there is a λ-term F satisfying that $F\,\mathtt{N}_{n_1} \cdots \mathtt{N}_{n_k} \twoheadrightarrow \mathtt{N}_{\mathtt{f}(n_1,\ldots,n_k)}$ for all $(n_1, \ldots, n_k) \in \mathbb{N}^k$.

(*i*) Show that the successor function , $\mathtt{succ}(n) = n + 1$, is Scott definable. [2 marks]

(*ii*) Show that for any λ-terms M and N, $\mathtt{N}_0\,M\,N \twoheadrightarrow M$ and $\mathtt{N}_{n+1}\,M\,N \twoheadrightarrow N\,\mathtt{N}_n$. Deduce that the predecessor function

$$\mathtt{pred}(n) = \begin{cases} 0 & \text{if } n = 0 \\ n - 1 & \text{if } n > 0 \end{cases}$$

is Scott definable. [2 marks]

(*iii*) By considering the λ-terms $P_m = \mathtt{A}\,\mathtt{A}\,(\lambda f.\lambda y.\, y\,\mathtt{N}_m\,(\lambda z.\, S(f\,z)))$ for a suitable choice of S, or otherwise, prove that the addition function $\mathtt{plus}(m, n) = m + n$ is Scott definable. [8 marks]

3 Computation Theory (AMP)

(*a*) What does it mean for a partial function to be *register machine computable*?

[3 marks]

(*b*) Give definitions of bijective codings of pairs of numbers $(x, y) \in \mathbb{N}^2$ as numbers $\langle x, y\rangle \in \mathbb{N}$; and of finite lists of numbers $\ell \in \mathit{list}\,\mathbb{N}$ as numbers $\ulcorner \ell \urcorner \in \mathbb{N}$.

[3 marks]

(*c*) Let T be the subset of $\mathbb{N}^3$ consisting of all triples $(e, \ulcorner[x_1, x_2, \ldots, x_m]\urcorner, t)$ such that the computation of the register machine with index e halts after t steps when started with $\mathtt{R}_0 = 0, \mathtt{R}_1 = x_1, \ldots, \mathtt{R}_m = x_m$ and all other registers zeroed. Define a function $s \in \mathbb{N}\to\mathbb{N}$ as follows. For each $n \in \mathbb{N}$, $s(n) \in \mathbb{N}$ is the maximum of the finite set of numbers $\{t \mid \exists e, x \in \mathbb{N}.\ \langle e, x\rangle \leq n \ \wedge\ (e, x, t) \in T\}$.

Prove that for all recursive functions $r \in \mathbb{N}\to\mathbb{N}$, there exists some $n \in \mathbb{N}$ with $r(n) < s(n)$. Any standard results about register machines and about recursive functions that you use should be clearly stated, but need not be proved.

[14 marks]

4 Computation Theory (AMP)

(*a*) Give the recursion equations for the function $\rho^n(f,g) \in \mathbb{N}^{n+1} \to \mathbb{N}$ *defined by primitive recursion* from functions $f \in \mathbb{N}^n \to \mathbb{N}$ and $g \in \mathbb{N}^{n+2} \to \mathbb{N}$. [2 marks]

(*b*) Define the class PRIM of *primitive recursive functions*, giving exact definitions for all the functions and operations you use. [5 marks]

(*c*) Show that the addition function $add(x,y) = x + y$ is in PRIM. [2 marks]

(*d*) Give an example of a function $\mathbb{N}^2 \to \mathbb{N}$ that is not in PRIM. [3 marks]

(*e*) The Fibonacci function $fib \in \mathbb{N} \to \mathbb{N}$ satisfies $fib(0) = 0$, $fib(1) = 1$ and $fib(x+2) = fib(x) + fib(x+1)$ for all $x \in \mathbb{N}$.

(*i*) Assuming the existence of primitive recursive functions $pair \in \mathbb{N}^2 \to \mathbb{N}$, $fst \in \mathbb{N} \to \mathbb{N}$ and $snd \in \mathbb{N} \to \mathbb{N}$ satisfying for all $x, y \in \mathbb{N}$

$$fst(pair(x,y)) = x \ \wedge \ snd(pair(x,y)) = y$$

prove by mathematical induction that any function $g \in \mathbb{N} \to \mathbb{N}$ satisfying

$$\begin{aligned} g(0) &= pair(0,1) \\ g(x+1) &= pair(snd(g(x)), fst(g(x)) + snd(g(x))) \end{aligned}$$

for all $x \in \mathbb{N}$, also satisfies

$$\forall x \in \mathbb{N}(fst(g(x)) = fib(x) \ \wedge \ snd(g(x)) = fib(x+1)).$$

[4 marks]

(*ii*) Deduce that the Fibonacci function fib is in PRIM. [4 marks]

3 Computation Theory (AMP)

(*a*) Explain how to code register machine programs P as numbers $\ulcorner P \urcorner \in \mathbb{N}$ so that each $e \in \mathbb{N}$ can be decoded to a unique register machine program $\mathit{prog}(e)$.
[10 marks]

(*b*) Find a number $e_1 \in \mathbb{N}$ for which $\mathit{prog}(e_1)$ is a register machine program for computing the function $\mathit{one} \in \mathbb{N} \to \mathbb{N}$ with $\mathit{one}(x) = 1$ for all $x \in \mathbb{N}$. [2 marks]

(*c*) Why is it important for the theory of computation that the functions involved in the coding and decoding given in part (*a*) are themselves register machine computable? (You are not required to prove that they are computable.)
[2 marks]

(*d*) Define what it means for a set of numbers $S \subseteq \mathbb{N}$ to be register machine *decidable*.
[2 marks]

(*e*) Let $\varphi_e \in \mathbb{N} \rightharpoonup \mathbb{N}$ denote the partial function of one argument computed by the register machine with program $\mathit{prog}(e)$. Prove that $\{e \in \mathbb{N} \mid \varphi_e = \mathit{one}\}$ is register machine undecidable (where one is the function mentioned in part (*b*)). State carefully any standard results that you use in your proof. [4 marks]

4 Computation Theory (AMP)

(*a*) (*i*) What does it mean for a λ-term to be a β-*normal form*? Defining the sets of canonical (C) and neutral (U) λ-terms by the grammar

$$C ::= \lambda x.\, C \mid U$$
$$U ::= x \mid U\, C$$

show that a λ-term is a β-normal form if and only if it is canonical.

