THEORY PROBLEMS FOR DATA STRUCTURES AND ALGORITHMS (COMP1009)

1 Consider the program (valid for inputs $n \geq 1$)

```
foo(int n) {
   bar();
   if (n==1)
      return;
   foo(n-1);
   foo(n-1);
   foo(n-1);
}
```

(a) Let $C(n)$ be the number of times the function `bar()` is called when we call `foo(n)`. Write down a recurrence relation for $C(n)$. *(2 marks)*

$$C(n) = 3C(n-1) + 1$$

(b) Write down the boundary condition for the recurrence relation. *(1 marks)*

$$C(1) = 1$$

(c) Using the recurrence relation to compute $C(2)$, $C(3)$ and $C(4)$. *(3 marks)*

$$C(2) = 3 \times 1 + 1 = 4$$

$$C(3) = 3 \times 4 + 1 = 13$$

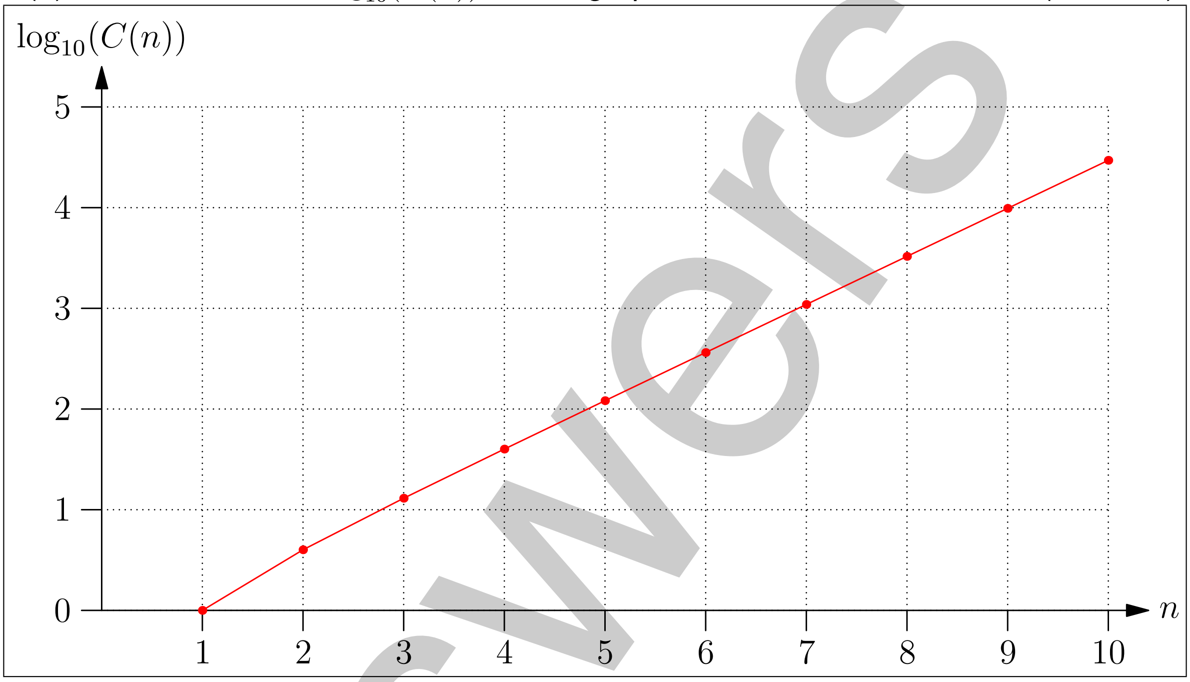
$$C(4) = 3 \times 13 + 1 = 40$$

(d) Prove by induction that $f(n) = \frac{3^n-1}{2}$ satisfies the recurrence relation for $C(n)$. *(5 marks)*

In the base case we have $f(1) = (3^1 - 1)/2 = 1 = C(1)$**.**
In the inductive step we assume that $C(n-1) = f(n-1) = (3^{n-1} - 1)/2$ **then**

$$\begin{aligned} C(n) &= 3\left(\frac{3^{n-1}-1}{2}\right) + 1 \\ &= \frac{3^n - 3 + 2}{2} = \frac{3^n - 1}{2} = f(n). \end{aligned}$$

(e) Sketch the curve $\log_{10}(C(n))$ on the graph below. *(2 marks)*



(f) Assume that the most time consuming operation is calling function `bar()` then, if it takes 100s to compute `foo(5)` approximately how long will it take to compute `foo(10)`? *(2 marks)*

We assume that the time to run `foo(n)` is $T(n) \approx c\,3^n$, where c is some constant we have to estimate empirically. Using the fact that running `foo(5)` takes 100s, i.e. $T(5) = 100 \approx c \times 3^5$ so that $c \approx 100 \times 3^{-5}$. The time to run `foo(10)` is $T(10) \approx 100 \times 3^{-5} \times 3^{10} = 3^5 \times 100s = 24\,300s$.

