1. The following higher order function represents the n-th composition of a function \( f \) with itself. \( \text{(2 + 8 marks)} \)

\[
\text{let rec selfcomp } f \ n = \text{if } n = 0 \text{ then (function } x \to x) \text{ else function } x \to f \left( \text{selfcomp } f \ (n-1) \ x \right) ;;
\]

(i) What is the type of \( \text{selfcomp} \)?

\[ (\alpha \to \alpha) \to (\text{int} \to (\alpha \to \alpha)) \]

which is displayed as \( ('a -> 'a) -> \text{int} -> 'a -> 'a \).

(ii) How would you use \( \text{selfcomp} \) (i.e., specify the appropriate parameters) to generate the 10th term of the arithmetic sequence

\[ 1, 4, 7, 10, \ldots 1 + 3(i - 1) \]

\[
\text{selfcomp (function } x \to x+3) 9 1
\]

starting with 1, apply the “add 3” function 9 (=10-1) times.

and the 5th term of the geometric series

\[ 2, 6, 18, 54, \ldots 2 \cdot 3^{i-1} \]

\[
\text{selfcomp (function } x \to x*3) 4 2
\]

starting with 2 apply the “multiply by 3” function 4 (=5-1) times.

2. A point \((x, y)\) is dominated by \((x', y')\) if \(x \leq x'\) and \(y \leq y'\). Given a set \(S\) of \(n\) points \((x_i, y_i)\), a point \(p \in S\) is maximal if it is not dominated by any point in \(S\). For example in the set \(\{(3, 4), (2, 1), (4, 0)\}\), points \(3, 4\) and \(4, 0\) are maximal. Complete the program below that computes all the maximal points of a given list \(1\) of points. The input list is given in non-increasing order of x coordinates. The algorithm scans the points of \(1\) and determines if the current point is maximal based on the y coordinates of all the points scanned before. The parameter \(\text{max}\) keeps track of the largest y coordinate and \(\text{last}\) keeps track of the x-coordinate of the previous point. Note that the point with the largest x coordinate is always maximal. For simplicity you may assume that all points are distinct. \(\text{(10+5 marks)}\)

\[
\text{let maximal } l = \text{let rec prefixmax } lst \text{ last max = (*l is reverse sorted on 1st component *)}
\]

\[
\begin{align*}
\text{match } lst \text{ with} \\
[] &\rightarrow [] |
\end{align*}
\]
Rewrite the function `prefixmax` as a tail recursive function with minimal modifications.

```ocaml
let rec prefixmax lst last max output =
  match lst with
  | [] -> output |
  | (a,b)::tail ->
    if a > last then raise Not_sorted
    else if b > max then prefixmax tail a b output @ [(a,b)]
    else prefixmax tail a max output ;;
```

The most common mistake was failure to append `(a,b)` when it was a maximal point, i.e., it won’t produce any output in part 1.
In part 2, you needed an extra parameter to create an output list - without that it won’t be tail recursive.

3. To compute all the prime numbers in `[2, n]`, the sieve of Eratosthenes proceeds by eliminating all factors of 2, then all factors of 3 etc. Initially an array of size `n+1` is initialised to 0. In the i-th iteration, all the multiples of `p_i` (the i-th prime where `p_1 = 2`) are marked as 1 (not prime). In the end, `i` is prime if `a[i] == 0` (`==` is the Java comparator operator for equality testing). (8+4+3 marks)

(i) Complete the following program.

```java
for (i = 2; i <= n; i++) /* 2 is prime */
{
  if (a[i] == 0)
  {
    j = _2__;
    while ( j <= _n/i_)
    {
      a[_i*j_] = 1;
      j++ ; // Invariant of while: ___________
      a[2i], a[3i] ....a[ji] is set to 1
      (a[i] is 0 since i is prime !)
    } //end while
  } //end of if
} //end of for
```

(ii) Complete the following program.

```java
for ( i = 2; i <= n ; i++) { if (a[i] == 0) System.out.println( i); }
```
Note that starting from \( j = i \) is also correct but \( j = i+1 \) is not correct since \( i \times i \) is not prime and must be marked as 1. Since \( i \) is the smallest prime factor it will not be marked earlier. With \( j = i \), the proof of part (b) must address it. (ii) In the line marked ***, what property can you claim about \( i \) and why?

The number \( i \) is prime. Inductively assume that all prime numbers have been correctly identified (before \( i \)). Clearly \( i \) is not a multiple of any prime number \( j, j < i \). Therefore \( i \) must be prime. Note that we are only marking the multiple of prime numbers as 1 and not the multiple of all numbers.

(iii) What is the number of basic operations as a function of \( n \)? Each non-prime number \( i \) is crossed out at most the number of prime factors. The right expression is \( n/2 + n/3 + n/5 + \ldots + n/p_i \ldots \) where \( p_i \) is the \( i \)-th prime. This is bounded by \( \sum_{i=2}^{n}(n/i) \leq n(1/2 + 1/3 + \ldots + 1/n) \leq n \cdot H_n \) where \( H_n \) is the \( n \)-th Harmonic and is roughly \( \log_e n \).

1 mark was deducted if closed form solution was not given. A tighter answer (hard to prove) is that the sum is bounded by \( n \log \log n \).