[5 marks]

(*ii*) Carefully stating any standard properties of β-reduction, explain why a λ-term reduces to at most one β-normal form (up to α-equivalence).

[4 marks]

(*iii*) Give an example of a λ-term that does not reduce to any β-normal form.

[2 marks]

(*b*) (*i*) Define what it means for a closed λ-term F to *represent* a partial function $f \in \mathbb{N} \rightharpoonup \mathbb{N}$. [4 marks]

(*ii*) The composition of partial functions $f, g \in \mathbb{N} \rightharpoonup \mathbb{N}$ is the partial function $g \circ f = \{(x, z) \mid (\exists y)\ (x, y) \in f \wedge (y, z) \in g\} \in \mathbb{N} \rightharpoonup \mathbb{N}$. Suppose F represents f, G represents g, and f and g are totally defined. Show that $\lambda x.\, G\,(F\,x)$ represents $g \circ f$. [2 marks]

(*iii*) Give an example to show that $\lambda x.\, G\,(F\,x)$ need not represent $g \circ f$ when f and g are not totally defined. [3 marks]

3 Computation Theory (AMP)

(*a*) What does it mean for a register machine to be *universal*? [4 marks]

(*b*) Define what it means for a partial function $f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ to be *register machine computable*. [3 marks]

(*c*) Show that the following functions f, g, h, k are register machine computable.

(*i*) The partial function $f \in \mathbb{N} \rightharpoonup \mathbb{N}$ that is everywhere undefined. [1 mark]

(*ii*) $g(x_1, x_2) = \begin{cases} x_1 - x_2 & \text{if } x_1 \geq x_2 \\ 0 & \text{if } x_1 < x_2 \end{cases}$ [4 marks]

(*iii*) $h(x_1) = \begin{cases} 2^{x_1 - 1} & \text{if } x_1 > 0 \\ \text{undefined} & \text{if } x_1 = 0 \end{cases}$ [4 marks]

(*iv*) $k(x_1, x_2) = 1$ if the register machine program with index x_1, when started with 0 in all registers, halts in at most x_2 steps; and $k(x_1, x_2) = 0$ otherwise. [4 marks]

4 Computation Theory (AMP)

(*a*) Define what it means for a set of numbers $S \subseteq \mathbb{N}$ to be register machine *decidable*. Why are there only countably many such sets? Deduce the existence of a set of numbers that is not register machine decidable. (Any standard results that you use should be clearly stated.) [4 marks]

(*b*) A set of numbers $S \subseteq \mathbb{N}$ is said to be *computably enumerable* if either it is empty or equal to $\{f(x) \mid x \in \mathbb{N}\}$ for some total function $f : \mathbb{N} \to \mathbb{N}$ that is register machine computable.

(*i*) Show that if S is register machine decidable, then it is computably enumerable. [*Hint:* consider separately the cases when S is, or is not empty.] [4 marks]

(*ii*) Show that if both S and its complement $\{x \in \mathbb{N} \mid x \notin S\}$ are computably enumerable, then S is register machine decidable. [*Hint:* consider a register machine that interleaves the enumeration of S and its complement.] [6 marks]

(*c*) Let $\varphi_e : \mathbb{N} \rightharpoonup \mathbb{N}$ denote the partial function computed by the register machine with code $e \in \mathbb{N}$ and consider the set $T = \{e \in \mathbb{N} \mid \varphi_e \text{ is a total function}\}$.

(*i*) Suppose that $f : \mathbb{N} \to \mathbb{N}$ is a register machine computable total function such that $f(x) \in T$ for all $x \in \mathbb{N}$. Define $\hat{f}(x)$ to be $\varphi_{f(x)}(x) + 1$. Show that $\hat{f} = \varphi_e$ for some $e \in T$. [3 marks]

(*ii*) Deduce that T is not computably enumerable. [3 marks]

3 Computation Theory (AMP)

(*a*) Define what is a *Turing machine* and a *Turing machine computation*. [7 marks]

(*b*) What is meant by a *partial function* from $\mathbb{N}^n$ to $\mathbb{N}$? Define what it means for such a partial function to be *Turing computable*. [4 marks]

(*c*) Describe the *Church-Turing Thesis* and some evidence for its truth. [4 marks]

(*d*) Assuming the existence of a universal *register* machine, give an example, with justification, of a partial function that is not Turing computable. [5 marks]

2011 Paper 6 Question 4

Computation Theory

(*a*) State precisely what it means for a function $f : \mathbb{N}^k \to \mathbb{N}$ to be *primitive recursive*, giving exact definitions for all operations you use. [5 marks]

(*b*) State precisely what it means for a function $f : \mathbb{N}^k \to \mathbb{N}$ to be λ-*definable*. [5 marks]

(*c*) For each of the following functions, show (using the definitions you gave) that it is primitive recursive and λ-definable.