---

2 Consider the program (valid for inputs $n \geq 1$)

```
foo(int n) {
   for(i=1; i<=2n-1; i++)
      bar();
   if (n==1)
      return;
   foo(n-1);
}
```

(a) Let $C(n)$ be the number of times the function `bar()` is called when we call `foo(n)`. Write down a recurrence relation for $C(n)$. *(2 marks)*

$C(n) = C(n-1) + 2\,n - 1$

(b) Write down the boundary condition for the recurrence relation. *(1 marks)*

$C(1) = 1$

(c) Using the recurrence relation to compute $C(2)$, $C(3)$ and $C(4)$. *(3 marks)*

$C(2) = 1 + 4 - 1 = 4$

$C(3) = 4 + 6 - 1 = 9$

$C(4) = 9 + 8 - 1 = 16$

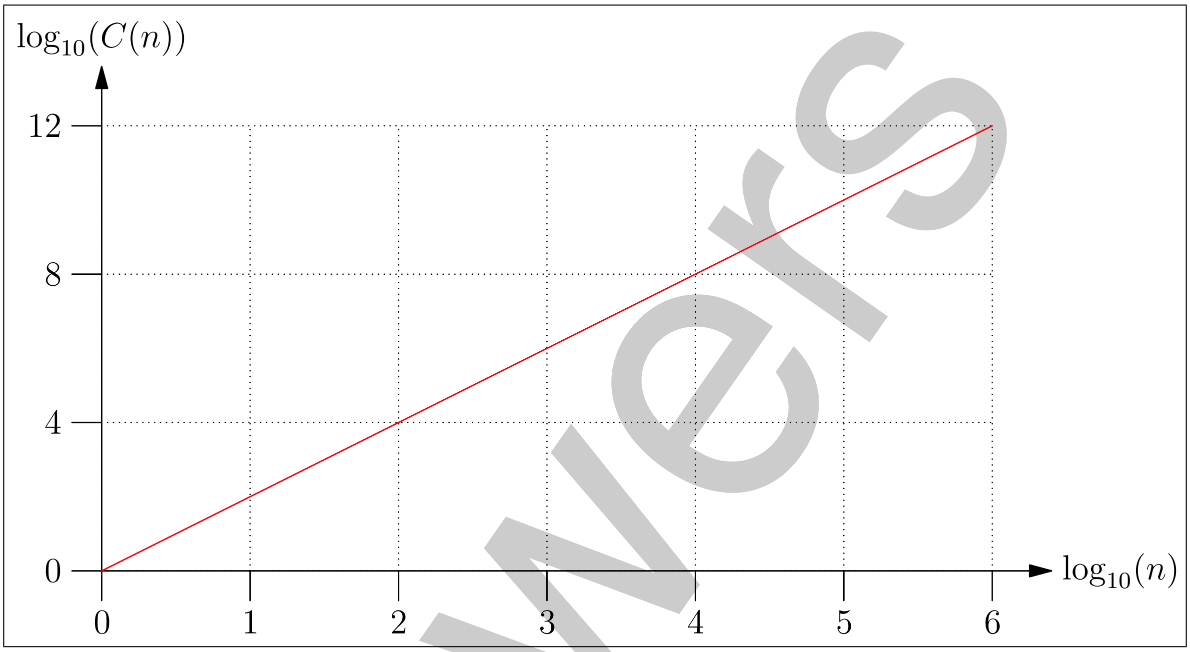
(d) Guess the solution for $C(n)$ and prove by induction that it satisfies the recurrence relation for $C(n)$. *(5 marks)*

A reasonable guess is $f(n) = n^2$**.**
In the base case we have $f(1) = 1^2 = 1 = C(1)$**.**
In the inductive step we assume that $C(n-1) = f(n-1) = (n-1)^2$ **then**

$$\begin{aligned} C(n) &= (n-1)^2 + 2\,n - 1 \\ &= (n^2 - 2\,n + 1) + 2\,n - 1 \\ &= n^2 = f(n). \end{aligned}$$

(e) Sketch the curve $\log_{10}(C(n))$ versus $\log_{10}(n)$ on the graph below. *(2 marks)*

TURN OVER



(f) If it takes 100s to compute `foo(1000)` approximately how long will it take to compute `foo(2000)`? *(2 marks)*

We assume that the time to run `foo(n)` **is** $T(n) \approx c\,n^2$**, where** c **is some constant we have to estimate empirically. Using the fact that running** `foo(1000)` **takes 100s, i.e.** $T(1000) = 100 \approx c \times 1000^2$ **so that** $c \approx 10^{-4}$**. The time to run** `foo(2000)` **is** $T(2000) \approx 10^{-4} \times 2000^2 = 400s$**.**

---

3 Consider the program (valid for inputs $n \geq 1$)

```
foo(int n) {
   bar();
   if (n==1)
      return;
   int m = (int) n/2
   foo(m);
}
```

where `(int) n/2` returns the greatest integer less than or equal to n/2 (i.e. $\lfloor n/2 \rfloor$).

