The University Of Sheffield.

Data Provided:
Master Theorem (page 4)

DEPARTMENT OF COMPUTER SCIENCE **Mock Exam**

ADVANCED PROGRAMMING TOPICS **2 hours**

Answer BOTH questions.

Both questions carry equal weight. Figures in square brackets indicate the percentage of available marks allocated to each part of a question.

1. a)

Put the asymptotic complexity hierarchy

$$O(n \log n) \subset O(n) \subset O(n^2) \subset O(2^n) \subset O(1) \subset O(\log n)$$

into the correct order [10%]

b)

Is $\Theta(n^2) \subset \Theta(2^n)$? Explain your answer. [20%]

c)

Determine an appropriate asymptotic complexity for an algorithm with the following running time (in seconds). Explain your answer. How would this algorithm perform on large inputs?

$$\sum_{i=1}^{n}(7+3i)$$

[30%]

d)

Apply the Master Theorem to determine the asymptotic complexities of the algorithm whose running time is given by the following recurrence.

$$T(n) = 2T\left(\frac{n}{2}\right) + n \log_2 n$$

If the Master Theorem is not applicable, state why. [40%]

2. a)

Explain the difference between *defined* and *undefined* values of recursive functions. [10%]

b)

Explain what properties make a problem amenable to solution with a dynamic programming algorithm. [20%]

c)

For the following implementation of a set Abstract Data Type prove that the axiom `isEmpty (addMember x s) == False` is satisfied.

```
module Set (emptySet, isEmpty, isMember, addMember, removeMember, union) where

newtype Set a = MySet [a] deriving (Show)

emptySet :: Set a
emptySet = MySet []                                  --(empty)

isEmpty :: Set a -> Bool
isEmpty (MySet []) = True                            --(isEmpty.1)
isEmpty _ = False                                    --(isEmpty.2)

addMember :: Ord a => a -> Set a -> Set a
addMember x (MySet ys) = MySet (addMemberToList x ys) --(addMember.1)

addMemberToList :: Ord a => a -> [a] -> [a]
addMemberToList x [] = [x]                           --(addMember.2)
addMemberToList x (y:ys)
  | x < y      = (x:(y:ys))                          --(addMember.3)
  | x == y     = (y:ys)                              --(addMember.4)
  | otherwise  = (y:(addMemberToList x ys))          --(addMember.5)
```

[30%]

d)

Using structural induction, prove that `reverse (xs ++ ys) == reverse ys ++ reverse xs`; you may assume associativity of the ++ operator.

```
[] ++ ys = ys                                         --(++1)
(x:xs) ++ ys = x:(xs ++ ys)                           --(++2)

reverse [] = []                                  --(reverse1)
reverse (x:xs) = reverse xs ++ [x]               --(reverse2)
```

[40%]

END OF QUESTIONS

Master Theorem

Theorem (Master Theorem): Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function and let $T(n)$ be defined for $n \geq 0$ by the recurrence

$$T(n) = aT(n/b) + f(n),$$

where n/b means either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ can be bounded asympotically according to the following cases:

1. If $f(n) \in O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) \in \Theta(n^{\log_b a})$.
2. If $f(n) \in \Theta(n^{\log_b a})$, then $T(n) \in \Theta(n^{\log_b a} \log n)$.
3. If $f(n) \in \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) \in \Theta(f(n))$.