This problem sheet is about computational complexity [1].

1. Determine the number of comparisons made by Bubble sort on the following inputs.
   (a) $[3, 2, 1]$
   (b) $[4, 3, 2, 1]$
   (c) $[5, 4, 3, 2, 1]$
   (d) $[6, 5, 4, 3, 2, 1]$
   (e) $[20, 19, 18, \ldots, 3, 2, 1]$
   (f) $[3, 4, 5, 2, 1]$
   (g) $[4, 5, 1, 2, 3]$

Solution:

(a) $2 + 1 = 3$

(b) $3 + 2 + 1 = 6$

(c) $4 + 3 + 2 + 1 = 10$

(d) $5 + 4 + 3 + 2 + 1 = 15$

(e) $6 + 4 + 3 + 2 + 1 = 21$

(f) $(20 \times 19) \div 2 = 190$

(g) $4 + 3 + 2 + 1 = 10$

(h) $5 + 4 + 3 = 12$, using the fact that if no swaps are made in one pass then the list is sorted.

2. Classify the following as polynomial, exponential, or logarithmic expressions.
   (a) $3n + 1$
   (b) $n^2 + 2n + 1$
   (c) $log_b(a)$
   (d) 10^n
   (e) $2^n + n^2$
   (f) $nlog_n$
   (g) n^n

Solution:

(a) Polynomial (linear).

(b) Polynomial.

(c) Logarithmic.

(d) Exponential.

(e) Ambiguous.

(f) Ambiguous.

(g) Ambiguous.

(h) Exponential.

3. Explain what the P computational complexity class is, and give an example of a problem known to be in P.

4. Explain what PRIMES is.

Solution:

$$PRIMES = \{2, 3, 5, 7, 11, \ldots\}$$

PRIMES is the set of all prime numbers, and is a subset of the natural numbers.

5. Describe two different algorithms the check if a number is a prime. The algorithms should accept a single positive integer as input, and output true if the number is prime and false otherwise.

6. Determine which of the following are in PRIMES (without Google).
   (a) 2
   (b) 3
   (c) 4
   (d) 10
   (e) 11
   (f) 13,109
   (g) 100,827
   (h) 102,203

Solution:

(a) Yes

(b) Yes

(c) No

(d) No

(e) Yes

(f) Yes

(g) No

(h) Yes

7. Explain what a decision problem is, and how decision problems relate to Turing machines.

Solution: As in notes.

8. Explain the decision problem related to PRIMES.
9. Explain the concept of complexity in terms of Turing machines.
10. Explain why if we can solve decision problem A in polynomial time, and we can convert decision problem B to problem A in polynomial time, then we can solve problem B in polynomial time too.

References

[1] Michael Sipser. *Introduction to the Theory of Computation.* International Thomson Publishing, 3rd edition, 1996.