Q1)

1. f(n) = f(n-1) + f (n-2)

Base cases:

f(1) = 1

f(2) = 2

Induction Hypothesis:

Let f(k) be true for $1 \leq k \leq n$

Induction Step:

$n+1^{th}$ step can be reached either by taking 1 step from nth step or by taking 2 steps from n-1th step.

Therefore: f(n+1) = f(n) + f(n-1)

f(n) and f(n-1) are true by induction hypothesis

Hence proved

3. f(n) – f(n-1) = f(n-2)
   f(n-1) – f(n-2) = f(n-3)
   .
   .
   .
   .
   f(4)-f(3) = f(2)
   f(3)-f(2) = f(1)

   Adding both RHS and LHS for all these equations we get

   f(n) – f(2) = $\sum_1^{n-2} f(n)$
   f(n) = $\sum_1^{n-2} f(n)$ + f(2)

   We know that f(2) = 2
   So, (n) = $\sum_1^{n-2} f(n)$ + 2

   Hence, Proved

Q2)

$f(n) = 2*f(n \text{ div } 10) + f(n \bmod 10)$ $\quad n \geq 10$

$= n$ $\quad n < 10$

Induction Hypothesis:

Let f(k) be true for $1 \leq k \leq n$

Induction Step:

If n is a k+1 digit number of the form $d_k d_{k-1} \ldots\ldots d_1 d_0$, then n div 10 is a k digit number

$f(n \text{ div } 10) = 2^{k-1}d_k + \ldots\ldots + 2^1 d_2 + 2^0 d_1$

$f(n) = 2*f(n \text{ div } 10) + f(n \bmod 10)$

$= 2^k d_k + \ldots\ldots + 2^2 d_2 + 2^1 d_1 + d_0$

Hence Proved

Q3)

```
fun checkFinal(a,b,n) = if a>n then false
                        else if b>a then checkFinal(a+1,0,n)
                        else if (a*a + b*b = n) then true
                        else checkFinal(a,b+1,n);

fun check(a) = checkFinal(0,0,a);

fun count(a,b) = if a>b then 0
                 else if check(a) then 1 + count(a+1,b)
                 else count(a+1,b);

fun squaredCount(n) = count(1,n);
```

Correctness Proof:

1) squaredCount(n): It just returns the value passed to it by count(a,b) function.
2) count(a,b): it counts how many times from a to b is the property given by function check(a) satisfied.

   It iterates b-a +1 times, and if check(a) is true then increments the count else just goes for the next iteration.

3) check(a): It just returns the Boolean value passed on to it by checkFinal(a,b,n).
   It is used to return if the number 'a' passed to it satisfies the property to be checked by checkFinal.
4) checkFinal(a,b,n): It is used to determine if n can be written as the sum of squares of two natural numbers

   while a<n it checks whether $a^2 + b^2 = n$ by keeping 'a' constant and incrementing b till it is less than a and checking the condition each time and when b becomes equal to a, it increments a by 1 and resets the count of b to 0 and reiterates till either the condition is satisfied or a becomes equal to n.

   If at any point $a^2 + b^2$ becomes equal to n, it returns true to the function call else it finally return false if the iteration ends with no case satisfying the condition.

Q4)

```
fun c(n)=2*n*(2*n + 1)*(2*n +2);

fun series(b)= 4.0/real(c(b));

fun nilkantha(a:real,b) = if series(b) < a
                          then series(b)
                          else series(b) - nilkantha(a,b+1);

fun nilakanthaSum(a:real)= 3.0 + nilkantha(a,1);
```

Correctness Proof:

1) nilakanthaSum: it accepts a real number a and returns 3 + nilkantha(a,1)
2) nilkantha(a,b): It accepts 1 real number and one ineger as an input.

   It checks whether series(b) is smaller than 'a' and until it is not, it keeps reiterating, incrementing b each time by 1 till series(b) becomes smaller than 'a'.

   It returns series(b) - (series(b+1) – (series(b+2) – (…… - series (k)))) where series(k) is the first term smaller than a.
3) series(b): It calculates 4/c(b)
4) c(n) : It calculates 2*n*(2*n + 1)*(2*n +2)