(*i*) The function $square : \mathbb{N} \to \mathbb{N}$ given by $square(x) = x^2$. [4 marks]

(*ii*) The function $fact : \mathbb{N} \to \mathbb{N}$ given by $fact(x) = x!$. [4 marks]

(*d*) Give a definition of a function that is λ-definable but not primitive recursive. [2 marks]

2011 Paper 6 Question 3

Computation Theory

(*a*) Explain how we can associate a unique number $\ulcorner P \urcorner$ to each register machine program P. [5 marks]

(*b*) Consider the following two partial functions $S : \mathbb{N} \rightharpoonup \mathbb{N}$ and $T : \mathbb{N} \rightharpoonup \mathbb{N}$ on the natural numbers.

$$S(\ulcorner P \urcorner) = \begin{cases} n & \text{if the program } P \text{ when started with 0 in all registers} \\ & \text{halts after } n \text{ steps;} \\ 0 & \text{otherwise.} \end{cases}$$

$$T(\ulcorner P \urcorner) = \begin{cases} n & \text{if the program } P \text{ when started with 0} \\ & \text{in all registers halts after } n \text{ steps;} \\ \textit{undefined} & \text{otherwise.} \end{cases}$$

(*i*) Which, if any, of S and T is computable and which is uncomputable? [4 marks]

(*ii*) Give full justification for your answers above. State carefully any standard results that you use. [11 marks]

2010 Paper 6 Question 4

Computation Theory

(a) Define Church's representation of numbers n as λ-terms $\underline{n}$. [3 marks]

(b) What does it mean for a partial function $f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ to be λ-*definable*? What is the relationship between λ-definability and computability? [3 marks]

(c) Show that $\texttt{succ}(x_1) = x_1 + 1$ is λ-definable. [4 marks]

(d) Ackermann's function $ack \in \mathbb{N}^2 \to \mathbb{N}$ is a total function of two arguments satisfying

$$\begin{aligned} ack(0, x_2) &= x_2 + 1 \\ ack(x_1 + 1, 0) &= ack(x_1, 1) \\ ack(x_1 + 1, x_2 + 1) &= ack(x_1, ack(x_1 + 1, x_2)). \end{aligned}$$

By considering $\lambda x.\, x\, T\, S$ where $T = \lambda f\, y.\, y\, f\, (f\, \underline{1})$ and S is chosen suitably, prove that Ackermann's function is λ-definable. [10 marks]

2010 Paper 6 Question 3

Computation Theory

(a) Define the notion of a *register machine* and the computation it carries out. [5 marks]

(b) What does it mean for a partial function $f(x_1, \ldots, x_n)$ of n arguments to be *register machine computable*? [3 marks]

(c) Why do there exist partial functions that are not register machine computable? (Any standard results you use in your answer should be carefully stated.) [3 marks]

(d) Consider the following register machine program.

$$
\begin{aligned}
L_0 &: R_1^- \to L_1, L_6 \\
L_1 &: R_2^- \to L_2, L_4 \\
L_2 &: R_0^+ \to L_3 \\
L_3 &: R_3^+ \to L_1 \\
L_4 &: R_3^- \to L_5, L_0 \\
L_5 &: R_2^+ \to L_4 \\
L_6 &: \mathrm{HALT}
\end{aligned}
$$

Assuming the contents of registers R_0 and R_3 are initially zero, what function of the initial contents of registers R_1 and R_2 does this program compute in register R_0 upon halting? (You may find it helpful to consider the graphical representation of the program.) [4 marks]

(e) Let $f(x_1, x_2)$ be the partial function that is equal to $x_1 - x_2$ if $x_1 \geq x_2$ and is undefined otherwise. Give a register machine program that computes f. [5 marks]

2009 Paper 6 Question 4

Computation Theory

(*a*) Define what it means for a subset $S \subseteq \mathbb{N}$ to be a *recursively enumerable* set of numbers. [2 marks]

(*b*) Show that if S and S' are recursively enumerable sets of numbers, then so are the following sets (where $\langle x, y \rangle = 2^x(2y+1) - 1$).

(*i*) $S_1 = \{x \mid x \in S \text{ or } x \in S'\}$

(*ii*) $S_2 = \{\langle x, x' \rangle \mid x \in S \text{ and } x' \in S'\}$

(*iii*) $S_3 = \{x \mid \langle x, x' \rangle \in S \text{ for some } x' \in \mathbb{N}\}$

(*iv*) $S_4 = \{x \mid x \in S \text{ and } x \in S'\}$

Any standard results about partial recursive functions you use should be clearly stated, but need not be proved. [16 marks]

(*c*) Give an example of a subset $S \subseteq \mathbb{N}$ that is not recursively enumerable. [2 marks]

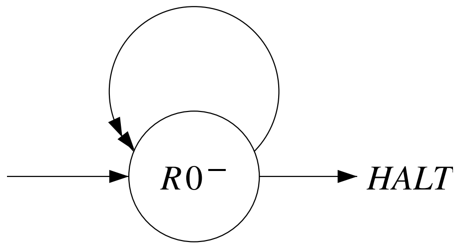
2009 Paper 6 Question 3

Computation Theory

(a) What is meant by a *state* (or *configuration*) of a register machine? [2 marks]

(b) A register machine program *Prog* is said to *loop at* $x \in \mathbb{N}$ if, when started with register $R1$ containing x and all other registers set to zero, the sequence of states *Prog* computes contains the same non-halted state at two different times.

(i) At which x does the following program loop?



