ICCS200: Assignment homework-number-3

Poonnarat Nakartit poonpptn@gmail.com
Recitation: Your recitation section
The date

Exercise 1: Tail Sum of Squares (2 points)

```
int sumHelper(int n , int a) {
if (n==0) return a;
else returnsumHelper(n-1,a+n*n);
}
int sumSqr(int n) { return sumHelper(n, 0); }
```

Prove that for $n \geq 1$, `sumSqr(n)` is $1^2 + 2^2 + 3^2 + ... + n^2$. To prove this, use induction to show that `sumHelper` computes the "right thing".

- Predicate: P(n):= $\forall n$ and $\forall a \in \mathbb{I}^+$+set of zero and $n > 0$ `sumhelper(int n, int a)` returns $1^2 + 2^2 + 3^2 + ... + n^2 + a$.

- Base case: P(0) `sumhelper(0, int a)` returns a

- Inductive step: Assume that $\forall k \leq n$ where $k \in \mathbb{I}^+$ `sumhelper(int , int a)` returns $1^2 + 2^2 + 3^2 + ... + k^2 + a$.
  Want to show that `sumhelper(k+1,a')` returns $1^2 + 2^2 + 3^2 + ... + k^2 + a + (k+1)^2 + a'$.

  Since $k + 1 > 0$ `sumhelper(k+1,a')` return `sumhelper(k,a)` returns `sumhelper(k, a'+ (k+1)*(k+1))`.
  Because of we know that `sumhelper(k, a)` returns $1^2+2^2+3^2+...+k^2+a+(k+1)^2+a$. Where a is equal to `a'+ (k+1)*(k+1)` ,so $1^2 + 2^2 + 3^2 + ... + k^2 + a + (k + 1)^2 + a'$ by inductive hypothesis. And we have just established that for $n > 0$, if P(k) is true, then P(k+1)is true.

Exercise 2: Mysterious Function (2 points)

- Predicate: P(n) := $n \geq 1$, if `foo(n) returns (p, q)`, then

$$\frac{p}{q} = \frac{n}{n+1}$$

- Base case: P(1) = foo(1) returns (1,2), so $\frac{1}{2} = \frac{1}{1+1}$

- Inductive step: Assume that$\forall k <= n$. P(k)= `foo(k) returns (p,q), which is equal to` $\frac{p}{q} = \frac{n}{n+1}$
  Want to show that P(k+1) = `foo(k+1)` = $\frac{p'}{q'} = \frac{k+1}{k+2}$

Since the k +1 is more than 1, the algorithm returns `foo(k)`, which is the inductive hypothesis. We will get $\frac{p}{q} = \frac{n}{n+1}$, and foo(k+1) returns (p',q'), which p' is equal to $(k+1) + k*(k+1)*(k+2)$, and q' is equal to $(k+1)*(k+1)*(k+2)$

Want to show that : $\frac{p'}{q'} == \frac{k+1}{k+2}$

$$\begin{aligned}
\frac{p'}{q'} &= \frac{(k+1)+k*(k+1)*(k+2)}{(k+1)*(k+1)*(k+2)} \\
&= \frac{k+1}{(k+1)*(k+1)*(k+2)} + \frac{k*k+1*k+2}{(k+1)*(k+1)*(k+2)} \\
&= \frac{1}{(k+1)(k+2)} + \frac{k}{k+1} \\
&= \frac{k(k+2)+1}{(k+1)(k+2)} \\
&= \frac{k^2+2k+1}{(k+1)(k+2)} \\
&= \frac{(k+1)(k+1)}{(k+1)(k+2)} \\
\frac{p'}{q'} &= \frac{k+1}{k+2}
\end{aligned}$$

By mathematical induction, $n \geq 1$, if `foo(n) returns (p, q)`, then

$$\frac{p}{q} = \frac{n}{n+1}$$

is true.

Exercise 3: Missing Tile (4 points)

Task I: We know youve seen this proof already. But so that you fully understand how it works, youll prove this theorem again, in your own words, by induction on n.

