CS-226 Computability Theory

(*Attempt 2 questions out of 3*)

Question 1.

(a) (i) What does it mean for a set to be *countable*?

(ii) Give two characterisations of countability, one using the notion of an *injective function*, the other using the notion of a *surjective function*.

[5 marks]

(b) (i) Show that $\mathbb{Q}$, the set of rational numbers, is countable.
You may use any of the characterisations of countability you gave in question 1 (a) (ii).

(ii) Show that $\mathbb{R}$, the set of real numbers, is uncountable.

(iii) Is the set of irrational numbers countable? Justify your answer.

(iv) Which of the following sets are countable? Justify your answers.

- $\{ f \mid f : \mathbb{N} \to \mathbb{Q} \}$
- $\{ f \mid f : \mathbb{N} \to \{0, 1\}, f \text{ computable} \}$

[12 marks]

(c) (i) State *Kleene's normal form theorem*.

(ii) What is a *universal Turing Machine*?

(iii) Sketch a proof that a universal Turing Machine exists.

Hint for (iii): Use Kleene's normal form theorem and the equivalence of different models of computation.

[8 marks]

Question 2.

(a) (i) What does it mean for a function $f : \mathbb{N}^k \to \mathbb{N}$ to be *URM-computable*?

(ii) Write a URM-program, only using the basic instructions `succ`, `pred`, `ifzero`, that computes

$$\text{double} : \mathbb{N} \to \mathbb{N}, \quad \text{double}(x) := 2x.$$

[8 marks]

(b) (i) What does it mean for a function $f : \mathbb{N}^k \to \mathbb{N}$ to be *Turing-computable*?

(ii) Construct a Turing Machine over the alphabet $\{0, 1, \llcorner\lrcorner\}$ that computes the same function as in question 2 (a) (ii), that is,

$$\text{double} : \mathbb{N} \to \mathbb{N}, \quad \text{double}(x) := 2x.$$

You may assume that the input is written in binary on the tape and that the head points to the first digit of the input.

[8 marks]

(c) Let $f : \mathbb{N}^2 \to \mathbb{N}$ be defined as follows:

$$f(e, n) = \begin{cases} g(n) & \text{if } e \text{ is the code of a unary primitive recursive function } g \\ 0 & \text{otherwise} \end{cases}$$

The function f is intuitively computable and therefore, by the Church-Turing Thesis, partial recursive.

Show that f is not primitive recursive.

[5 marks]

(d) Which are the three models of computation we studied in the course? Briefly explain –from the perspective of Computability theory– what is the advantage of these models over real world programming languages such as Java or Haskell.

[4 marks]

Question 3.

(a) Explain the differences between *primitive recursive*, *recursive* and *partial recursive* functions.

Give suitable examples which underline these differences.

[5 marks]

(b) Show that the following functions are primitive recursive:

(i) $f : \mathbb{N}^2 \to \mathbb{N}, \quad f(x, n) := x^n$

(ii) $g : \mathbb{N} \to \mathbb{N}, \quad g(x) := x^x$

(iii) $h : \mathbb{N} \to \mathbb{N}, \quad h(x) := \lfloor \sqrt{x} \rfloor$

Remark: It is assumed that $x^0 = 1$ and, for a real number y, $\lfloor y \rfloor$ is defined as the largest natural number less or equal to y.

[9 marks]

(c) (i) What does it mean for a problem X to be *reducible* to a problem Y?

(ii) Assume that X is an undecidable problem and that X is reducible to a problem Y. What can you say about the decidability of problem Y.

Justify your answer.

[5 marks]

(d) (i) What is the *Halting Problem*?

(ii) Sketch a proof of the *undecidability* of the Halting Problem.

[6 marks]