[2 marks]

(ii) Show that if *Prog* loops at x, then the computation of *Prog* does not halt when started with register $R1$ containing x and all other registers set to zero. Is the converse true? [4 marks]

(iii) Consider the set $S = \{\langle e, x \rangle \mid Prog_e \text{ loops at } x\}$ of codes of pairs of numbers (e, x) such that the register machine program $Prog_e$ with index e loops at x. By adapting the usual proof of undecidability of the halting problem, or otherwise, show that S is an undecidable set of numbers. [Hint: if M were a register machine that decided membership of S, first consider replacing each *HALT* instruction (and each jump to a label with no instruction) with the program in part (i).] [12 marks]

2008 Paper 6 Question 10

Computation Theory

(*a*) Explain what is meant by each of the following statements:

(*i*) "c is a code for the total recursive function $f : \mathbb{N} \to \mathbb{N}$." [2 marks]

(*ii*) "F is a recursively enumerable set each of whose elements is a total recursive function $f : \mathbb{N} \to \mathbb{N}$." [3 marks]

(*b*) In each of the following cases state with reasons whether the set is recursively enumerable:

(*i*) the set A of all total recursive functions $a : \mathbb{N} \to \mathbb{N}$ such that $a(n+1) \geqslant a(n)$ for all $n \in \mathbb{N}$ [6 marks]

(*ii*) the set D of all total recursive functions $d : \mathbb{N} \to \mathbb{N}$ such that $d(n+1) \leqslant d(n)$ for all $n \in \mathbb{N}$ [9 marks]

2008 Paper 5 Question 10

Computation Theory

(a) *The Halting Problem for register machines is unsolvable.* State, without proof, a precise form of this result. [3 marks]

(b) Let the computation by program c on data d be represented by the natural number k that codes the pair (c, d). By considering the set $H(k)$ of the HALTing computations represented by codes $k' < k$, show that there is an increasing total function $h(k)$ which *grows too fast* to be computable. [6 marks]

(c) Given $h : \mathbb{N} \to \mathbb{N}$ with the above property

$$\begin{aligned} \text{let } \; f(k) &= h(k) + k \\ \text{and } \; g(x) &= \sup\{k : f(k) \leqslant x\}. \end{aligned}$$

Then $f : \mathbb{N} \to \mathbb{N}$ is strictly increasing, and $g : \mathbb{N} \to \mathbb{N}$ satisfies

$$g(f(k)) = k, \quad g(x) < k \quad \text{for all } x < f(k).$$

Show that g *grows too slowly* to be computable in the following sense:

given $G : \mathbb{N} \to \mathbb{N}$ such that

(i) $\{G(n) : n \in \mathbb{N}\}$ is unbounded

(ii) $G(n) \leqslant g(n)$ for all $n \in \mathbb{N}$

then $G(n)$ is *not* computable. [11 marks]

2007 Paper 4 Question 6

Computation Theory

(*a*) What does it mean for a set of natural numbers $S \subseteq \mathbb{N}$ to be

(*i*) *recursive*? [1 mark]

(*ii*) *recursively enumerable*? [2 marks]

(*b*) Show that if a set is recursive, then it is also recursively enumerable. [5 marks]

(*c*) Let ϕ_e denote the partial function from $\mathbb{N}$ to $\mathbb{N}$ computed by the register machine with code $e \in \mathbb{N}$. Is either of the following sets of numbers recursively enumerable? Justify your answer in each case, stating clearly any standard results that you use.

(*i*) $S_1 = \{e \in \mathbb{N} \mid \text{for all } x \in \mathbb{N},\ \phi_e(x) \text{ is defined}\}$. [6 marks]

(*ii*) $S_2 = \{e \in \mathbb{N} \mid \text{for some } x \in \mathbb{N},\ \phi_e(x) \text{ is defined}\}$. [6 marks]

2007 Paper 3 Question 7

Computation Theory

(*a*) (*i*) Define the notion of a *register machine* and the computations that it carries out. [5 marks]

(*ii*) Explain, in general terms, what is meant by a *universal* register machine. (You should make clear what scheme for coding programs as numbers you are using, but you are not required to describe a universal register machine program in detail.) [5 marks]

(*b*) (*i*) Explain what it means for a partial function f from $\mathbb{N}$ to $\mathbb{N}$ to be *computable* by a register machine. [2 marks]

(*ii*) Let $n > 1$ be a fixed natural number. Show that the partial function from $\mathbb{N}$ to $\mathbb{N}$

$$f_n(x) = \begin{cases} nx & \text{if } x > 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$$

is computable. [3 marks]

(*iii*) Explain why there are only countably many computable functions from $\mathbb{N}$ to $\mathbb{N}$. Deduce that there exists a partial function from $\mathbb{N}$ to $\mathbb{N}$ that is not computable. (Any standard results you use about countable and uncountable sets should be clearly stated, but need not be proved.) [3 marks]

(*iv*) If a partial function f from $\mathbb{N}$ to $\mathbb{N}$ is computable, how many different register machine programs are there that compute f? [2 marks]

2006 Paper 4 Question 9

Computation Theory

(*a*) Define the collection of *primitive recursive* functions. [6 marks]

(*b*) Why is a primitive recursive function always total? [1 mark]

(*c*) Show that the function m from $\mathbb{N}^2$ to $\mathbb{N}$ given by

$$m(x, y) = \begin{cases} x - y & \text{if } x \geq y \\ 0 & \text{if } x < y \end{cases}$$

is primitive recursive. [3 marks]

(*d*) Define the collection of *partial recursive* functions. [3 marks]

(*e*) What is meant by a *total recursive* function? [1 mark]

