T–79.1001 **Autumn 2006**
Introduction to Theoretical Computer Science T
Tutorial 11, 4 to 5 December (sic!)
Problems

Homework problems:

1. As you know, failure of numerical operations (division by zero, register over- or underflow) can result in the execution of a computer program terminating in a run-time error. Explain why this possibility cannot be eliminated in advance, e.g. by testing for the risk of division by zero before initiating the execution of a program. (*Hint:* Interpret the task as a variant of the halting problem.)

2. Consider application programs running under some given operating system. Let us say that a program P is *dangerous*, if it on some input modifies the operating system's system files. A *general purpose virus tester* is a program that receives as input an arbitrary application program text P, analyses it and returns output "DANGER", if the program is dangerous, and "OK" otherwise. Show that if any dangerous programs exist at all, then general-purpose virus testing is impossible. (*Hint:* Interpret Rice's theorem in the given setting.)

3. You are being offered the following programming assignment:

   Intel Septium code optimisation
   A large company producing embedded systems software would like to have a code optimiser that, given any machine language program for the Intel Septium processor, outputs the smallest machine language program that is functionally equivalent to the given one (i.e. that has the same input-output behaviour).

   Your comments? Under what conditions would you accept? Justify your answer.

Demonstration problems:

4. Prove, without using Rice's theorem, that the following problem is undecidable:

   Given a Turing machine M; does M accept the empty string?

5. Prove the following connections between recursive functions and languages:

   (i) A language $A \subseteq \Sigma^*$ is recursive ("Turing-decidable"), if and only if its characteristic function

   $$\chi_A : \Sigma^* \to \{0,1\}, \qquad \chi_A(x) = \begin{cases} 1, & \text{if } x \in A; \\ 0, & \text{if } x \notin A \end{cases}$$

   is a recursive ("Turing-computable") function.

   (ii) A language $A \subseteq \Sigma^*$ is recursively enumerable ("semidecidable", "Turing-recognisable"), if and only if either $A = \emptyset$ or there exists a recursive function $g : \{0,1\}^* \to \Sigma^*$ such that

   $$A = \{g(x) \mid x \in \{0,1\}^*\}.$$