- Predicate: P(n):= Any where$b \leq 2$. $2^n - by - 2n$ grid with one painted cell can be tiled using $L - \textit{shaped}$ triominoes such that the entire grid is covered by triominoes but no triominoes overlap with each other nor the painted cell.
- Base cases: P(2):=

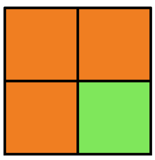

Figure 1: The green color is the painted grid, orenge is the l-shape. The other basecases are just the oreantation of this figure

- Inductive step: = Assume that $\forall k \leq n$ P(k):= Any where$b \leq 2$. $2^k - by - 2k$ grid with one painted cell can be tiled using $L - \textit{shaped}$ triominoes such that the entire grid is covered

by triominoes but no triominoes overlap with each other nor the painted cell.
Want to show that: P(k+1):= $2^{k+1} - by - 2k + 1$ grid with one painted cell can be tiled using $L - shaped$ triominoes such that the entire grid is covered by triominoes but no triominoes overlap with each other nor the painted cell.
This $2^{k+1} - by - 2^{k+1}$ can be break down into four pieces of size 2×2^k

$$\begin{aligned} 2^{2+1} \times 2^{2+1} &= \\ 2^2 \times 2^k \times 2^k &= 4 \times 2^k \times 2^k \end{aligned}$$

Since we know that $2^k is hold by inductive hypothesis. By matematical induction, Anywhere$b ≤ 2. $2^n - by - 2n$ grid with one painted cell can be tiled using $L - shaped$ triominoes such that the entire grid is covered by triominoes but no triominoes overlap with each other nor the painted cell.

Exercise 5: Midway Tower of Hanoi (6 points)

Task II: Solve the following inputs by hand. They are presented in the same format as described above. Document the reasoning you use to reach the answers. You want a process that will work with larger inputs.

$\{2, 2, 1\} \to \{0, 2, 1\} \to \{0, 1, 1\} \to \{1, 1, 1\}$

$\{2, 1, 0\} \to \{0, 1, 0\} \to \{0, 2, 0\} \to \{2, 2, 0\} \to \{2, 2, 1\} \to \{0, 2, 1\} \to \{0, 1, 1\} \to \{1, 1, 1\}$

My Logic is just trying to move the bigest disk in place to do that, I need to move the smaller $n-1$ disks first, which It is $2^{n-1} - 1$ moves. so I can move the biggest one in place,add 1 move, and moves the others back to the destination peg using $2^{n-1} - 1$ moves.

Task III: Prove, using induction, that for any $n \geq 0$, `solve_hanoi(n, ...)`generates exactly $2^n - 1$ lines of instructions.

```
def solve_hanoi(n, from_peg, to_peg, aux_peg):
       if n>0:
           solve_hanoi(n-1, from_peg, aux_peg, to_peg)
           print "Move disk", n-1, "from Peg", from_peg, "to Peg", to_peg
           solve_hanoi(n-1, aux_peg, to_peg, from_peg)
solve_hanoi(n, 0, 1, 2)
```

- Predicate: P(n):= `solve_hano(n, from_peg, to_peg, aux_peg)` use $2^n - 1$ lines of instructions.
- P(1):= `solve_hano(1, from_peg, to_peg, aux_peg)` use $2^1 - 1 = 1$
- Inductive step: Assume that $\forall k <= n$ P(k):= `solve_hano(k, from_peg, to_peg, aux_peg)` use $2^k - 1$ lines of instructions.
  Want to show that P(k+1):= `solve_hano(k+1, from_peg, to_peg, aux_peg)` use $2^{k+1} - 1$ lines of instructions.
  In the first recursive call, `solve_hano(k, from_peg, to_peg, aux_peg)` performs 2^{k-1}

Poonnarat Nakartit
poonpptn@gmail.com

lines of instructions as well as the second the second call by inductive hypothesis. and perfrom one line of instruction on the print stagement.

$$(2^k - 1) + 1 + (2^k - 1) = 2 * 2^k - 1 = 2^{k+1} - 1$$

by mathematical induction texttsolve_hano(n, from_peg, to_peg, aux_peg) use $2^n - 1$ lines of instructions. P(1):= `solve_hano(1, from_peg, to_peg, aux_peg)` use $2^1 - 1 = 1$ is true.