(*f*) Show that there exist total recursive functions that are not primitive recursive. Any standard results about register machines or recursive functions that you use need not be proved, but should be clearly stated. [6 marks]

2006 Paper 3 Question 7

Computation Theory

(*a*) (*i*) Give a graphical representation of the following register machine program.

$$\begin{array}{ll} L0: & Z^+ \to L1 \\ L1: & L^- \to L2, L3 \\ L2: & Z^+ \to L0 \\ L3: & Z^- \to L4, L5 \\ L4: & L^+ \to L3 \\ L5: & X^- \to L1, L6 \\ L6: & \text{HALT} \end{array}$$

[3 marks]

(*ii*) Assuming the contents of register Z is initially 0, when the program is run starting at instruction $L0$ what functions of the initial contents of registers X and L are computed in X and L when the machine halts? [5 marks]

(*b*) (*i*) What is meant by a *Turing machine*, its *configurations*, *transition relation* and the *computations* it carries out? What does it mean to say that a computation *halts*? [6 marks]

(*ii*) Given a Turing machine, is it decidable whether or not for all possible initial configurations the machine will not halt after 100 steps of transition? Justify your answer. [6 marks]

2005 Paper 4 Question 9

Computation Theory

(a) What does it mean for a subset S of the set $\mathbb{N}$ of natural numbers to be register machine *decidable*? [3 marks]

(b) For each $e \in \mathbb{N}$, let $\varphi_e \in \mathit{Pfn}(\mathbb{N}, \mathbb{N})$ denote the partial function computed by the register machine program with index e. Let $e_0 \in \mathbb{N}$ be an index for the totally undefined partial function (so that $\varphi_{e_0}(x)\uparrow$, for all $x \in \mathbb{N}$).

Suppose that a total function $f \in \mathit{Fun}(\mathbb{N}, \mathbb{N})$ is *extensional*, in the sense that for all $e, e' \in \mathbb{N}$, $f(e) = f(e')$ if φ_e and $\varphi_{e'}$ are equal partial functions. Suppose also that the set $S_f = \{x \in \mathbb{N} \mid f(x) = f(e_0)\}$ is not the whole of $\mathbb{N}$, so that for some $e_1 \in \mathbb{N}$, $f(e_1) \neq f(e_0)$.

(i) If membership of S_f were decided by a register machine M, show informally how to construct from M a register machine M' that, started with $R1 = e$ and $R2 = n$ (any $e, n \in \mathbb{N}$) always halts, with $R0 = 0$ if $\varphi_e(n)\downarrow$ and with $R0 = 1$ if $\varphi_e(n)\uparrow$. Make clear in your argument where you use the fact that f is extensional.

[Hint: For each $e, n \in \mathbb{N}$ consider the index $i(e, n) \in \mathbb{N}$ of the register machine that inputs x, computes $\varphi_e(n)$ and if that computation halts, then computes $\varphi_{e_1}(x)$.] [14 marks]

(ii) Deduce that if f is extensional, then S_f is either the whole of $\mathbb{N}$, or not decidable. [3 marks]

2005 Paper 3 Question 7

Computation Theory

(a) Explain informally, i.e. without reference to any particular model of computation, why the *Halting Problem* is undecidable. [6 marks]

(b) Briefly describe two mathematical problems, other than the Halting Problem, that are algorithmically undecidable. [4 marks]

(c) What does it mean for a partial function to be *register machine computable*? Show how the informal argument in part (a) can be turned into a rigorous proof that there is no register machine deciding the Halting Problem for register machine computable functions. [10 marks]

2004 Paper 4 Question 9

Computation Theory

(a) Explain what is meant by the following statements:

(i) $f : \mathbb{N} \to \mathbb{N}$ is a *total recursive* (TR) function; [3 marks]

(ii) the sequence $\{f_n : \mathbb{N} \to \mathbb{N}\}_{n \in \mathbb{N}}$ of TR functions of a single variable is recursively enumerable. [4 marks]

(b) Show that no recursive enumeration can include the set of *all* TR functions of a single variable. [4 marks]

(c) Suppose $u(n, x)$ is a recursive enumeration of the sequence of TR functions $f_n(x) = u(n, x)$. Show how to define a sequence $\{g_n : \mathbb{N} \to \mathbb{N}\}$ of TR functions of a single variable such that each g_n is distinct from every function f_n, and also from each g_k for $k \neq n$. [5 marks]

(d) Express the sequence $\{g_n\}$ as an explicit recursive enumeration $v(n, x) = g_n(x)$. [4 marks]

2004 Paper 3 Question 7

Computation Theory

(*a*) What is Turing's Thesis? [2 marks]

(*b*) Explain the action of a Turing machine that is specified by a quintuplet description. [4 marks]

(*c*) Define the *configuration* of a Turing machine at step t, and establish equations that specify the configuration of a k-symbol Turing machine at step $(t+1)$ in terms of the configuration at the previous step t. [6 marks]

(*d*) Explain how you would use your equations to simulate a specific Turing machine by a register machine whose program encodes the quintuplet description. To what extent does this support Turing's Thesis? [Explicit program for a register machine is *not* required.] [8 marks]

2003 Paper 4 Question 9

Computation Theory

What is the *Church–Turing Thesis*? Briefly describe some evidence that it is true. [4 marks]

Using the Church–Turing Thesis, or otherwise, show that if $f(x)$ and $g(x)$ are partial recursive functions of a single argument, then so are the following functions, where *dom(f)* denotes the set of integers x for which $f(x)$ is defined, and similarly for $dom(g)$.

