Problem 1

• Consider the function

fun foo (a:string) = (fn (b:string) => (a,b));

• What is the value and type of the following computations:
  1. foo “100” “101”;
  2. val q = foo "happy"; q "really happy";

• Rewrite this function in the abbreviated curried style.
Solution 1

• Consider the function

```haskell
fun foo (a:string) = (fn (b:string) => (a,b));
```

• What is the value and type of the following computations:
  1. `foo “100” “101”`
     
     `ans: val it = ("100","101"):string*string`
  2. `val q = foo "happy"; q "really happy"`
     
     `ans: val q = fn: string -> string*string
     val it = ("happy","really happy"):string*string`

• Rewrite this function in the abbreviated curried style.

```haskell
ans: fun foo (a:string) (b:string) = (a,b);
```
Problem 2

• Convert the following function

\[
\text{fun pow}(b, m) = \text{if } m = 0 \text{ then } 1 \text{ else } b * \text{pow}(b, m-1);
\]

1. to a function using patterns
2. to a function using currying
3. to function using patterns and currying
Solution 2

• Convert the following function

\[
\text{fun pow}(b, m) = \text{if } m = 0 \text{ then } 1 \text{ else } b^*\text{pow}(b, m-1);
\]

1. to a function using patterns
   ans: fun pow(_,0) = 1
   \hspace{1cm} | pow(b,m) = b^*\text{pow}(b,m-1);

2. to a function using currying
   ans: fun pow b = (fn m => if m=0 then 1 else b^*(pow b (m-1)));

3. to function using patterns and currying
   ans: fun pow _ 0 = 1
   \hspace{1cm} | pow b m = b^*(pow b (m-1));
Problem 3

• Write a curried function `hdmap` that takes a function and a list of integers and applies the function to the first element of the list. If the list is empty return ~1,

\[
\text{hdmap} = \text{fn} : (\text{int} \rightarrow \text{int}) \rightarrow \text{int list} \rightarrow \text{int}
\]

• Show that your function works by computing:
\[
\text{hdmap} (\text{fn} \ x \rightarrow x + 1) \ [3,4]
\]
Solution 3

• Write a curried function `hdmap` that takes a function and a list of integers and applies the function to the first element of the list. If the list is empty return ~1,

  \[
  \text{hdmap} = \text{fn} : (\text{int} \rightarrow \text{int}) \rightarrow \text{int list} \rightarrow \text{int}
  \]

  \[
  \text{ans: fun hdmap \_ \[_] = \sim 1} \\
  | \text{hdmap } f (a::L) = f \ a;
  \]

• Show that your function works by computing: `hdmap (fn x => x + 1) [3,4]`

  \[
  \text{ans: val it = 4: int}
  \]