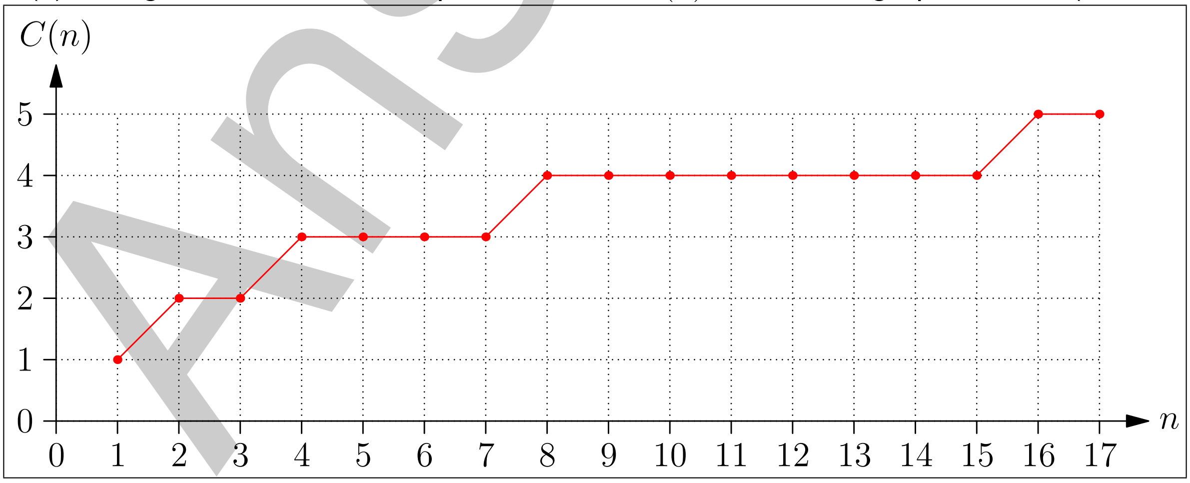
(a) Let $C(n)$ be the number of times the function `bar()` is called when we call `foo(n)`. Write down a recurrence relation for $C(n)$. *(2 marks)*

$$C(n) = C(\lfloor n/2 \rfloor) + 1$$

(b) Write down the boundary condition for the recurrence relation. *(1 marks)*

$$C(1) = 1$$

(c) Using the recurrence compute values of $C(n)$ to draw the graph below. *(4 marks)*



(d) Prove by induction that $f(n) = \lfloor \log_2(n) \rfloor + 1$ satisfies the recurrence relation for $C(n)$. This is simplified if we perform the inductive step over the set of integers $S_m = \{2^m, 2^m + 1, \ldots, 2^{m+1} - 1\}$. *(6 marks)*

(This is hard—harder than an examination question.)

In the base case we have $f(1) = \lfloor \log_2(1) \rfloor + 1 = 1 = C(1)$**. This is true for the set** $S_0 = \{1\}$**.**
In the inductive step we assume that for $n \in S_{m-1}$ **we have** $C(n) = f(n) = m$ **then for** $n \in S_m$ **we have**

$$C(n) = C(\lfloor n/2 \rfloor) + 1$$

but if $n \in S_m$ **then** $\lfloor n/2 \rfloor \in S_{m-1}$ **so** $C(\lfloor n/2 \rfloor) = m$**. Thus** $C(n) = m + 1 = f(n)$**.**

(e) If it takes 100s to compute `foo(512)` approximately how long will it take to compute `foo(1024)`? *(2 marks)*

We assume that the time to run `foo(n)` **is** $T(n) \approx c\,(\log_2(n) + 1)$**, where** c **is some constant we have to estimate empirically. Using the fact that running** `foo(512)` **takes 100s, then** $T(512) = 100 \approx c \times (\log_2(512) + 1) = 10\,c$ **so that** $c \approx 10$**. The time to run** `foo(1024)` **is** $T(1024) \approx 10 \times (\log_2(1024) + 1) = 10 \times 11 = 110s$**.**

END OF PAPER