$$h(x) = \begin{cases} x & \text{if } x \in dom(f) \text{ and } x \in dom(g) \\ \text{undefined} & \text{otherwise} \end{cases}$$ [4 marks]

$$k(x) = \begin{cases} x & \text{if } x \in dom(f) \text{ or } x \in dom(g) \\ \text{undefined} & \text{otherwise} \end{cases}$$ [6 marks]

Is the partial function defined by

$$f'(x) = \begin{cases} x & \text{if } x \notin dom(f) \\ \text{undefined} & \text{otherwise} \end{cases}$$

necessarily partial recursive if f is? Justify your answer. [6 marks]

2003 Paper 3 Question 7

Computation Theory

What is meant by a *register machine*? Explain the action of a register machine program. [6 marks]

What does it mean for a partial function $f(x_1, \ldots, x_n)$ of n arguments to be *register machine computable*? [3 marks]

Design register machines to compute the following functions.

$$f(x_1, x_2) = x_1 + x_2$$ [2 marks]

$$g(x_1) = \begin{cases} 42 & \text{if } x_1 > 0 \\ \text{undefined} & \text{otherwise} \end{cases}$$ [2 marks]

$$h(x_1) = 2^{x_1}$$ [4 marks]

Give an example of a function that is not register machine computable, stating clearly any well-known results you use. [3 marks]

2002 Paper 4 Question 6

Computation Theory

(*a*) Your mathematician friend can prove to you that there are uncountably many functions from numbers to numbers, but does not know any computation theory. Explain to her what is meant by a *partial recursive* function and by a (total) *recursive* function. How would you convince her that there must exist functions that are not recursive? [12 marks]

(*b*) What does it mean for a set of numbers $S \subseteq \mathbb{N}$ to be

(*i*) *decidable*; [1 mark]

(*ii*) *recursively enumerable* (r.e.)? [2 marks]

(*c*) S is called *co-r.e.* if its complement $\{x \in \mathbb{N} \mid x \notin S\}$ is r.e. Show that S is decidable if it is both r.e. and co-r.e. (Any standard results about computable functions that you use should be clearly stated.) [5 marks]

2002 Paper 3 Question 6

Computation Theory

(a) Explain how each number $e \in \mathbb{N}$ can be decoded uniquely as a register machine program $Prog_e$. [6 marks]

(b) What would it mean for a register machine to *decide the halting problem*? [4 marks]

(c) Prove that such a register machine cannot exist. (You may assume the existence of suitable register machines for copying registers and manipulating lists of numbers so long as you specify their behaviour clearly.) [10 marks]

2001 Paper 4 Question 8

Computation Theory

(a) Define precisely what is meant by the following:

(i) $f(x_1, x_2, \ldots x_n)$ is a Primitive Recursive (PR) function of arity n. [5 marks]

(ii) $f(x_1, x_2, \ldots x_n)$ is a Total Recursive (TR) function of arity n. [3 marks]

(b) Ackermann's function is defined by the following recursive scheme:

$$\begin{aligned} f(0, y) &= S(y) = y + 1 \\ f(x+1, 0) &= f(x, 1) \\ f(x+1, y+1) &= f(x, f(x+1, y)) \end{aligned}$$

For fixed n define

$$g_n(y) = f(n, y).$$

Show that for all n, $y \in \mathbb{N}$,

$$g_{n+1}(y) = {g_n}^{(y+1)}(1),$$

where $h^{(k)}(z)$ is the result of k repeated applications of the function h to initial argument z. [4 marks]

(c) Hence or otherwise show that for all $n \in \mathbb{N}$, $g_n(y)$ is a PR function. [4 marks]

(d) Deduce that Ackermann's function $f(x, y)$ is a TR function. [3 marks]

(e) Is Ackermann's function PR? [1 mark]

2001 Paper 3 Question 9

Computation Theory

(*a*) Describe the action of a Turing machine. [4 marks]

(*b*) Define what is meant by a configuration of an N-state, k-symbol Turing machine. [2 marks]

(*c*) Explain briefly how to enumerate all possible Turing machine computations, so that a given computation can be characterised by a single natural number code c. [5 marks]

(*d*) Show that it is not possible to compute the maximum distance travelled by the Turing machine head from its initial position during halting computations as a function of the code c. Any results that you use should be stated clearly. [9 marks]

2000 Paper 4 Question 8

Computation Theory

Let $\mathbb{N}$ be the natural numbers $\{0, 1, 2 \ldots\}$.

What is meant by each of the following statements?

- The subset $S \subseteq \mathbb{N}$ is *recursive*.
- The subset $S \subseteq \mathbb{N}$ is *recursively enumerable*.

[5 marks]

How would you extend the definition of *recursive enumeration* to sets of computable functions? [3 marks]

A sequence of natural numbers is a total function $s : \mathbb{N} \to \mathbb{N}$. The sequence is *recursive* if and only if s is computable.

A finite sequence σ of natural numbers is specified by a pair (l, x), where $l \in \mathbb{N}$ is the number of elements, and $x : [1, l] \to \mathbb{N}$ is a function that defines those elements. The case $l = 0$ defines the null sequence.

In each of the following cases, establish whether the set defined is recursively enumerable:

(*a*) the set of all recursive subsets of $\mathbb{N}$ [5 marks]

(*b*) the set of all recursive sequences of natural numbers [2 marks]

(*c*) the set of all finite sequences of natural numbers [5 marks]

2000 Paper 3 Question 9

Computation Theory

One of the most important contributions of the theory of computation has been to establish that the halting problem is not decidable. Give a clear statement of this result (you are not asked to prove it). [5 marks]

Define a *configuration* of a 2-register machine at a particular point during the execution of some program. [3 marks]

By considering the total number of configurations or otherwise, show that it is not possible to compute an upper bound for the contents of the two registers during halting computations as a function of the program code and the initial contents of the two registers. [12 marks]

1999 Paper 4 Question 1

Computation Theory

Define the *primitive recursive* and *partial* ($\mu-$) *recursive* functions. [6 marks]

Suppose you are given a Turing machine with state set Q and k-symbol alphabet S whose action is defined by transition functions

$$
\begin{aligned}
q' = f(q,s) &\in Q \uplus \{H\} && \text{(disjoint union)} \\
s' = r(q,s) &\in S && \text{(replacement symbol)} \\
d' = d(q,s) &\in \{L,R,C\} && \text{(movement)}
\end{aligned}
$$

where the head moves to L or R on the tape unless $q' = H$, in which case $d' = C$ and the machine stops.

Extend the action of the machine by an additional state symbol D so that for all $s \in S$,

$$
\begin{aligned}
f(H,s) &= f(D,s) = D \\
r(H,s) &= r(D,s) = s \\
d(H,s) &= d(D,s) = C
\end{aligned}
$$

Show that the action of the Turing machine as extended in this way can be described by a primitive recursive function $T(t,x)$, where t is a step counter and x is a code specifying the initial configuration. [10 marks]

Hence show that computation by any Turing machine may be represented by a partial recursive function. [4 marks]

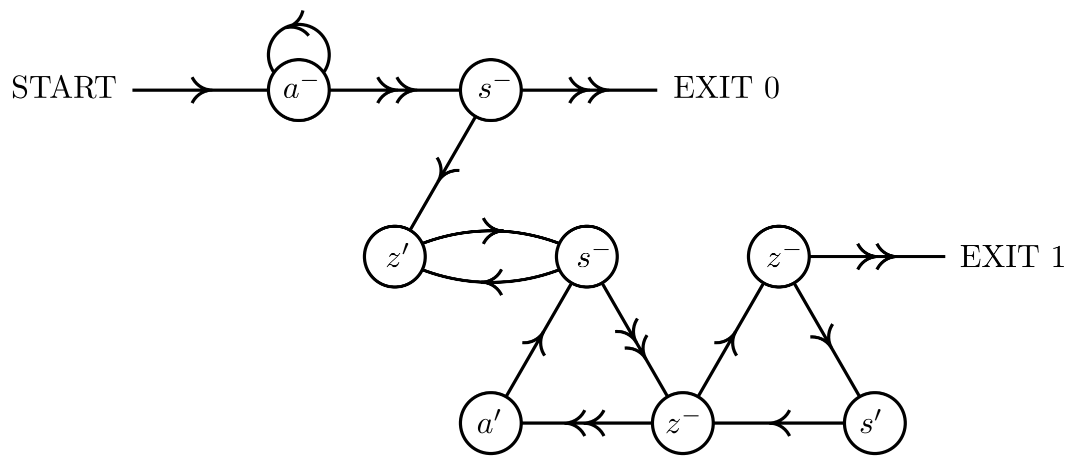
1999 Paper 3 Question 9

Computation Theory

Define computation by a register machine, explaining the action of the program. [4 marks]

What is meant by the *current configuration* during a register machine computation? [2 marks]

In the following program, assume that register Z holds 0 initially. What is its effect?



[2 marks]

Show how to encode a general register machine program and the initial configuration of one of its computations into a pair of natural numbers. [6 marks]

Outline the design of a register machine that simulates a general register machine computation specified by a single natural number. Your machine should take appropriate action for all possible inputs. [6 marks]

1998 Paper 4 Question 8

Computation Theory

Explain *Church's Thesis*, making clear its connection with computability.

[3 marks]

Define precisely what is meant by the set of all *Primitive Recursive* (*PR*) functions.

[4 marks]

Outline steps that would enable you to recursively enumerate the set of all PR functions, showing how to determine the arity of each function generated (little detail is required).

[7 marks]

Suppose that $V(n,x)$ is a recursive enumeration of all the PR functions of arity 1. By considering the function $v(x) = S\big(V(x,x)\big)$ or otherwise, show that

(a) the enumerating function $V(n,x)$ cannot itself be Primitive Recursive;

[4 marks]

(b) there are Total Recursive functions that are not Primitive Recursive.

[2 marks]

1998 Paper 3 Question 9

Computation Theory

What is meant by saying that a model for computation offers unlimited data storage but is restricted to finite logic? [3 marks]

How would you record the *configuration* during computation within such a model? Illustrate your answer by considering both Turing machines and register machines. [5 marks]

Suppose you are given a k-symbol Turing machine having searching states. Show how to represent the transition from the configuration at time t to the configuration at time $t + 1$ by a system of arithmetic equations. Hence show that any Turing machine computation may be simulated by a register machine having a suitable program. [12 marks]

1997 Paper 4 Question 8

Computation Theory

Define what is meant by saying that a set of *partial recursive* (μR) functions is *recursively enumerable*. Explain briefly how the universal register machine might be used to define a universal μR function $\mu(e, x)$ that enumerates the set of *all* partial recursive functions of a single variable x. [6 marks]

(a) Prove that the set of all total recursive functions of a single variable is not recursively enumerable. [4 marks]

(b) Show that there are recursively enumerable sets that are not recursive. [6 marks]

(c) Show that there is a partial recursive function that cannot be extended to any total recursive function. [4 marks]

[Any properties of recursively enumerable sets that you assume should be clearly stated.]

1997 Paper 3 Question 9

Computation Theory

Explain the action of a Turing machine, and show how the progress of a computation may be tracked by maintaining a record of the configuration at each time t. Prove that a computation which enters the same configuration twice will not terminate.

[8 marks]

Suppose you are given a Turing machine T having r states and k symbols. It is known that in a particular computation the head moves on the tape so that it is never more than l squares from its starting point. Calculate a bound on the number of configurations that the machine may enter during the computation. [4 marks]

State a precise form of the unsolvability of the HALTing problem for Turing machines. Assuming this result, show that it is not possible to compute a bound on the distance of the head from its starting position during HALTing Turing machine computations. [8 marks]

[You may assume that a Turing machine computation may be characterised by codes q and d that specify the quintuplet description and initial tape contents uniquely.]

1996 Paper 4 Question 8

Computation Theory

A *bag* B of natural numbers is a total function $f_B : \mathbb{N} \to \mathbb{N}$ giving for each natural number x the count $f_B(x)$ of occurrences of x in B. If each $f_B(x) = 0$ or 1, then f_B is the characteristic function χ_s of a set S: every set can thus be regarded as a bag.

(a) A bag B is *recursive* if the function f_B is computable. Suppose that the sequence of bags $\{B_n \mid n \in \mathbb{N}\}$ is recursively enumerated by the computable function $e(n, x) = f_n(x)$, which gives the count of x in each bag B_n. Show that there is a recursive set S that is different from each bag B_n. [7 marks]

Hence prove that the set of all recursive bags cannot be recursively enumerated. [3 marks]

(b) A bag B is *finite* if there is $X \in \mathbb{N}$ such that $f_B(x) = 0$ for all $x \geqslant X$. Show that the set of all finite bags is recursively enumerable. [10 marks]

1996 Paper 3 Question 9

Computation Theory

Show how to code the program and initial data for an n-register machine into natural numbers p and d. In what sense do the codes p and d determine a unique computation? [9 marks]

Using your codes establish a precise statement of the Halting Problem for Register Machines. [3 marks]

Assume that the Halting Problem is in general undecidable. Prove that it cannot be decided whether a general program p terminates when the initial data is zero in every register. [8 marks]

1995 Paper 4 Question 9

Computation Theory

Explain what is meant by a *primitive recursive* function and by a *partial recursive* function. [6 marks]

Show that the function giving the next state of a register machine in terms of the current state is primitive recursive. (You may assume the existence of primitive recursive functions for coding any n-element list of numbers $(x_1, \ldots, x_n)$ as a number $[x_1, \ldots, x_n]$ (for each n), and for extracting the head x_1 and the (coded) tail $[x_2, \ldots, x_n]$ from such a coded list.) [8 marks]

Deduce that every register machine computable partial function is partial recursive. [5 marks]

Is the converse true? [1 mark]

1994 Paper 6 Question 11

Computation Theory

Explain *Turing's Thesis.* [5 marks]

(*a*) What is meant by saying that a Turing machine has *searching* states? Show that any Turing machine computation can be effected by a machine with searching states, equivalent in the sense that the head movements are identical and the same symbols are written to the tape. [5 marks]

(*b*) Show that, subject to suitable encoding, any computation can be carried out by a Turing machine having only two states. [10 marks]

1994 Paper 5 Question 11

Computation Theory

The Halting Problem for register machines is unsolvable. State, without proof, a precise form of this result. [3 marks]

Let the computation by program p on data d be represented by the natural number k that codes the pair (p, d). By considering the set $H(k)$ of the HALTing computations represented by codes $k' \leqslant k$, show that there is an increasing total function $h(k)$ which *grows too fast* to be computable. [6 marks]

Given $h : \mathbb{N} \to \mathbb{N}$ with the above property

$$\begin{aligned} \text{let} \quad f(k) &= h(k) + k \\ \text{and} \quad g(x) &= \sup\{k : f(k) \leqslant x\}. \end{aligned}$$

Then $f : \mathbb{N} \to \mathbb{N}$ is strictly increasing, and $g : \mathbb{N} \to \mathbb{N}$ satisfies

$$g(f(k)) = k, \quad g(x) < k \quad \text{for all } x < f(k).$$

Show that g *grows too slowly* to be computable in the following sense...
given $G : \mathbb{N} \to \mathbb{N}$ such that

(a) $\{G(n) : n \in \mathbb{N}\}$ is unbounded

(b) $G(n) \leqslant g(n)$ for all $n \in \mathbb{N}$

then $G(n)$ is *not* computable. [11 marks]

1993 Paper 6 Question 10

Computation Theory

Explain what is meant by the following:

> 'F is a recursively enumerable set each of whose elements is a total recursive function $f : \mathbb{N} \to \mathbb{N}$.' [3 marks]

In each of the following cases state with reasons whether the set is recursively enumerable:

(*a*) the set A of all total recursive functions $a : \mathbb{N} \to \mathbb{N}$ such that $a(n+1) \geqslant a(n)$ for all $n \in \mathbb{N}$ [7 marks]

(*b*) the set D of all total recursive functions $d : \mathbb{N} \to \mathbb{N}$ such that $d(n+1) \leqslant d(n)$ for all $n \in \mathbb{N}$ [10 marks]

1993 Paper 5 Question 10

Computation Theory

Show that there is no way of deciding by algorithms whether a general register machine program with code p will terminate when started with initial data of 0 in every register. [10 marks]

Show that there is no way of deciding by algorithm whether the blank character will be printed during the course of a general Turing machine computation. [10 marks]

Note: any standard form of the undecidability result for the general halting problem may be assumed, but should be stated clearly.